Finding Influential Cores via Normalized Ricci Flows in Directed and Undirected Hypergraphs with Applications

Prithviraj Sengupta* and Nazanin Azarhooshang†
Department of Computer Science,
University of Illinois Chicago
Chicago, IL 60607, USA

Réka Albert‡
Department of Physics
Pennsylvania State University
University Park, PA 16802, USA

Bhaskar DasGupta§
Department of Computer Science
University of Illinois Chicago
Chicago, IL 60607, USA
(Dated: April 7, 2025)

Many biological and social systems are naturally represented as edge-weighted directed or undirected hypergraphs since they exhibit *group* interactions involving *three or more* system units as opposed to pairwise interactions that can be incorporated in graph-theoretic representations. However, finding influential cores in hypergraphs is still not as extensively studied as their graph-theoretic counter-parts. To this end, we develop and implement a hypergraph-curvature guided discrete time diffusion process with suitable topological surgeries and edge-weight re-normalization procedures for both undirected and directed weighted hypergraphs to find influential cores. We successfully apply our framework for directed hypergraphs to seven metabolic hypergraphs and our framework for undirected hypergraphs to two social (co-authorship) hypergraphs to find influential cores, thereby demonstrating the practical feasibility of our approach. In addition, we prove a theorem showing that a certain edge weight re-normalization procedure in a prior research work for Ricci flows for edge-weighted graphs has the undesirable outcome of modifying the edge-weights to negative numbers, thereby rendering the procedure impossible to use. To the best of our knowledge, this seems to be one of the first articles that formulates algorithmic approaches for finding core(s) of (weighted or unweighted) directed hypergraphs.

I. INTRODUCTION

Useful insights for many complex systems are often obtained by representing them as graphs and analyzing them using graph-theoretic and combinatorial tools [1–3]. Such graphs may vary in diversity from simple undirected graphs to edge-labeled directed graphs. In such graphs, nodes represent the basic units of the system (*e.g.*, system variables) and edges represent relationships (*e.g.*, correlations) between pairs of such basic units. Once such graphs are constructed, they can be analyzed using graph-theoretic measures such as degree-based measures (*e.g.*, degree distributions), connectivity-based measures (*e.g.*, clustering coefficients), geodesic-based measures (*e.g.*, betweenness centralities) and other more novel network measures such as in [4–7] to give meaningful insights into the properties and the dynamics of the system.

However, many real-world systems exhibit *group* (*i.e.*, higher order) interactions involving three or more system units [8–10]. One way to handle these higher order interactions is to encode them by a suitable combination of pairwise interactions and then simply use the existing graph-theoretic tools (*e.g.*, see [11, 12]). While such approaches have been successful in the context of many real-world networks, they obviously do not encode the higher-order interactions in their full generalities. A more direct approach would be to use *directed* or *undirected* hypergraphs to encode these interactions, and this is the approach we follow in this article. Although the theory of hypergraphs has been considerably developed during the last few decades (*e.g.*, see [13]), applications of hypergraphs to real-world networks face their own challenges. Sometimes it is not clear how to generalize a concept from graphs to hypergraphs such that it best serves its purpose in the corresponding application, and some computationally tractable graph-theoretic algorithms may become intractable when generalized to hypergraphs (*e.g.*, the maximum matching problem for graphs is polynomial-time solvable whereas the 3-dimensional matching problem for hypergraphs is NP-complete [14]).

Suitable notions of curvatures are natural measures of shapes of higher dimensional objects in mainstream

* prithvi1096@gmail.com; www.linkedin.com/in/prithviraj-sengupta/
† nazanin.azarhoushang@gmail.com; www.linkedin.com/in/nazaninazarhooshang/
‡ rza1@psu.edu; www.ralbert.me
§ bdasgup@uic.edu; http://bdasgup.github.io/

physics and mathematics [15, 16]. There have been several attempts to extend these curvature measures to graphs and hypergraphs. Two major notions of curvatures of graphs can be obtained via extending Forman's discretization [17] of Ricci curvature for (polyhedral or CW) complexes to undirected graphs (the "*Forman-Ricci curvature*") [18–23] and via Ollivier's discretization of manifold Ricci curvature to undirected edge-weighted graphs (the "*Ollivier-Ricci curvature*") [24–27]. Both Ollivier-Ricci curvature and Forman-Ricci curvature assign a number to each edge of the given graph, but the numbers are calculated in very *different* ways since they capture *different* metric properties of a Riemannian manifold; some comparative analysis of these two measures can be found in [22, 23]. Recently, the Ollivier-Ricci curvature and the Forman-Ricci curvature measures have been generalized in a few ways to unweighted directed and undirected hypergraphs [28–32].

A curvature-guided diffusion process called Ricci flow, along with "topological surgery" procedures to avoid topological singularities, was originally introduced in the context of a Riemannian manifold by Hamilton [33] to provide a continuous change of the metric of the manifold to provide a continuous transformation (homeomorphism) of one manifold to another manifold. One of the most ground-breaking application of this technique was done by Perelman [34] to solve the Poincaré conjecture, which asserts that any three-dimensional manifold that is closed, connected, and has a trivial fundamental group is homeomorphic to the three-dimensional sphere. In the context of a weighted undirected graphs, these techniques were extended in papers such as [20, 35–40] to iteratively and synchronously change the weights of the edges of the graph. Motivated by the fact that connected sum decomposition can be detected by the geometric Ricci flows in manifolds, these techniques were then used to find communities or modules mostly in the context of *undirected social graphs*. To prevent lack of convergence of graph Ricci flows within reasonable time, edge-weight re-normalization methods after every iteration were suggested and investigated in [37].

In this article we devise a computational framework to detect *influential cores* in *both* undirected and directed weighted hypergraphs by formulating and using a hypergraph-curvature guided discrete time diffusion process with suitable topological surgeries and edge-weight re-normalization procedures. We demonstrate the practical feasibility of our approach by successfully applying our computational framework for directed hypergraphs to seven metabolic hypergraphs and our computational framework for undirected hypergraphs to two social (co-authorship) hypergraphs to find influential cores. In addition, we prove a theorem showing that a certain edge weight re-normalization procedure in [37] for Ricci flows for edge-weighted graphs has an undesirable outcome of modifying the edge-weights to negative numbers, thereby rendering the procedure *impossible* to use.

a. Motivation for finding core(s) of a complex system Broadly speaking, the core of a complex system (also studied under the name "cohesive subgraph" in the network science literature [41]) is a smaller sub-system that contributes *significantly* to the functioning of the overall system, and thus focussing the analysis of the smaller sub-system, which could be *easier* than analyzing the system as a whole, may reveal important characteristics of the overall system. For example, in the context of brain graphs, identifications of core(s) of the graph where neurons strongly interact with each other provide effective characterizations of these graph topologies; such cores are very important for various brain functions and cognition [42]. As another example, cores in attributed social networks can be directly utilised for a recommendation system [41].

For systems represented by *undirected graphs*, prior research works have used various definitions of what actually constitutes a core, such as via modularity [43], via rich clubs [44], or using information-theoretic ideas [45].

Our definition for cores of hypergraphs requires sufficient connectivity and cohesiveness, nontrivial size, and a large centrality, evidenced by a large loss of short paths in the network when the core is removed. The specific definition and constraints are given in Section II C. The specific usefulness of finding cores for metabolic (directed) hypergraphs and for co-authorship (undirected) hypergraphs are discussed in Section III E 1 and in Section III E 2, respectively. **To the best of our knowledge, this seems to be one of the first articles that formulates algorithmic approaches for finding core(s) of (weighted or unweighted) directed hypergraphs**. Note that since graphs are special cases of hypergraphs, our methodologies are also applicable to graphs; however, in this article our focus is on hypergraphs that are *not* graphs.

b. Finding core(s) vs. modular decomposition Note that finding a core of a system is *different* than the modular decomposition of graphs that has been extensively studied in the network science literature [46–50]: the overall goal of graph decomposition (partitioning) into modules (also called communities or clusters) is to partition the *entire* node set into modules and requires the optimization of a *joint* fitness function of these modules to evaluate the quality of the decomposition. In particular, the centrality parameters (quantifying loss of short paths when removing the core, see Section II C) are *not* relevant for typical modular decomposition applications and the size constraints are unimportant if the joint fitness function is satisfactory (*e.g.*, papers such as [49] show that a good approximation to Newman's modularity value may be obtained even though the corresponding modules will *not* satisfy our size constraints).

A. Basic Definitions and Notations

A weighted *directed* hypergraph $H = (V, E, w)$ consists of a node set V, a set E of (directed) hyperedges and a hyperedge-weight function $w : E \mapsto \mathbb{R}_{\geq 0}$. A *directed* hyperedge $e \in E$ is an ordered pair $(\mathcal{T}\text{ail}_e, \mathcal{H}\text{ead}_e)$ where $\emptyset \subset \mathcal{T}\text{ail}_e \subset V$ is the *tail*, $\emptyset \subset \mathcal{H}\text{ead}_e \subset V$ is the *head* and $\mathcal{T}\text{ail}_e \neq \mathcal{H}\text{ead}_e$; for convenience we will also denote the hyperedge by $\mathcal{T}\text{ail}_e \rightarrow \mathcal{H}\text{ead}_e$. For a node $x \in V$, the *in-degree* deg_x^{in} is the number of "incoming" hyperedges, *i.e.*, the number of hyperedges e' such that $x \in \mathcal{H}\text{ead}_{e'}$, and the *out-degree* deg_x^{out} is the number of "outgoing" hyperedges, *i.e.*, the number of hyperedges e' such that $x \in \mathcal{T}\text{ail}_{e'}$. A (directed) *path* $\mathscr{P}_{x,y}$ from node x to node y is an alternating sequence $(x = v_1, e_1, \ldots, v_k, e_k, v_{k+1} = y)$ of *distinct* nodes and directed hyperedges such that $v_i \in \mathcal{T}\text{ail}_{e_i}$ and $v_{i+1} \in \mathcal{H}\text{ead}_{e_i}$ for each $i \in \{1, \ldots, k\}$; the *length* of the path is $\sum_{i=1}^{k} w(e_i)$. We will denote by $\text{dist}_H(u, v)$ the *minimum* length of any path from u to v; note that $\text{dist}_H(u, v)$ need *not* be same as $\text{dist}_H(v, u)$. A directed hypergraph is *weakly connected* provided for every pair of nodes x and y there is *either* a path from x to y *or* a path from y to x. **We assume from now onwards that our original (input) directed hypergraph is weakly connected**.

A weighted *undirected* hypergraph $H = (V, E, w)$ consists of a node set V, a set E of (undirected) hyperedges and a hyperedge-weight function $w : E \mapsto \mathbb{R}_{\geq 0}$. A (undirected) hyperedge $e \in E$ is a subset of nodes $\mathcal{A}_e$ where $\emptyset \subset \mathcal{A}_e \subseteq V$. For a node $x \in V$, the *degree* deg_x is the number of the number of hyperedges e' such that $x \in \mathcal{A}_{e'}$. An (undirected) *path* $\mathscr{P}_{x,y}$ between nodes x and y is an alternating sequence $(x = v_1, e_1, \ldots, v_k, e_k, v_{k+1} = y)$ of *distinct* nodes and undirected hyperedges such that $v_i, v_{i+1} \in \mathcal{A}_{e_i}$ for each $i \in \{1, \ldots, k\}$; the *length* of the path is $\sum_{i=1}^{k} w(e_i)$. We will denote by $\text{dist}_H(u, v)$ the *distance* (*i.e.*, *minimum* length of any path) between u and v. A undirected hypergraph is *connected* provided for every pair of nodes there is a path between them. **We assume from now onwards that our original (input) undirected hypergraph is connected**.

B. Discussions on Prior Relevant Research

There are some prior published articles that deal with finding cores or core-like structures for *undirected* hypergraphs, *mostly* for unweighted undirected hypergraphs [51–60] but some also on weighted undirected hypergraphs [61]. However, there seem to be very few peer-reviewed prior articles dealing with finding cores for *directed* (and more so for *weighted* directed) hypergraphs where cores are defined in the same sense as used in this article; the authors themselves were unable to get any relevant peer-reviewed prior works via google search. For example, the articles by Pretolani [62] and by Volpentesta [63] investigate intriguing but *different* concepts that refer to sub-structures in unweighted directed hypergraphs based on hyperpaths. Note that although for graphs replacing an undirected edge by two directed edges make the core finding problem for undirected graphs solvable from the core finding problem for directed graphs, a similar trick cannot directly be used for hypergraphs since the head or tail may contain more than one node (*i.e.*, a core finding algorithm for directed hypergraphs does not readily translate to an algorithm for finding cores in undirected hypergraphs). One possible way to get around the difficulty would be to "average out" over all possible combinatorial ways of choosing direction of a hyperedge to convert an undirected (weighted or unweighted) edge to a set of directed weighted edges. In other words, an undirected hyperedge e is replaced by a set of directed hyperedges $T_e = \{e_S \,|\, \emptyset \subset S \subset \mathcal{A}_e\}$ where $\mathcal{T}\text{ail}_{e_S} = S$, $\mathcal{H}\text{ead}_{e_S} = \mathcal{A}_e \setminus S$, and $\sum_{\emptyset \subset S \subset \mathcal{A}_e} w(e_S) = w(e)$. However, this approach has at least two potential problems. First, this approach leads to an obvious combinatorial explosion since a single undirected hyperedge e is replaced by a set of $2^{|\mathcal{A}_e|} - 2$ directed edges. Second, it may not be a priori clear how one should select the values of $w(e_S)$ for each individual hyperedge e_S, *e.g.*, whether the weight should be the same for each e_S or if it should depend on the value of $|\mathcal{T}\text{ail}_{e_S}| - |\mathcal{H}\text{ead}_{e_S}|$. A strength of our framework is that we have a single overall algorithmic method (albeit with different calculations of certain quantities) that can be adopted for *both* directed and undirected hypergraphs.

Most of the prior works on finding cores for undirected unweighted hypergraphs involve iteratively identifying a set of nodes such that the degree of every node in the set is at least k for a given k. We go deeper than these prior works by defining the centrality quality parameters of the core as stated via Equations (12) and (13). These centrality quality parameters, which are combinatorial analogs of the information loss quantifications used in prior research works such as [45], are significant in real-world applications to biological and social networks, as mentioned in prior publications such as [64] in the context of undirected unweighted graphs (*e.g.*, see Figure 5 for biological networks and Figures 7–8 for social networks in [64] with associated texts), and are closely related to the concept of structural holes [65] in social networks. Another crucial advantage of our approach over most prior approaches is scalability via easy parallelization. Since within each iteration the calculations of the EMD values for the hyperedges can be done in parallel in the algorithmic framework of Table I, a clustered computing environment could be used for a direct linear speedup, *i.e.*, using a cluster of c computing nodes would result in a speedup by about a factor c. This makes our method more suitable for larger hypergraphs in a networked computing environment.

II. METHODS AND MATERIALS

A. Definition of Curvatures of Weighted Hypergraphs

Curvatures of weighted hypergraphs are defined in somewhat different ways depending on whether the hypergraph is directed or undirected. However, both definitions use a common paradigm of EMD (*Earth Mover's Distance*, also known as the L_1 *transportation distance*, the L_1 *Wasserstein distance* or the *Monge-Kantorovich-Rubinstein distance* [66–69]) defined on the hypergraph in the following manner based on the notations and terminologies in [70]. Let $H = (V, E, w)$ be a directed or undirected hypergraph. Suppose that we have two probability distributions $\mathbb{P}_{\text{left}}$ and $\mathbb{P}_{\text{right}}$ over the set of nodes V, *i.e.*, two real numbers $0 \leq \mathbb{P}_{\text{left}}(v), \mathbb{P}_{\text{right}}(v) \leq 1$ for every node $v \in V$ with $\sum_{v \in V} \mathbb{P}_{\text{left}}(v) = \sum_{v \in V} \mathbb{P}_{\text{right}}(v) = 1$. We can think of $\mathbb{P}_{\text{left}}(v)$ as the total amount of "earth" (dirt) at node v that need to be moved to other nodes, and $\mathbb{P}_{\text{right}}(v)$ as the *maximum* total amount of earth node v can store. The cost of transporting *one* unit of earth from node u to node v is $\text{dist}_H(u, v)$, and the goal is to move *all* units of earth (determined by $\mathbb{P}_{\text{left}}$) while simultaneously satisfying *all* storage requirements (dictated by $\mathbb{P}_{\text{right}}$) and *minimizing* the total transportation cost. Letting the real variable $z_{u,v} \in [0, 1]$ denote the amount of shipment from node u to node v in an optimal solution, EMD for the two probability distributions $\mathbb{P}_1$ and $\mathbb{P}_2$ on V is the *linear programming* (LP) problem shown in Fig. 1 which can be solved in polynomial time. We will use the notation $\text{EMD}_H(\mathbb{P}_{\text{left}}, \mathbb{P}_{\text{right}})$ to denote the value of the objective function in an optimal solution of the LP in Fig. 1.

Given a hypergraph H and an edge e of H, the curvature value of e is then computed as follows:

- ▷ Fix appropriate distributions for $\mathbb{P}_{\text{left}}$ and $\mathbb{P}_{\text{right}}$.
- ▷ Use a formula for Ricci curvature of the hyperedge e The formula is *different* depending on whether the hypergraph is directed or undirected, and shown below:
  - ▷ For directed hypergraphs, the Ricci curvature of the hyperedge e is calculated as:

$$\boxed{\text{RIC}(e) = 1 - \frac{\text{EMD}_H(\mathbb{P}_{\text{left}}, \mathbb{P}_{\text{right}})}{w(u, v)}} \tag{1}$$

    An informal intuitive understanding of the connection of EMD to Ricci curvature in the above formula, as explained in prior research works such as [24] in the context of graphs, is as follows. The Ricci curvature at a point x in a smooth Riemannian manifold can be thought of transporting a small ball centered at x along that direction and measuring the "distortion" of that ball. In (1) the role of the direction is captured by the hyperedge (u, v), the roles of the balls at the two nodes are played by the distributions $\mathbb{P}_{\text{left}}$ and $\mathbb{P}_{\text{right}}$, and the role of the distortion due to transportation is captured by the EMD measure.

  - ▷ For undirected hypergraphs, the Ricci curvature of the hyperedge e is calculated as:

$$\boxed{\text{RIC}(e) = 1 - \frac{1}{\binom{|\mathcal{A}_e|}{2}} \times \sum_{\substack{p,q \in \mathcal{A}_e \\ p \neq q}} \text{EMD}_H(\mathbb{P}^p_{\text{left}}, \mathbb{P}^q_{\text{right}})} \tag{2}$$

    The calculation for the curvature averages out weighted-lazy random walk probabilities over all pairs of distinct nodes in e. There is one special case not covered by the above definition but may occur in our undirected co-authorship hypergraphs: namely when $\mathcal{A}_e = \{u\}$ for some node u corresponding to a paper written by *just one* author. For this case we treat the hyperedge as a self-loop from u to u giving an EMD value of zero.

The exact calculations of $\mathbb{P}_{\text{left}}$ and $\mathbb{P}_{\text{right}}$ are somewhat different depending on whether the hypergraph is directed or undirected, and this is described in next two sections. Let $H = (V, E, w)$ be the weighted (directed or undirected) hypergraph, and $e \in E$ be the hyperedge considered.

1. *Calculations of $\mathbb{P}_{\text{left}}$ and $\mathbb{P}_{\text{right}}$ for a Weighted Directed Hypergraph*

The distributions $\mathbb{P}_{\text{left}}$ and $\mathbb{P}_{\text{right}}$ are determined by the nodes in $\mathcal{T}\text{ail}_e$ and $\mathcal{H}\text{ead}_e$, respectively. $\mathbb{P}_{\text{left}}$ is determined in the following manner by adopting the calculations in [29]:

- ▷ Initially, $\mathbb{P}_{\text{left}}(u) = 0$ for all $u \in V$. In our subsequent steps, we will add to these values as appropriate.
- ▷ We divide the total probability 1 equally among the nodes in $\mathcal{T}\text{ail}_e$, thus "allocating" a value of $(|\mathcal{T}\text{ail}_e|)^{-1}$ to each node in question.
- ▷ For every node $x \in \mathcal{T}\text{ail}_e$ with $\mathsf{deg}^{in}_x = 0$, we add $(|\mathcal{T}\text{ail}_e|)^{-1}$ to $\mathbb{P}_{\text{left}}(x)$.
- ▷ For every node $x \in \mathcal{T}\text{ail}_e$ with $\mathsf{deg}^{in}_x > 0$, we perform the following:
  - ▷ We divide the probability $(|\mathcal{T}\text{ail}_e|)^{-1}$ equally among the hyperedges e' such that $x \in \mathcal{H}\text{ead}_{e'}$, thus "allocating" a value of $(|\mathcal{T}\text{ail}_e| \times \mathsf{deg}^{in}_x)^{-1}$ to each hyperedge in question.
  - ▷ For each such hyperedge e' such that $x \in \mathcal{H}\text{ead}_{e'}$, we divide the allocated value equally

variables: $z_{u,v}$ for every pair of nodes $u, v \in V$

minimize $\sum_{u \in V} \sum_{v \in V'} \text{dist}_H(u,v)\, z_{u,v}$ (* minimize total transportation cost *)

subject to

$\sum_{v \in V} z_{u,v} = \mathbb{P}_{\text{left}}(u)$, for each $u \in V$ (* ship from u as much as it has *)

$\sum_{u \in V} z_{u,v} = \mathbb{P}_{\text{right}}(v)$, for each $v \in V$ (* ship to v as much as it can store *)

$z_{u,v} \geq 0$, for each $u, v \in V$

FIG. 1. LP-formulation for EMD on hypergraph $H = (V, E, w)$ corresponding to distributions $\mathbb{P}_{\text{left}}$ and $\mathbb{P}_{\text{right}}$. Comments are enclosed by (* and *).

among the nodes in $\mathcal{T}\text{ail}_{e'}$ and add these values to the probabilities of these nodes. In other words, for every node $y \in \mathcal{T}\text{ail}_{e'}$ we add $(|\mathcal{T}\text{ail}_e| \times \deg_x^{in} \times |\mathcal{T}\text{ail}_{e'}|)^{-1}$ to $\mathbb{P}_{\text{left}}(y)$.

Note that the final probability for each node is calculated by summing all the contributions from each bullet point. In closed form, $\mathbb{P}^x_{\text{left}}(y)$ is given by:

$$\mathbb{P}^x_{\text{left}}(y) = \frac{\delta_{\deg_y^{in},0} \times \delta_{|\mathcal{T}\text{ail}_e|-1,|\mathcal{T}\text{ail}_e \setminus \{y\}|}}{|\mathcal{T}\text{ail}_e|} + \sum_{\substack{x \in \mathcal{T}\text{ail}_e \\ x \in \mathcal{H}\text{ead}_{e'} \\ y \in \mathcal{T}\text{ail}_{e'}}} \frac{1 - \delta_{\deg_x^{in},0}}{|\mathcal{T}\text{ail}_e| \times \deg_x^{in} \times |\mathcal{T}\text{ail}_{e'}|}$$

where $\delta(i,j)$ is the Kronecker delta function, *i.e.*,

$$\delta_{i,j} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases}$$

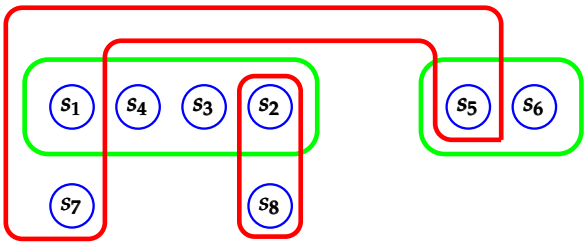
$\mathbb{P}_{\text{right}}$ is determined in a symmetric manner. The details are provided in the appendix for the sake of completeness.

2. *Calculations of $\mathbb{P}_{\text{left}}$ and $\mathbb{P}_{\text{right}}$ for an Undirected Hypergraph*

For this case, $\mathbb{P}^x_{\text{left}}(y) = \mathbb{P}^x_{\text{right}}(y)$ for all $x, y \in \mathcal{A}_e$ since the hypergraph is undirected. Let $0 < \alpha < 1$ be a parameter that encodes the "laziness" of the random walk. Then, $\mathbb{P}^x_{\text{left}}$ is determined in the following manner by adopting the calculations in [30] (see Fig. 2 for an illustration):

- ▷ Initially, $\mathbb{P}^x_{\text{left}}(x) = \alpha$ and $\mathbb{P}^x_{\text{left}}(y) = 0$ for all $y \neq x$. In our subsequent steps, we will add to these values as appropriate.
- ▷ We divide and allocate the remaining total probability $1 - \alpha$ among all the hyperedges that contain x proportionally to the cardinality of these hyperedges *excluding* the node x, *i.e.*, a hyperedge e' such that $x \in \mathcal{A}_e$ gets $\delta_{e'} \stackrel{\text{def}}{=} \frac{|\mathcal{A}_{e'}|-1}{\sum_{x \in \mathcal{A}_{e''}} (|\mathcal{A}_{e''}|-1)}$. Then, we divide the allocated value $\delta_{e'}$ equally among the nodes in $\mathcal{A}_{e'}$ *excluding* x and add these values to the probabilities of these nodes. In other words, for every node $y \in \mathcal{A}_{e'}$ such that $y \neq x$ we add $\frac{\delta_{e'}}{|\mathcal{A}_{e'}|-1}$ to $\mathbb{P}^x_{\text{left}}(y)$.

FIG. 2. An illustration of the calculations of $\mathbb{P}_{\text{left}}$ and $\mathbb{P}_{\text{right}}$ for an undirected hypergraph, as outlined in Section II A 2, where the node set is $\{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$ and the four hyperedges are $\{s_1, s_2, s_3, s_4\}$, $\{s_1, s_5, s_7\}$, $\{s_5, s_6\}$ and $\{s_2, s_8\}$.



$$\mathbb{P}^{s_1}_{\text{left}}(s_1) = \mathbb{P}^{s_1}_{\text{right}}(s_1) = \alpha$$
$$\mathbb{P}^{s_1}_{\text{left}}(s_2) = \mathbb{P}^{s_1}_{\text{right}}(s_2) = (1-\alpha) \times \tfrac{3}{2+3} \times \tfrac{1}{3}$$
$$\mathbb{P}^{s_1}_{\text{left}}(s_3) = \mathbb{P}^{s_1}_{\text{right}}(s_3) = (1-\alpha) \times \tfrac{3}{2+3} \times \tfrac{1}{3}$$
$$\mathbb{P}^{s_1}_{\text{left}}(s_4) = \mathbb{P}^{s_1}_{\text{right}}(s_4) = (1-\alpha) \times \tfrac{3}{2+3} \times \tfrac{1}{3}$$
$$\mathbb{P}^{s_1}_{\text{left}}(s_5) = \mathbb{P}^{s_1}_{\text{right}}(s_5) = (1-\alpha) \times \tfrac{2}{2+3} \times \tfrac{1}{2}$$
$$\mathbb{P}^{s_1}_{\text{left}}(s_6) = \mathbb{P}^{s_1}_{\text{right}}(s_6) = 0$$
$$\mathbb{P}^{s_1}_{\text{left}}(s_7) = \mathbb{P}^{s_1}_{\text{right}}(s_7) = (1-\alpha) \times \tfrac{2}{2+3} \times \tfrac{1}{2}$$
$$\mathbb{P}^{s_1}_{\text{left}}(s_8) = \mathbb{P}^{s_1}_{\text{right}}(s_8) = 0$$

In closed form, $\mathbb{P}^x_{\text{left}}(y)$ is given by:

$\mathbb{P}^x_{\text{left}}(y) =$

$$\begin{cases} \alpha, & \text{if } x = y \\ (1-\alpha) \times \sum\limits_{x,y \in e'} \frac{|\mathcal{A}_{e'}|-1}{\sum_{x \in \mathcal{A}_{e''}}(|\mathcal{A}_{e''}|-1)} \times \frac{1}{|\mathcal{A}_{e'}|-1} & \\ \qquad = (1-\alpha) \times \dfrac{\left|\{e' \,|\, x, y \in \mathcal{A}_{e'}\}\right|}{\sum_{x \in \mathcal{A}_{e''}}(|\mathcal{A}_{e''}|-1)} & , \text{ otherwise} \end{cases}$$

The parameter α was used in the context of Ricci flows on undirected graphs by Ni *et al.* [35], and by Lai *et al.* [37]. These works suggested using a non-zero value for α, but the exact choice of α was left to the specific application in question. Thus, we decided to use a non-zero value of α and found that a value of $\alpha = 0.1$ works best in our applications.

B. Ricci Flow, Weight-renormalization, Topological Surgery and Flow Convergence for Weighted Hypergraphs

Before proceeding with the technical descriptions, we first provide a brief informal intuition behind the proposed approach. The curvature values of the hyperedges provide a (positive or negative) value to each hyperedge of the hypergraph. The Ricci flow is an iterative process that produces a sequence of hypergraphs, where each iteration of the Ricci flow process dynamically alters the weights of hyperedges based on their current weights and curvature values. On average, these weight alterations tend to increase the weights of the hyperedges connecting the core(s) to the rest of the hypergraph while decreasing the weights of those hyperedges within the core(s). The weight re-normalization procedure after every iteration ensures that the weights of the hyperedges do not increase in an unbounded manner and instead eventually converge to some "steady-state" values. The goal of the topological surgery procedure performed once in a few iterations is to remove the hyperedges connecting the core(s) to the rest of the hypergraph so that at the end of our Ricci flow process, when the hyperedge weights have converged to some stable values, we can recover the core(s) from the connected components of the hypergraph. This process of topological surgeries and hyperedge weight updates via Ricci flows is somewhat analogous to the Newman-Girvan's algorithm [71] which, in the context of undirected graphs, iteratively removes edges of high betweenness centrality and recomputes the edge betweenness centrality.

We now present the precise technical descriptions of these concepts. Let $t = 0, 1, 2, \cdots$ is the discrete iteration index, and let $H^{(0)} = (V^{(0)}, E^{(0)}, w^{(0)}), H^{(1)} = (V^{(1)}, E^{(1)}, w^{(1)}), H^{(2)} = (V^{(2)}, E^{(2)}, w^{(2)}), \cdots$ denote the sequence of hypergraphs produced by the Ricci flow with $H^{(0)}$ being equal to the original (starting) hypergraph $H = (V, E, w)$. The Ricci flow equation (*without* topological surgeries and hyperedge-weight renormalization) is as follows [35–37]:

$$\boxed{w^{(t+1)}(e) = w^{(t)}(e) - w^{(t)}(e) \times \textsc{Ric}^{(t)}(e)} \tag{3}$$

where $\textsc{Ric}^{(t)}(e)$ is the curvature value based on the edge-weights $W^{(t)} = \left\{w^{(t)}(e) \,|\, e \in E^{(t)}\right\}$. Note that if $w^{(t)}(e) = 0$ for some value $t = t_0$ then $w^{(t)}(e)$ stays zero for all $t > t_0$ based on (3), so we may simply remove such edges e from all $E^{(t)}$ with $t \geq t_0$. Also note that $w^{(t)}(e) \geq 0$ for all t since $\textsc{Ric}^{(t)}(e) \leq 1$ for all t.

Unfortunately, as observed in papers such as [37], there *are* problematic aspects to the Ricci flow equation in (3). In particular, in applications such as in this article, we would like the iterations to eventually *converge* (within a reasonable time), but it can be easily seen that there exists hypergraphs for which the hyperedge weights may keep on increasing in successive iterations. As a remedy for graphs only, Lai, Bai and Lin [37] proposed changing (3) to (4) as shown below such that the Ricci flow is "normalized" in the sense that the sum of edge weights of the graph remain the same and therefore edge weights *cannot* become arbitrarily large:

$$\boxed{\begin{aligned} w^{(t+1)}(e) = w^{(t)}(e) &- w^{(t)}(e) \times \textsc{Ric}^{(t)}(e) \\ &+ \frac{s \times w^{(t)}(e) \times \left(\sum_{h \in E}\left(w^{(t)}(h) \times \textsc{Ric}^{(t)}(h)\right)\right)}{\sum_{h \in E}\left(w^{(0)}(h) \times \textsc{Ric}^{(0)}(h)\right)} \end{aligned}} \tag{4}$$

In the above equation, $s > 0$ is a *constant* (called "step size" in [37]). Unfortunately, as we will prove in Theorem in Section III C, there are infinitely many graphs for which $w^{(1)}(e)$ will become negative thus rendering the iterative process in (4) *impossible* to execute beyond the first step. Instead, in our algorithm we perform hyperedge-weight re-normalization by applying a sigmoidal function to hyperedge weights to ensure that *all* hyperedge weights do *not* exceed 1, *i.e.*, for $t \geq 1$ we replace $w^{(t)}(e)$ by $\frac{1}{1+\mathbf{e}^{-w^{(t)}(e)}}$. **In the sequel, unless explicitly mentioned otherwise, when we refer to weight $\boldsymbol{w^{(t)}(e)}$ for $\boldsymbol{t \geq 1}$ we refer to the weight after re-normalization.**

Unfortunately, there is *no* closed-form analytical solution of Equation (3) yet and it is *not* even clear if such a solution is possible. Here we introduce our topological surgery operation, and provide an *informal* intuition behind our approach of finding cores by using Ricci flows with topological surgery. Note that since all hyperedge weights are non-negative at *all* times, Equation (3) shows that $w^{(t+1)}(e) < w^{(t)}(e)$ if $\textsc{Ric}^{(t)}(e) > 0$ and $w^{(t+1)}(e) > w^{(t)}(e)$ if $\textsc{Ric}^{(t)}(e) < 0$, *i.e.*, each iteration of the Ricci flow process dynamically alters the weights of hyperedges based on their curvature values, increasing the weights of those with negative curvature while decreasing the weights of those with positive curvature. Using the observation in [36] that states (quoted verbatim) "positively curved edges are well connected in the sense that none of them are essential for the proper

transport operation", which is also supported by research works such as [35, 36] on graphs, it follows that this effect should on an average lead to pairs of nodes within a core being connected by hyperedges with a smaller weight whereas pairs of nodes outside cores being connected by hyperedges with a larger weight. Consequently, we can use the following "topological surgery" method to isolate the core(s) from the remaining parts of the hypergraph: *remove hyperedges with substantial weights following every several iterations of the Ricci flow.* This strategic manipulation enhances the clarity of core structures within the network. Moreover, since each hyperedge in a hypergraph typically involves many nodes, surgical removal of hyperedges may disconnect more nodes (as compared to graphs in which each edge always involves two nodes), thus leading to the survival of a few very well-connected sets of nodes as cores. See Fig. 3 for a visual illustration of some of these intuitions.

For our experiments, we did surgery every 2 iterations and ran our algorithm for a total of 40 iterations in total for every hypergraph. The amount of hyperedges to be removed is an adjustable parameter that can be tweaked accordingly to get cores of different sizes. For our experiments, we set our "surgery amount" to be those hyperedges that have weights in the largest 8% of the weights of all hyperedges in the previous iteration. These combinations of adjustable parameters provided us with a reasonable combination of rapid rate of convergence and acceptable core quality parameters.

To check if the edge-weights have converged to a fixed-point we use a standard convergence criterion in which the *average* of the *absolute* differences of edge-weights in successive iterations is sufficiently small, *i.e.*,

$$\Delta_{\text{AVE}} = \frac{1}{|E^{(t)}|} \times \sum_{e \in E^{(t)}} |w^{(t+1)}(e) - w^{(t)}(e)| \tag{5}$$

is at most ε for some small real number $\varepsilon \geq 0$. We use $\varepsilon = 0.005$ for directed hypergraph applications and $\varepsilon = 0.000005$ for undirected hypergraph applications. To check the dispersion of these absolute differences around mean we calculate the standard deviation, *i.e.*,

$$\Delta_{\text{STD}} = \sqrt{\frac{1}{|E^{(t)}|} \times \sum_{e \in E^{(t)}} \left(\Delta_{AVE} - |w^{(t+1)}(e) - w^{(t)}(e)|\right)^2} \tag{6}$$

C. Quality Measures of Influential cores

A *core* is a subset of nodes that are more connected to each other as opposed to the rest of the hypergraph. In this article, following the approach in [64] we also want our cores for hypergraphs to be *central* and *influential* in the sense that removal of the nodes in the core *significantly* disrupts short paths between nodes *not* in the core. This leads to the following quality constraints and parameters for a core that generalizes similar conventions used by the network science community for graphs.

1. Connectivity Constraint

For an undirected hypergraph (respectively, directed hypergraph) a core is a connected component (respectively, a weakly connected component) of the hypergraph. This is a bare minimum constraint that a core should satisfy.

2. Size Constraint

The size (number of nodes) of a core should be *non-trivial*, *i.e.*, neither too small nor too large. For example, a core containing more than 50% of all nodes or containing only 5 nodes is *hardly* an interesting core. For the real hypergraphs investigated in this paper, our algorithm *always* produces only one or two cores of non-trivial size (*cf.* Table VI and Table VII) in the sense that each of the remaining connected components has *very* few nodes.

3. Cohesiveness Measure

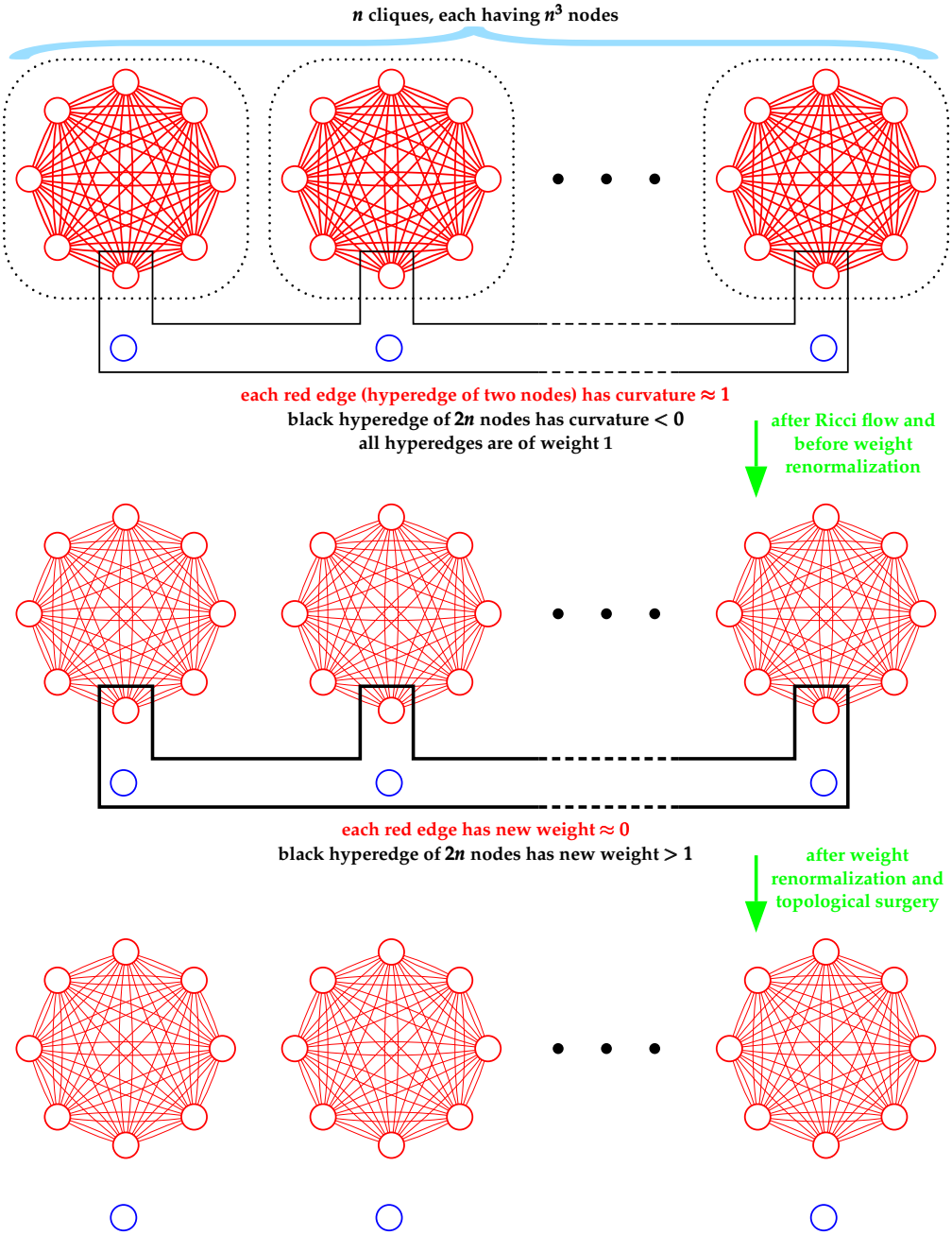
This goal of quantifying this measure is to ensure that the nodes in the core should be connected more among themselves as opposed to nodes outside the core. For our hypergraphs, we quantify this in the following manner.

Undirected hypergraph

For an undirected hypergraph $H = (V, E, w)$, a non-empty proper subset V' of V and a node $x \in V$, let the notation $H \setminus V'$ denote the hypergraph $(V \setminus V', E', w)$ obtained by removing the nodes in V' from V and removing any hyperedge e with $\mathscr{A}_e \cap V' \neq \emptyset$ from E. Let the notation $\mathsf{deg}_x(H)$ denote the degree of x in H. For undirected graphs, several measures of cohesiveness have been used in prior published literatures [72], *e.g.*, via *distance*, via *degree*, via *density*, *etc.* However, most prior published articles dealing with finding cores or core-like structures for undirected hypergraphs use a simple "degree" cohesive measure by extending the concept of a k-core (or its minor variations) from undirected graphs to undirected hypergraphs [51–61]. In the notations and terminologies of this article, a k-core of an undirected hypergraph $H = (V, E, w)$ is a subset of nodes $\mathcal{S}$ such that $\mathsf{deg}_x(H \setminus (V \setminus \mathcal{S})) \geq k$ for every node $x \in \mathcal{S}$; larger values of k signify a better core. In particular, a 1-core trivially exists in any hypergraph and therefore is not considered a core at all.

We however believe that the above-mentioned k-core measure is not very suitable for the type of undirected

FIG. 3. A visual illustration of the intuition behind our approach of finding cores by using Ricci flows with topological surgery as discussed in Section II B. The notation "≈ 0" refers to a function $f(n)$ such that $\lim_{n\to\infty} f(n) = 0$. The nodes are colored blue and red for visual clarity: red nodes are involved in cliques of hyperedges of two nodes (i.e., cliques of edges) and all the blue nodes together with an equal number of red nodes appear in a single hyperedge of $2n$ nodes. The cliques are enclosed by dotted black bounding boxes for visual clarity (the cliques do not correspond to hyperedges). The second figure from top indicates the hypergraph after one iteration of Ricci flow but before the weight renormalization. The thicknesses of the red edges are reduced to indicate the decrease of their weights from approximately 1 to approximately 0 and the thickness of the black hyperedge is increased to indicate an increase of its weight. The third figure from top shows that the black hyperedge of $2n$ nodes gets deleted as a result of weight renormalization and topological surgery, thus giving us the n cores corresponding to the n cliques.



hypergraphs studied in this article (namely, co-author hypergraphs (see Section III B 2) and similar other social interaction hypergraphs). The condition of a k-core is too strict and changes the quality of the core too abruptly. For example, if a node x in the core $\mathcal{S}$ satisfies $\mathsf{deg}_x(H \setminus (V \setminus \mathcal{S})) = k - 10$ then $\mathcal{S}$ defines a $(k-10)$-core even if every remaining node y in $\mathcal{S}$ satisfies $\mathsf{deg}_y(H \setminus (V \setminus \mathcal{S})) = k$. Density-based measures centered on average degrees have been used extensively in the computer science literature for graphs [73–82], and in one instance for undirected hypergraphs [83]. Following these research works, we consider a measure of cohesion based on average degrees. Note that for the example mentioned above the average degree changes from $\frac{k}{|\mathcal{S}|}$ to $\frac{k\times(|\mathcal{S}|-1)+1\times(k-10)}{|\mathcal{S}|} = k - \frac{10}{|\mathcal{S}|}$, showing a more gradual deterioration of the quality with an increasing value of $|\mathcal{S}|$. However, one should also account for the nodes outside of $\mathcal{S}$. For example, for the node x it is possible that either **(*a*)** $\mathsf{deg}_x(H \setminus (V \setminus \mathcal{S})) = \mathsf{deg}_x(H) = k - 10$, or **(*b*)** $\mathsf{deg}_x(H \setminus (V \setminus \mathcal{S})) = k - 10$ but $\mathsf{deg}_x(H) = k - \alpha$ for some $\alpha < 10$. Our cohesiveness measure should indicate better cohesiveness for case **(*a*)** as opposed to case **(*b*)**, and for case **(*b*)** our cohesiveness measure should indicate worse cohesiveness with increasing α. Thus, we use the following measure for cohesiveness of a core $\mathcal{S}$:

$$\varkappa^{deg} = \frac{\sum_{x\in\mathcal{S}} \frac{\mathsf{deg}_x(H\setminus(V\setminus\mathcal{S}))}{\mathsf{deg}_x(H)}}{|\mathcal{S}|} \tag{7}$$

In the above equation, $H \setminus (V \setminus \mathcal{S})$ is the sub-hypergraph of H induced by the nodes in $\mathcal{S}$, $\mathsf{deg}_x(H \setminus (V \setminus \mathcal{S}))$ is the degree of node x in this induced sub-hypergraph, the ratio $\mathsf{deg}_x(H \setminus (V \setminus \mathcal{S}))/\mathsf{deg}_x(H) \in [0, 1]$ provides a measure of how much the node x is connected to only nodes in $\mathcal{S}$ when we exclude all connections to nodes outside of $\mathcal{S}$ via hyperedges, and the entire equation averages out the ratio over nodes in $\mathcal{S}$. Simple calculations show that for case **(*b*)** of our example $\varkappa^{deg} = \frac{(k-1)\times 1+1\times\frac{k-10}{k-\alpha}}{k} = 1 - \frac{10-\alpha}{k\times(k-\alpha)}$, which decreases with increasing α, as desired. Note that obviously $0 \leq \varkappa^{deg} \leq 1$. Any value of $\varkappa^{deg}$ in the range $(1/2, 1]$ with a statistical significance indicator p-value (see Section II C 5) below 10^{-5} indicates a valid core since in that case the nodes in the core on average are connected more to other nodes in the core as opposed to nodes outside the core and this property is not satisfied by the null hypothesis model; in other words, a core found by any algorithm will be considered to be invalid if either $\varkappa^{deg} \leq 0.5$ or if the p-value is greater than or equal to 10^{-5}. Higher values of $\varkappa^{deg}$ indicate better cohesiveness of the core.

Directed hypergraph

For a directed hypergraph $H = (V, E, w)$, a non-empty proper subset V' of V and a node $x \in V$, let the notation $H \setminus V'$ denote the hypergraph $(V \setminus V', E', w)$ obtained by removing the nodes in V' from V and removing any hyperedge e with $(\mathcal{T}\mathrm{ail}_e \cup \mathcal{H}\mathrm{ead}_e) \cap V' \neq \emptyset$ from E, and the notations $\mathsf{deg}^{in}_x(H), \mathsf{deg}^{in}_x(H \setminus V'), \mathsf{deg}^{in}_x(out)$ and $\mathsf{deg}^{out}_x(H \setminus V')$ denote the corresponding in-degrees and out-degrees of x in H and $H \setminus V'$. As we mentioned already, we could not find existing peer-reviewed published materials for finding cores in directed weighted or unweighted hypergraphs, and thus *no* prior cohesive measures for directed hypergraphs were available. Our cohesiveness quality measures for directed hypergraphs are a direct generalization the cohesiveness quality measure for undirected hypergraphs as stated in Equation (7). Due to the directionality of a hyperedge in a directed hypergraph, we get *two* cohesive parameters. For a directed hypergraph $H = (V, E, w)$ our cohesiveness measures for a core $\mathcal{S}$ are the following two values:

$$\varkappa^{deg}_{\text{in}} = \frac{\sum_{x\in\mathcal{S}} \frac{\mathsf{deg}^{in}_x(H\setminus(V\setminus\mathcal{S}))}{\mathsf{deg}^{in}_x(H)}}{|\mathcal{S}|} \tag{8}$$

$$\varkappa^{deg}_{\text{out}} = \frac{\sum_{x\in\mathcal{S}} \frac{\mathsf{deg}^{out}_x(H\setminus(V\setminus\mathcal{S}))}{\mathsf{deg}^{out}_x(H)}}{|\mathcal{S}|} \tag{9}$$

In the above two equations, $H \setminus (V \setminus \mathcal{S})$ is the directed sub-hypergraph of H induced by the nodes in $\mathcal{S}$, $\mathsf{deg}^{in}_x(H \setminus (V \setminus \mathcal{S}))$ (respectively, $\mathsf{deg}^{out}_x(H \setminus (V \setminus \mathcal{S}))$) is the in-degree (respectively, out-degree) of node x in this induced directed sub-hypergraph, the ratio $\mathsf{deg}^{in}_x(H \setminus (V \setminus \mathcal{S}))/\mathsf{deg}^{in}_x(H) \in [0, 1]$ (respectively, the ratio $\mathsf{deg}^{out}_x(H \setminus (V \setminus \mathcal{S}))/\mathsf{deg}^{in}_x(H) \in [0, 1]$) provides a measure of how much the node x is connected only to nodes in $\mathcal{S}$ (respectively, nodes in $\mathcal{S}$ are connected to the node x) when we exclude all connections to nodes outside of $\mathcal{S}$ via directed hyperedges, and the entire equation averages out the ratio over nodes in $\mathcal{S}$. For example, $\frac{\mathsf{deg}^{in}_x(H\setminus(V\setminus\mathcal{S}))}{\mathsf{deg}^{in}_x(H)} = 0.7$ indicates that 70% of the the in-degree of node x is contributed by hyperedges that contain only nodes from $\mathcal{S}$ both in their head and tail and only 30% of the hyperedges contributing to the in-degree of x have one or more outside nodes either in their head or in their tail. Note that obviously $0 \leq \varkappa^{deg}_{\text{in}}, \varkappa^{deg}_{\text{out}} \leq 1$. Again for a similar reason as in the undirected case, values of *both* $\varkappa^{deg}_{\text{in}}$ and $\varkappa^{deg}_{\text{out}}$ in the range $(1/2, 1]$ with a statistical significance indicator p-value (see Section II C 5) below 10^{-5} indicate a valid core; in other words, a core found by any algorithm will be considered to be invalid if the values of at least one of $\varkappa^{deg}_{\text{in}}$ or $\varkappa^{deg}_{\text{out}}$ is at most 0.5 or if at least one of their p-values is greater than or equal to 10^{-5}. Higher values of $\varkappa^{deg}_{\text{in}}$ and $\varkappa^{deg}_{\text{out}}$ indicate better cohesiveness of the core.

4. Centrality Measure

These types of measures are a "combinatorial analog" of the *information loss* quantifications used in prior research works such as [45], and are significant in real-world applications to biological and social networks as mentioned in prior publications such as [64] in the context of undirected unweighted graphs. For our hypergraphs, these measures quantify centrality and influential nature of the core in the sense that removal of the nodes in the core significantly disrupts short paths between nodes not in the core. In the definitions below we use the notation $H \setminus V'$ as defined in Section II C 3.

Directed hypergraph

Let $H = (V, E, w)$ be the directed hypergraph and $\emptyset \subset \mathcal{S} \subset V$ be the core we are evaluating. Our *first* goal is to measure the fraction of ordered pairs of nodes for which there *was* a path in the given input hypergraph but there *no longer* is a path after removing the core. Let ζ denote the number of ordered pairs (u, v) of nodes u, v not in *any* core for which there was a (directed) path from u to v in H but no (directed) path in $H \setminus \mathcal{S}$. We then calculate the following quantity:

$$\boxed{\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}} = \frac{\zeta}{|V \setminus \mathcal{S}| \times (|V \setminus \mathcal{S}| - 1)}} \tag{10}$$

In addition, our *second* goal is to measure the *average percentage* increase in the length of paths among ordered pairs of nodes that remain connected both before and after removing the core. This is done as follows. Let ξ be the number of ordered pair (u, v) of nodes u, v not in *any* core for which there was a (directed) path from u to v in both H and $H \setminus \mathcal{S}$. We then calculate the following quantity:

$$\boxed{\varkappa^{directed}_{\text{dist\_stretch}} = \frac{1}{\xi} \times \sum_{\substack{(u,v) \in V \setminus \mathcal{S}: \\ u \neq v \\ \text{dist}_{H \setminus \mathcal{S}}(u,v) < \infty \\ \text{dist}_{H}(u,v) < \infty}} \frac{\text{dist}_{H \setminus \mathcal{S}}(u, v)}{\text{dist}_{H}(u, v)}} \tag{11}$$

Note that every ordered pair of nodes from $V \setminus \mathcal{S}$ appears in the calculation of either $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$ or $\varkappa^{directed}_{\text{dist\_stretch}}$ but *not* both, the reason being that incorporating the ordered pair of nodes used in (10) in (11) instead would have make the value of $\varkappa^{directed}_{\text{dist\_stretch}}$ become ∞. Note that $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$ is at most 1, and $\varkappa^{directed}_{\text{dist\_stretch}}$ is at least 1 (since edge removal does not decrease the distance values).

The values of $\varkappa^{directed}_{\text{dist\_stretch}}$ and $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$ are considered to be valid only if their statistical significance indicator p-values (see Section II C 5) are below 10^{-5}, and higher values of both $\varkappa^{directed}_{\text{dist\_stretch}}$ and $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$ indicate stronger central and influential quality. Note that a small value of $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$ does not signify a weak-quality core as long as $\varkappa^{directed}_{\text{dist\_stretch}}$ is sufficiently large; however if $\varkappa^{directed}_{\text{dist\_stretch}}$ is not sufficiently large then $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$ must be sufficiently large to signify the centrality of the core. In this article, we adopt the strict criterion that for a valid core either $\varkappa^{directed}_{\text{dist\_stretch}}$ must be at least $3/2$ (*i.e.*, shortest paths are stretched by at least 50%), or if $\varkappa^{directed}_{\text{dist\_stretch}}$ is below $3/2$ then $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$ must be at least $1/2$ (*i.e.*, at least 50% of the ordered pairs of nodes are disconnected); in other words, a core found by any algorithm will be considered to be invalid if both $\varkappa^{directed}_{\text{dist\_stretch}} < 3/2$ and $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}} < 1/2$.

Undirected hypergraph

Let $H = (V, E, w)$ be the undirected hypergraph and $\emptyset \subset \mathcal{S} \subset V$ be the core we are evaluating. Our *first* goal is to measure the fraction of pairs of nodes for which there *was* a path in the given input hypergraph but there *no longer* is a path after removing the core. Let ζ denote the number of unordered pairs $\{u, v\}$ of nodes u, v not in *any* core for which there is no path between them in $H \setminus \mathcal{S}$. We then calculate the following quantity:

$$\boxed{\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}} = \frac{\zeta}{\binom{|V \setminus \mathcal{S}|}{2}}} \tag{12}$$

In addition, our *second* goal is to measure the *average percentage* increase in the length of paths among unordered pairs of nodes that remain connected both even after removing the core. This is done as follows. Let ξ be the number of unordered pair $\{u, v\}$ of nodes u, v not in *any* core for which there was still a path between u to v in $H \setminus \mathcal{S}$. We then calculate the following quantity:

$$\boxed{\varkappa^{undirected}_{\text{dist\_stretch}} = \frac{1}{\xi} \times \sum_{\substack{\{u,v\} \in V \setminus \mathcal{S}: \\ u \neq v \\ \text{dist}_{H \setminus \mathcal{S}}(u,v) < \infty}} \frac{\text{dist}_{H \setminus \mathcal{S}}(u, v)}{\text{dist}_{H}(u, v)}} \tag{13}$$

Note that every (unordered) pair of nodes from $V \setminus \mathcal{S}$ appears in the calculation of either $\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}}$ or $\varkappa^{undirected}_{\text{dist\_stretch}}$ but *not* both, as incorporating the pair of nodes used in (12) in (13) instead would yield $\varkappa^{undirected}_{\text{dist\_stretch}} = \infty$ Note that $\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}}$ is at most 1, and $\varkappa^{undirected}_{\text{dist\_stretch}}$ is at least 1 (since edge removal does not decrease the distance values). The values of $\varkappa^{undirected}_{\text{dist\_stretch}}$ and $\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}}$ are considered to be valid only if their statistical significance indicator p-values (see Section II C 5) are below 10^{-5}, and higher values of both $\varkappa^{undirected}_{\text{dist\_stretch}}$ and $\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}}$ indicate stronger central and influential quality. Note that a small value of $\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}}$ does not signify a weak-quality core as long as $\varkappa^{undirected}_{\text{dist\_stretch}}$ is sufficiently large; however if $\varkappa^{undirected}_{\text{dist\_stretch}}$ is not sufficiently large then $\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}}$ must be sufficiently large to justify centrality of the core. In this article, we adopt the strict criterion that

for a valid core either $\varkappa^{undirected}_{\text{dist\_stretch}}$ must be at least $3/2$ (*i.e.*, shortest paths are stretched by at least 50%), or if $\varkappa^{undirected}_{\text{dist\_stretch}}$ is below $3/2$ then $\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}}$ must be at least $1/2$ (*i.e.*, at least 50% of pairs of nodes are disconnected); in other words, a core found by any algorithm will be considered to be invalid if both $\varkappa^{undirected}_{\text{dist\_stretch}} < 3/2$ and $\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}} < 1/2$.

5. *Statistical Significance Measure: Calculations of p-values for Core Quality Parameters*

Statistical significance (p-value) calculations for core quality parameters require a null hypothesis model corresponding a random hypergraph similar in some essential characteristics to the one studied. We explain below why two most common methods for generating random graphs used by the network science community for p-value calculations *fail* to generalize to hypergraphs:

Generative models: The random graphs are generated so that they statistically match some key topological characteristics of the given graph such as node degree distributions for undirected graphs and distribution of in-degrees and out-degrees of nodes for directed graphs. There are *two* reasons that prevented us from using these methods for our hypergraphs. First, there are *no* broadly accepted evidences of topological characteristics such as degree distributions for metabolic and co-authorship hypergraphs. Secondly, it is *not* clear how we will generate random hypergraphs so that they statistically match key topological characteristics of the given hypergraph, *e.g.*, the methods outlined in [46, 47, 71, 84, 85] for generating random graphs with prescribed degree-distributions are *not* easily generalizable to hypergraphs.

Random-swap models: For graphs, these kind of random graphs are generated using a Markov-chain algorithm [86] by starting with the real graph and repeatedly swapping randomly chosen "compatible" pairs of edges. However, it is not very clear if there is an useful generalization of this hypergraphs. For graphs, an edge contributes exactly 1 to the degrees of nodes at its endpoints, leading to many compatible edges as candidates for swap and thus providing statistical validity of the model. In contrast, hyperedges may contribute to the degree of an *arbitrary* number of nodes in a more complicated fashion.

Based on the above observations, we design the following method to generate the p-values. Let $H = (V, E, w)$ be the (directed or undirected) hypergraph, let $\mathcal{S} \subset V$ be the core in question with $\alpha_1, \ldots, \alpha_r$ being the values of its quality parameters (for directed hypergraphs $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ are the values of $\varkappa^{deg}_{\text{in}}$, $\varkappa^{deg}_{\text{out}}$, $\varkappa^{directed}_{\text{dist\_stretch}}$ and $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$, respectively; for undirected hypergraphs $\alpha_1, \alpha_2, \alpha_3$ are the values of $\varkappa^{deg}$, $\varkappa^{undirected}_{\text{dist\_stretch}}$ and $\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}}$, respectively). We generate 100 *random* subsets of V, say $\mathcal{B}_1, \ldots, \mathcal{B}_{100}$, such that $|\mathcal{B}_1| = \cdots = |\mathcal{B}_{100}| = |\mathcal{S}|$ and compute the values of $\beta_{i,j}$ for $i \in \{1, \ldots, 100\}$ and $j \in \{1, \ldots, r\}$, where $\beta_{i,j}$ is the value of the j^{th} property for $\mathcal{B}_i$. We calculate the p-value for the j^{th} property by performing a *one-sample* $\mathtt{t}$-test with $\beta_{1,j}, \ldots, \beta_{100,j}$ as the values of the samples and α_j as the value of the hypothesis.

The p-value is a real number between 0 and 1; lower p-values indicate better statistical significance. Following standard practice in network science, in this article we adopt a strict constraint on the acceptable p-values: **a p-value that is more than 10^{-5} even for a single quality measure for a core will invalidate the selection of that core.**

D. Data Sources

1. *Metabolic Systems (for Directed Hypergraphs)*

We collected seven metabolic systems from BiGG Models [87], a comprehensive public repository managed by the Systems Biology Research Group at UC San Diego. These seven metabolic systems pertained to the seven species Escherichia Coli, Homo Sapiens, Helicobacter Pylori, Methanosarcina berkeri str. Fusaro, Mycobacterium Tuberculosis, Synechococcus elongatus and Synechocystis.

2. *Co-authorships data (for Undirected Hypergraphs)*

We build two undirected hypergraphs corresponding to two co-authorship datasets, which we will call the *Computer Science Papers* (CSP) dataset and the *Network Science Papers* (NSP) dataset. Each individual item in each dataset is a peer-reviewed publication in the respective (computer science or network science) research field. Our datasets are constructed following similar approaches used by prior researchers such as [88].

a. CSP *dataset* We selected three influential papers [89–91], by three *Turing Award* winner researchers S. Goldwasser, R. M. Karp and A. Wigderson working in the same general research area (*Theoretical Computer Science*). We then selected 300 of the *most* cited papers that cite each of these 3 papers giving us a list of 900 papers.

b. NSP *dataset* We selected three influential papers [71, 92, 93] in network science that have been used by previous researchers for related research works [88]. We then selected 200 of the *most* cited papers that cite each of these 3 papers giving us a list of 600 papers.

We faced a situation regarding co-authorships in network science that is usually not encountered in computer science, mathematics or theoretical physics: there are papers co-authored by a *large* number of authors.

For example, there are 355 and 141 co-authors (after including corresponding consortium authors) respectively in the following two papers: **(*a*)** *Gene expression imputation across multiple brain regions provides insights into schizophrenia risk*, Nature Genetics 51, 659–674, 2019, and **(*b*)** *Brain structural covariance networks in obsessive-compulsive disorder: a graph analysis from the ENIGMA Consortium*, Brain 143(2), 684–700, 2020. These kind of papers act as a bottleneck in the calculation of the Ricci curvature via Equation (2) that was adopted from [30] (graph-theoretic methods as illustrated in Fig. 5 will *not* be helpful either since they will include all or almost all of the 355 or 141 authors in the core).

Fortunately, there are *only* 18 of the 600 papers in our collection (3% of all the papers) that were co-authored by 15 or more authors. We removed these papers from our dataset.

III. RESULTS AND DISCUSSIONS

Before presenting our results in details in the following subsections, we provide a brief *synopsis* of them below:

- ▷ In Section III A we present our algorithm for finding core(s) in a hypergraph.
- ▷ In Section III B we show how to construct our directed and undirected hypergraphs from the corresponding datasets mentioned in Section II D.
- ▷ In Section III C we prove a theorem showing that there are infinitely many graphs for which the iterative process in (4) is *impossible* to execute beyond the first step.
- ▷ In Section III D we show that the initial convergence of Ricci flows based on the values of Δ_{AVE} occurs within a *small* number of iterations for all our hypergraphs. We then show that increasing the number of iterations further improves the values of Δ_{STD} thereby making the cores *more* stable.
- ▷ In Section III E we provide the final cores with their quality parameter values for all hypergraphs as determined by our algorithm and observe that these values *are* satisfactory.
  - ▷ In Subsection III E 1 we discuss how to *interpret* the cores and its quality parameters for the seven metabolic systems corresponding to the seven directed hypergraphs. We also briefly comment here on optimally designing the biological experimental mechanism to remove a core in subsection III E 1 a.
  - ▷ In Subsection III E 2 we discuss how to *interpret* the cores and its quality parameters for the two co-authorship data corresponding to the two undirected hypergraphs.

A. Overall Algorithmic Approach

Based on our discussions in Sections II A–II C, we design an algorithm for finding core(s) in a (directed or undirected) hypergraph whose high-level overview is presented in Table I. Below we provide brief comments on the adjustable parameters in Table I:

- ▷ η controls the *number* of iterations. Selecting larger η will make the algorithm slower but is likely to generate smaller cores, although smaller cores may *not* necessarily have better values of other quality parameters. We recommend selecting η sufficiently high to ensure that the diffusion process has actually converged in several successive iterations based on the value of Δ_{AVE} (and, optionally, Δ_{STD}) and that the quality parameters are within acceptable bounds.
- ▷ κ controls the *number* of cores selected for further analysis. We suggest a *small* value for this parameter.
- ▷ δ and τ control the *frequency* and the *amount* of the topological surgery operation, respectively. Selecting larger values of δ and smaller values of τ may help with faster convergence, but may also end up providing smaller cores by subdividing them.

Readers, especially from the algorithms or the computational complexity community, may be curious to have an expression in the theoretical worst-case running time in the big-O notation for the above algorithm. Unfortunately, it does not seem possible to derive such an expression that would provide meaningful bound to the reader since it involves too many parameters for the hypergraph. We illustrate this point for an undirected hypergraph $H = (V, E)$. Let $\textsc{Time}_{\textsc{Emd}}(p)$ denote the running time for solving the Earth Mover's distance on a hypergraph with p nodes; note that obviously $\textsc{Time}_{\textsc{Emd}}(p) = \Omega(p)$. Consider a hyperedge $e \in E$. For every pair of nodes $x, y \in \mathcal{A}_e$, the total time taken to compute $\mathbb{P}^x_{\text{left}}(y)$, $\mathbb{P}^x_{\text{right}}(y)$, $\mathbb{P}^y_{\text{left}}(x)$ and $\mathbb{P}^y_{\text{right}}(x)$ is $O\left(\sum_{e':x\in\mathcal{A}_{e'}} |\mathcal{A}_{e'}| + \sum_{e':y\in\mathcal{A}_{e'}} |\mathcal{A}_{e'}|\right)$, and the time to compute the corresponding $\textsc{Emd}_H(\mathbb{P}^p_{\text{left}}, \mathbb{P}^q_{\text{right}})$ value is $\textsc{Time}_{\textsc{Emd}}\left(\sum_{e':x\in\mathcal{A}_{e'}} |\mathcal{A}_{e'}| + \sum_{e':y\in\mathcal{A}_{e'}} |\mathcal{A}_{e'}|\right)$. Summing over all pairs of nodes in e and then summing over all hyperedges gives us the following time bound for one iteration of Ricci flow:

$$O\left(\textsc{Time}_{\textsc{Emd}}\left(\sum_{e\in E}\sum_{x,y\in\mathcal{A}_e}\left(\sum_{e':x\in\mathcal{A}_{e'}}|\mathcal{A}_{e'}| + \sum_{e':y\in\mathcal{A}_{e'}}|\mathcal{A}_{e'}|\right)\right)\right)$$

The above bound depends in a non-trivial manner on the frequencies of nodes and the lengths of hyperedges, and further simplification is not possible without making additional assumptions. For the very special case when

TABLE I. High-level overview of our algorithmic approach to find core(s) in a directed or undirected hypergraph. See Section III A for comments on the adjustable parameters. In our experiments, $\eta = 40$, $\tau = 2$, $\kappa = 2$ and $\delta = 8$.

Input:	A directed or undirected hypergraph $H = (V, E, w)$.
Adjustable parameters:	η, κ, τ, δ
Output:	A set of $\mu \leq \kappa$ mutually disjoint node subsets (cores) $\mathcal{S}_1, \ldots, \mathcal{S}_\mu \subset V$.

1. Starting with H, perform Ricci flow iterations (see Equation (3) and Section II B) for a suitable large number η of steps such that the edge-weights has converged at the end of iterations (see Equation (5) and Section II B).
   - 1.1. Perform topological surgery with "surgery amount" δ% after every τ iterations (see Section II B).
   - 1.2. Perform edge-weight normalization before the start of the next iteration (see Section II B).
2. If G is an undirected hypergraph (respectively, directed hypergraph) then output up to κ connected (respectively, weakly connected) components of G that best satisfy the quality parameters for the particular application (see Section II C).

every node occurs in *exactly* f hyperedges and every hyperedge has *exactly* the same length ℓ, letting n denote the number of nodes in the hypergraph the above running time can be simplified to $O\left(\textsc{Time}_{\textsc{Emd}}\left(\frac{nf}{\ell} \times \ell^2 \times f\right)\right) = O\left(\textsc{Time}_{\textsc{Emd}}\left(n f^2 \ell\right)\right)$. A somewhat more complicated expression for the running time for directed hypergraphs can also be calculated in a similar manner.

a. Implementation and source codes We implemented our algorithm in python. The linear program for calculation of Emd was solved using the python library of the Gurobi Optimizer whose LP solver is known for its superior performance, often solving optimization models faster than other LP solvers in the industry (we used the free academic license to use the optimizer for our work). The source codes for our implementation are freely available via GitHub at the link `https://github.com/iamprith/Ricci-Flow-on-Hypergraphs`.

B. Hypergraph Construction

1. Directed Hypergraphs

We modeled various metabolic and biochemical reactions as directed hypergraphs. Each reaction in the data had always at least one reactant but sometimes did not have a product. We represent a reaction of the form "$R_1 + \cdots + R_k \to P_1 + \cdots + P_\ell$", where R_i's are the reactants, P_j's are the products, as a directed hyperedge e (see Fig. 4) with $\mathcal{T}\text{ail}_e = \{R_1, \ldots, R_k\}$ and $\mathcal{H}\text{ead}_e = \{P_1, \ldots, P_\ell\}$[94]. In other words, each hyperedge points from the set of reactants to the set of products of the corresponding reaction. For reactions of the form "$R_1 + \cdots + R_k \to$" without a product we use $\mathcal{H}\text{ead}_e = \{\textsf{sink}\}$ for a unique node named "sink" following the *same* conventions used in the network science literature (note that there is exactly *one* sink node in the entire hypergraph and it does *not* appear in $\mathcal{T}\text{ail}_f$ for any hyperedge f in the hypergraph). *All* the directed hypergraphs that we construct are weakly connected. We computed some first-order statistics of the constructed directed hypergraphs as shown in Table II. We show in Fig. 4 a visual comparison of our hypergraph-theoretic representation with two common graph-theoretic representations of biochemical reactions.

FIG. 4. A visual illustration of the hypergraph-theoretic representation (Section III B 1) vs. two common graph-theoretic representations of biochemical reactions. If the reaction times are known accurately then they can be used as the weights of the corresponding hyperedges.

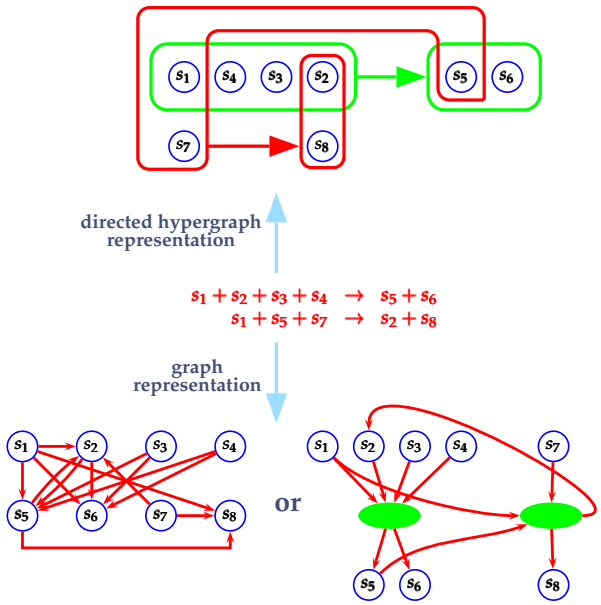
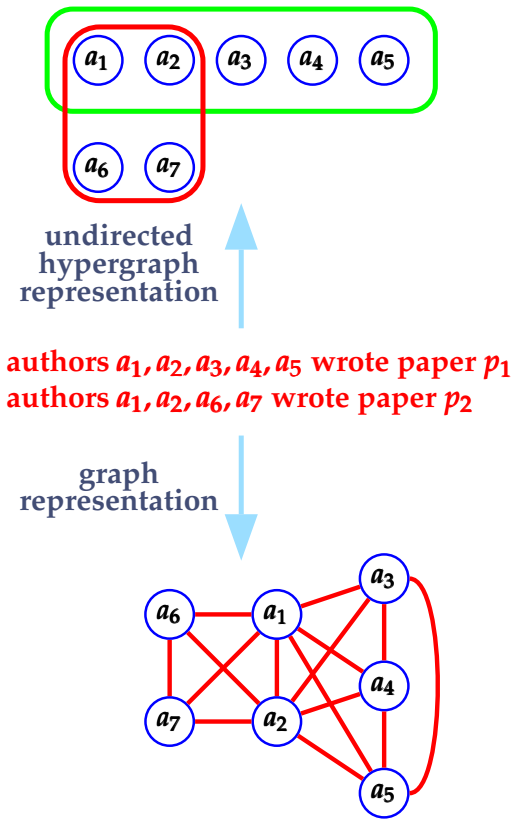


TABLE II. Some first-order statistics of constructed directed hypergraphs from metabolic systems in [87].

	Constructed directed hypergraph (V, E, w)							
			in-degree			out-degree		
Name of metabolic system	$\lvert V\rvert$	$\lvert E\rvert$	average	max	min	average	max	min
Escherichia Coli	762	1335	3.63	380	0	3.56	231	0
Homo Sapiens	343	672	3.58	165	0	3.27	114	1
Helicobacter Pylori	486	740	3.20	191	0	3.10	129	0
Methanosarcina berkeri str. Fusaro	629	905	3.22	255	0	3.09	167	0
Mycobacterium Tuberculosis	826	1297	3.68	364	0	3.49	253	0
Synechococcus elongatus	769	849	3.09	244	0	2.97	193	0
Synechocystis	796	863	3.00	277	0	3.26	222	0

FIG. 5. A visual illustration of the hypergraph-theoretic representation (Section III B 2) vs. graph-theoretic representation of co-authorships.



2. *Undirected Hypergraphs*

The hypergraph has a node corresponding to every authors and an undirected hyperedge e with $\mathcal{A}_e = \{a_1, \ldots, a_k\}$ corresponding to each paper co-authored by authors $a_1, \ldots, a_k$. For the Csp dataset, we build an undirected hypergraph out of the 900 papers and take the largest connected component as our input undirected hypergraph. For the Nsp dataset, we found the largest connected component in the resulting undirected hypergraph of 582 papers, and took that connected component as our input undirected hypergraph. We computed some first-order statistics of the constructed undirected hypergraphs as shown in Table III. We show in Fig. 5 a visual comparison of our hypergraph-theoretic representation with a common graph-theoretic representation of co-authorship relationships.

TABLE III. Some first-order statistics of constructed undirected hypergraphs in Section III B 2.

	Constructed undirected hypergraph (V, E, w)				
			degree		
	$\lvert V\rvert$	$\lvert E\rvert$	average	max	min
Csp dataset	496	609	2.97	39	1
Nsp dataset	518	213	1.61	20	1

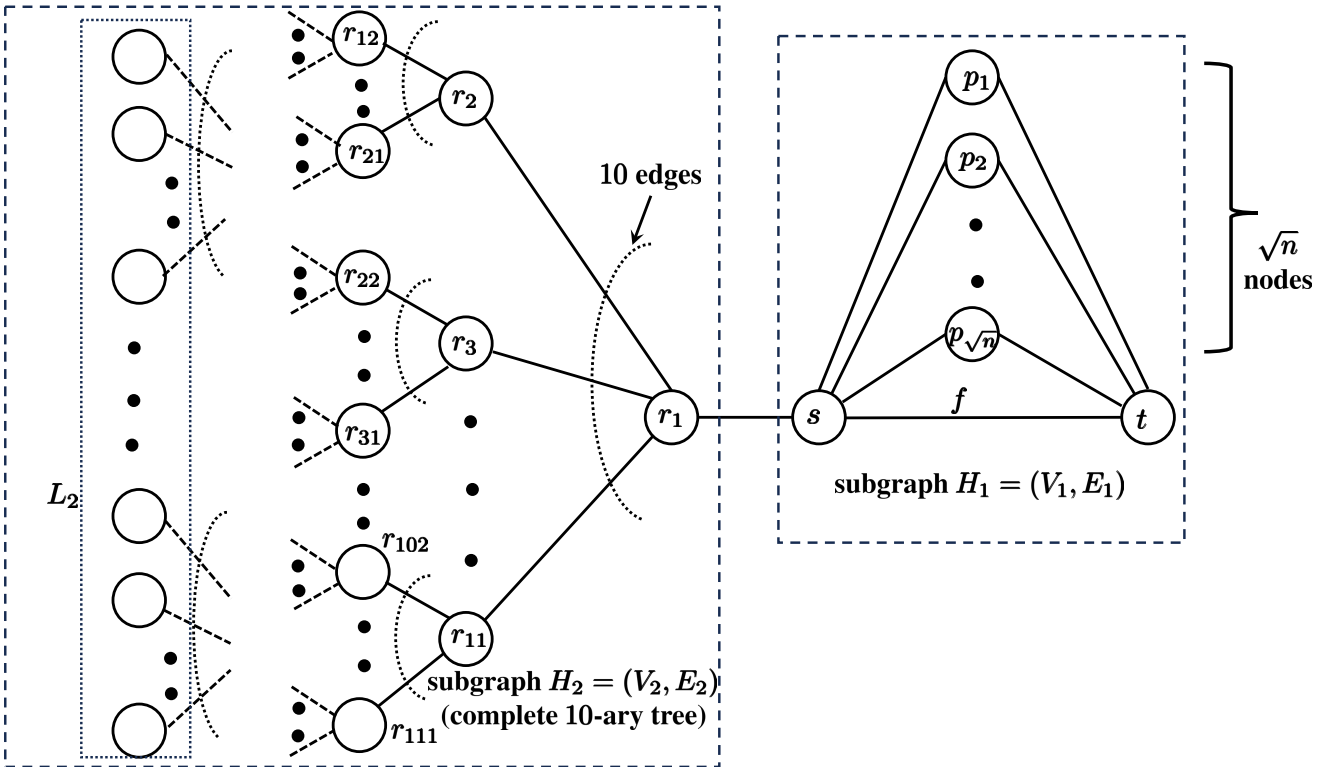
C. Proof of Inapplicability of Normalized Ricci Flow equation (4) to graphs

The following theorem shows that there are infinitely many graphs for which the normalized Ricci flow equation (4) will make $w^{(1)}(f)$ negative for some edge f thus rendering the iterative process in (4) *impossible* to execute beyond the first step.

Theorem 1. *For all sufficiently large n, there exists an undirected graph G_n on n nodes for which $w^{(1)}(f) < 0$ for some edge f of G_n.*

Proof. For convenience and ease of proof, we first explicitly state the standard definition of the Ricci curvature for an undirected graph $G = (V, E, w)$ where the weight $w(e)$ is 1 for every edge $e \in E$ [24–27, 37, 70, 95].

FIG. 6. The graph G_n used in the proof of Theorem 1.



Consider an edge $e = \{u, v\} \in E$ of our input (undirected unweighted) graph $G = (V, E)$. For a node u of G, let $\mathsf{Nbr}_G(u) = \{u\} \bigcup \{v \,|\, \{u, v\} \in E\}$ and $\mathsf{deg}_G(u) = |\,\mathsf{Nbr}_H(u) \setminus \{u\}\,|$ denote the *closed* neighborhood and the *degree* of u in G, respectively. Let $\mathbb{P}_{\mathsf{Nbr}_G(u)}$ and $\mathbb{P}_{\mathsf{Nbr}_G(v)}$ denote the two uniform distributions over the nodes in $\mathsf{Nbr}_G(u)$ and $\mathsf{Nbr}_G(v)$, respectively. Extend the distributions $\mathbb{P}_{\mathsf{Nbr}_G(u)}$ and $\mathbb{P}_{\mathsf{Nbr}_G(v)}$ to all nodes in G by assigning zero probabilities to nodes in $V \setminus \mathsf{Nbr}_G(u)$ and $V \setminus \mathsf{Nbr}_G(v)$, respectively. The Ollivier-Ricci curvature $\mathfrak{C}_G(e)$ of the edge $e = \{u, v\}$ is then defined as

$$\mathfrak{C}_G(e) = 1 - \textsc{Emd}_{\,H}(\mathbb{P}_{\mathsf{Nbr}_G(u)}, \mathbb{P}_{\mathsf{Nbr}_G(v)}) \qquad (14)$$

where we use the same calculations of EMD as in Section II A.

We will show the graph $G_n = (V, E, w)$ of m edges where $w(e) = 1$ for every edge e (and thus the sum of all edge weights is m). For this case, for $t = 0$ and an edge e of $G_n = G_n^{(0)} = (V^{(0)}, E^{(0)}, w^{(0)})$ equation (4) simplifies to

$$w^{(1)}(e) = 1 - \mathfrak{C}_{G_n}(e) + \frac{s}{m} \sum_{h \in E} \mathfrak{C}_{G_n}(h)$$

It thus suffices to show the graph G_n with an edge e satisfying $\mathfrak{C}_{G_n}(e) - \frac{s}{m} \sum_{h \in E} \mathfrak{C}_{G_n}(h) > 1$. We show this by showing a graph G_n of n nodes in which $\mathfrak{C}_{G_n}(e) \geq 1 - o(1)$ and $\frac{s}{m} \sum_{h \in E} \mathfrak{C}_{G_n}(h) \leq -\varepsilon$ for some positive *constant* $\varepsilon > 0$. Our graph G_n, as shown in Fig. 6, consists of two subgraphs $H_1 = (V_1, E_1)$ and $H_2 = (V_2, E_2)$ connected by an edge, where H_1 has $n_1 = \sqrt{n} + 2$ nodes and $m_1 = 2\sqrt{n} + 1$ edges H_2 is a complete 10-ary tree having $n_2 = n - n_1$ nodes and $m_2 = n_2 - 1 = n - \sqrt{n} - 3$ edges. (since the graph is unweighted, we omit mentioning the weights of the edges). Thus, the total number of edges of G_n is $m = m_1 + m_2 + 1 = n + \sqrt{n} + 2$. Let L_2 be the set of leaf nodes of H_2. We calculate the number of leaves $\ell_2 = |L_2|$ of H_2 in the following simple manner. Letting k be the depth of H_2, we have $\sum_{j=0}^{k} 10^j = n_2$. This gives the number of leaves ℓ_2 of H_2 as $\ell_2 = 10^k = \frac{9n_2+1}{10} = \frac{9n-9\sqrt{n}-17}{10}$. The edge $f = \{s, t\}$ shown in Fig. 6 is the edge for which we will show that $w^{(1)}(f) < 0$.

Consider any edge $e = \{u, v\} \in E$ of G_n and the associated distributions $\mathbb{P}_{\mathsf{Nbr}_{G_n}(u)}$ and $\mathbb{P}_{\mathsf{Nbr}_{G_n}(v)}$ as defined before. The (standard) *total variation distance* $||\mathbb{P}_{\mathsf{Nbr}_{G_n}(u)} - \mathbb{P}_{\mathsf{Nbr}_{G_n}(v)}||_{\text{TVD}}$ between the two distributions $\mathbb{P}_{\mathsf{Nbr}_{G_n}(u)}$ and $\mathbb{P}_{\mathsf{Nbr}_{G_n}(v)}$ is defined as

$$||e||_{\text{TVD}} \stackrel{\text{def}}{=} ||\mathbb{P}_{\mathsf{Nbr}_{G_n}(u)} - \mathbb{P}_{\mathsf{Nbr}_{G_n}(v)}||_{\text{TVD}} =$$
$$\frac{1}{2} \times \Bigg(\sum_{\alpha \in \mathsf{Nbr}_{G_n}(u) \cap \mathsf{Nbr}_{G_n}(v)} \big|\, \mathbb{P}_{\mathsf{Nbr}_{G_n}(u)}(\alpha) - \mathbb{P}_{\mathsf{Nbr}_{G_n}(v)}(\alpha) \,\big|$$
$$+ \sum_{\beta \in \mathsf{Nbr}_{G_n}(u) \setminus \mathsf{Nbr}_{G_n}(v)} \mathbb{P}_{\mathsf{Nbr}_{G_n}(u)}(\beta) + \sum_{\gamma \in \mathsf{Nbr}_{G_n}(v) \setminus \mathsf{Nbr}_{G_n}(u)} \mathbb{P}_{\mathsf{Nbr}_{G_n}(v)}(\gamma) \Bigg)$$

By Proposition 1 of [70] we have

$$1 - 3 \times ||e||_{\text{TVD}} \leq \mathfrak{C}_{G_n}(e) \leq 1 - ||e||_{\text{TVD}} \tag{15}$$

Now suppose that the following condition holds for the edge e:

$$\forall\, \alpha \in \mathsf{Nbr}_{G_n}(u) \setminus \{u, v\} \;\; \forall\, \beta \in \mathsf{Nbr}_{G_n}(v) \setminus \{u, v\} : \\ \text{dist}_{G_n}(\alpha, \beta) = 3 \quad \text{(C1)}$$

Using the discussions surrounding Proposition 1 and Proposition 2 of [70], we get the following bound for this case:

$$\begin{aligned} &\text{EMD}_{\,e}(\mathbb{P}_{\mathsf{Nbr}_{G_n}(u)}, \mathbb{P}_{\mathsf{Nbr}_{G_n}(v)}) \geq \\ &\quad 3 \times ||e||_{\text{TVD}} - \left| \mathbb{P}_{\mathsf{Nbr}_{G_n}(u)}(u) - \mathbb{P}_{\mathsf{Nbr}_{G_n}(v)}(u) \right| \\ &\quad\quad - \left| \mathbb{P}_{\mathsf{Nbr}_{G}(u)}(v) - \mathbb{P}_{\mathsf{Nbr}_{G}(v)}(v) \right| \end{aligned} \tag{16}$$

Finally, suppose that edge e satisfies $\mathsf{Nbr}_{G_n}(u) = \{u, v\}$. Since $\text{dist}_{G_n}(u, \alpha) = 2$ for all $\alpha \in \mathsf{Nbr}_{G_n}(v) \setminus \{u, v\}$, in this case we get the following bound:

$$\begin{aligned} &\text{EMD}_{\,e}(\mathbb{P}_{\mathsf{Nbr}_{G_n}(u)}, \mathbb{P}_{\mathsf{Nbr}_{G_n}(v)}) \geq \\ &\frac{1}{2} \times \left[\frac{2(\mathsf{deg}_{G_n}(v) - 1)}{\mathsf{deg}_{G_n}(v) + 1} + 2 \left(\frac{1}{2} - \frac{1}{\mathsf{deg}_{G_n}(v) + 1} \right) \right] \\ &\quad = \frac{3}{2} - \frac{1}{2\mathsf{deg}_{G_n}(v) + 2} \end{aligned} \tag{17}$$

We now use relatively straightforward calculations to calculate the Ricci curvature values of various edges of G_n:

***(i)*:** For the edge $f = \{s, t\}$, we get

$$\begin{aligned} ||f||_{\text{TVD}} &= \tfrac{\sqrt{n}+2}{2} \times \left(\tfrac{1}{\sqrt{n}+2} - \tfrac{1}{\sqrt{n}+3} \right) + \tfrac{1}{2(\sqrt{n}+3)} \\ &= \tfrac{1}{2} - \tfrac{\sqrt{n}+1}{2\sqrt{n}+6} = \tfrac{1}{\sqrt{n}+3} \end{aligned}$$

and thus using (15) we get $\mathfrak{C}_{G_n}(f) \geq 1 - \frac{3}{\sqrt{n}+3} = 1 - o(1)$, as required.

***(ii)*:** For any edge $e = \{p_i, s\} \in E_1$ for $i \in \{1, \ldots, \sqrt{n}\}$, we have

$$||e||_{\text{TVD}} = \frac{1}{2} \times \frac{1}{4} + 3 \times \frac{1}{2} \times \left(\frac{1}{3} - \frac{1}{4} \right) = \frac{1}{4}$$

and thus using (15) we get

$$\Lambda_1 = \sum_{e = \{p_i, s\} \in E_1,\, i \in \{1, \ldots, \sqrt{n}\}} \mathfrak{C}_{G_n}(e) \leq \frac{3}{4}\sqrt{n}$$

***(iii)*:** Since trivially $\mathfrak{C}_{G_n}(e) \leq 1$ for any edge $e = \{p_i, t\} \in E_1$ for $i \in \{1, \ldots, \sqrt{n}\}$, we get

$$\Lambda_2 = \sum_{e = \{p_i, t\} \in E_1,\, i \in \{1, \ldots, \sqrt{n}\}} \mathfrak{C}_{G_n}(e) \leq \sqrt{n}$$

***(iv)*:** For any edge $e = \{\{r_i, r_j\} \,|\, r_i \in L_2\} \in E_2$ connecting a leaf node r_i to another non-leaf node r_j in H_2, since $\mathsf{Nbr}_{G_n}(r_i) = \{r_i, r_j\}$ and $\mathsf{deg}_{G_n}(v) = 11$, using (17) we get $\text{EMD}_{\,e}(\mathbb{P}_{\mathsf{Nbr}_{G_n}(r_i)}, \mathbb{P}_{\mathsf{Nbr}_{G_n}(r_j)}) \geq \frac{3}{2} - \frac{1}{24} = \frac{35}{24}$, and thus

$$\begin{aligned} \Lambda_3 = \sum_{e = \{\{r_i, r_j\} \,|\, r_i \in L_2\} \in E_2} \mathfrak{C}_{G_n}(e) &\leq \left(1 - \frac{35}{24}\right) \times \ell_2 \\ &= \frac{-99n + 99\sqrt{n} + 187}{240} \end{aligned}$$

***(v)*:** Condition (C1) applies to any edge $e = \{\{r_i, r_j\} \,|\, r_i, r_j \notin L_2\} \in E_2$ connecting a pair of non-leaf nodes r_i, r_j in H_2, or to the edge $\{u, r_1\}$. Thus, by using (16) we get the following bounds:

- For an edge $e = \{\{r_i, r_j\} \,|\, r_i, r_j \notin L_2\} \in E_2$, $\text{EMD}_{\,e}(\mathbb{P}_{\mathsf{Nbr}_{G_n}(r_i)}, \mathbb{P}_{\mathsf{Nbr}_{G_n}(r_j)}) = \frac{5}{4}$ and thus

$$\begin{aligned} \Lambda_4 = \sum_{e = \{\{r_i, r_j\} \,|\, r_i, r_j \notin L_2\} \in E_2} &\mathfrak{C}_{G_n}(e) \\ &\leq -\frac{1}{4} \times (n_2 - \ell_2) = \frac{-n + \sqrt{n} + 3}{10} \end{aligned}$$

- If $e = \{u, r_1\}$ then

$$\begin{aligned} &\text{EMD}_{\,e}(\mathbb{P}_{\mathsf{Nbr}_{G_n}(r_1)}, \mathbb{P}_{\mathsf{Nbr}_{G_n}(u)}) = \\ &\quad 3 \times \frac{1}{2} \times \left(\frac{10}{12} + \frac{\sqrt{n}+1}{\sqrt{n}+3} \right) = \frac{5}{4} + \frac{3}{2} \times \frac{\sqrt{n}+1}{\sqrt{n}+3} \end{aligned}$$

and thus

$$\Lambda_5 = \mathfrak{C}_{G_n}(e) \leq -\frac{1}{4} - \frac{3}{2} \times \frac{\sqrt{n}+1}{\sqrt{n}+3}$$

Adding up the relevant quantities, we get $\sum_{h \in E} \mathfrak{C}_{G_n}(h) \leq \mathfrak{C}_{G_n}(f) + \sum_{j=1}^{5} \Lambda_j \leq 1 + \sum_{j=1}^{5} \Lambda_j$. Since $m = m_1 + m_2 + 1 = n + \sqrt{n} + 2$ and $s > 0$ is a constant, it now follows that $\lim_{n \to \infty} \frac{s}{m} \sum_{h \in E} \mathfrak{C}_{G_n}(h) \leq -\frac{123\, s}{240}$, thus there exists a constant $\varepsilon > 0$ such that $\frac{s}{m} \sum_{h \in E} \mathfrak{C}_{G_n}(h) < -\varepsilon$ for all sufficiently large n. □

D. Rapid Initial Convergence of the Ricci Flow for Directed and Undirected Hypergraphs

As we shown in Table IV below, the first iteration in which the edge-weights converge under Equation (5) is a *small* number for all of our (directed and undirected) hypergraphs.

Table IV show fast initial convergence of Ricci flows via small value of the parameter Δ_{AVE}. However, as mentioned in Section III A, it is preferable to execute more iterations to ensure that the diffusion process has actually converged in several successive iterations based on the value of Δ_{AVE} and that the quality parameters are

TABLE IV. Fast convergence of the Ricci flows for all constructed hypergraphs (cf. Section II B). η_{first} is the *smallest* iteration after which $\Delta_{\text{AVE}} \leq \varepsilon$; $\varepsilon = 0.005$ for directed hypergraphs and $\varepsilon = 0.000005$ for undirected hypergraphs.

Hypergraph name	η_{first}
Escherichia Coli	4
Homo Sapiens	4
Helicobacter Pylori	4
Methanosarcina berkeri str. Fusaro	4
Mycobacterium Tuberculosis	4
Synechococcus elongatus	4
Synechocystis	4
CSP dataset	10
NSP dataset	5

within acceptable bounds. Moreover, even if Δ_{AVE} is within acceptable bounds, some individual edge weights may still change significantly in subsequent iterations since the value of Δ_{STD} (*cf.* Equation (6)) may not be acceptably low, and this may lead to changes in the cores. In our case, we indeed found that further iterations produced decreasing values of Δ_{STD} leading to more stable cores (see Table V).

E. Final Cores and Their Qualities Determined by Our Algorithm

We report in Table VI and Table VII the cores found by our algorithm with their quality parameters via Equations (8)–(12). As can be seen, taken together the parameters $\varkappa^{deg}_{\text{in}}$, $\varkappa^{deg}_{\text{out}}$, $\varkappa^{directed}_{\text{dist\_stretch}}$, $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$ for directed hypergraphs and the parameters $\varkappa^{deg}$, $\varkappa^{undirected}_{\text{dist\_stretch}}$, $\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}}$ for undirected hypergraphs indicate a good quality of modularity and centrality of the cores and satisfy all the validity criteria set forth in Sections II C 1–II C 5. For example, for *Synechococcus elongatus* hyperedges with only nodes from the core both in their head and tail contributed 80% on average to the in-degrees of the nodes and similarly hyperedges with only nodes from the core both in their head and tail contributed 79% on average to the out-degrees of the nodes. Furthermore, for *Synechococcus elongatus* about 73% of ordered pairs of nodes not in the core that were connected by a path lose this path when the core is removed, and ordered node pairs that stay connected increase their distance by about 272% on average.

An inspection of the values in Table VI and Table VII shows that removal of the cores disconnect *fewer* pairs of nodes in the core for undirected hypergraphs as compared to the directed hypergrahs (*i.e.*, the $\varkappa^{\text{unordered\_pairs}}_{\text{disconnected}}$ values are smaller than the $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$ values), even though the stretch factors of shortest paths surviving after core removals are comparable (*i.e.*, the $\varkappa^{deg}$, $\varkappa^{undirected}_{\text{dist\_stretch}}$ values are of similar magnitudes to the values of $\varkappa^{directed}_{\text{dist\_stretch}}$). There are *two* possible reasons for this behavior. Firstly, the sizes of cores for undirected hypergraphs are *smaller* than the typical core sizes of directed hypergraphs, and intuitively one would expect removal of *fewer* nodes to disconnect less number of remaining paths. Secondly, the *directionality* constraints on paths for directed hypergraphs (*i.e.*, edges may *not* be traversed in the wrong direction) allow fewer avenues to "bypass" the nodes in the core; indeed, majority of biochemical reactions in our dataset are *not* bidirectional reactions.

In the following two sections, we comment on interpreting the cores and their implications to future research works.

1. *Interpretation and Usefulness of Cores for Directed Hypergraphs (Metabolic Systems)*

A core $\mathcal{S} \subset V$ of the directed hypergraph $G = (V, E, w)$ corresponds to a set of molecules (reactants and/or products) in the biochemical system under consideration. Our analysis indicates the following properties for this subset of molecules:

(I) The high values of $\varkappa^{deg}_{\text{in}}$ and $\varkappa^{deg}_{\text{out}}$ in Table VI suggest that each molecule (node) u in the core is highly dependent on other molecules (nodes) in the core, *e.g.*, for another molecule v in the core either u and v are both reactants in the same biochemical reaction, which itself is part of the core or one of the them is a product produced in a core biochemical reaction in which the other one is a reactant.

(II) The values of $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$ and $\varkappa^{directed}_{\text{dist\_stretch}}$ signify the importance of the molecules in the core in the overall functioning of the biochemical system in the following manner.

- **(A)** Consider an ordered pair (u, v) which contributes towards the value of $\varkappa^{\text{ordered\_pairs}}_{\text{disconnected}}$ by losing their path when the core is removed. This implies that the production of v can be significantly reduced or perhaps completely disrupted by the removal of the core $\mathcal{S}$.
- **(B)** Consider an ordered pair (u, v) such that $\frac{\text{dist}_{H\setminus\mathcal{S}}(u,v)}{\text{dist}_{H\setminus\mathcal{S}}(u,v)}$ is large enough to contribute significantly towards the value of $\varkappa^{directed}_{\text{dist\_stretch}}$. Then removal of the core $\mathcal{S}$ may significantly delay the production of v by increasing the number of reactions needed for its production.

Since removal of the core(s) *significantly* affects the overall functioning of the biochemical system, one can conclude that the core plays a dominant role in the system. Following a standard practice used in computa-

TABLE V. Values of Δ_{STD} (*cf.* Equation (6)) value **at the end** of η Ricci flows iterations for all constructed hypergraphs (cf. Section II B). For all these cases, as observed in Table IV, the value of Δ_{AVE} is at most $\varepsilon = 0.005$ for directed hypergraphs and is at most $\varepsilon = 0.000005$ for undirected hypergraphs.

	Δ_{STD} values at end of $\eta =$				
Hypergraph name	$\eta^{\ddagger}_{\mathrm{conv}}$	10	20	30	40
Escherichia Coli	0.0818	0.0650	0.0498	0.0411	0.0358
Homo Sapiens	0.0286	0.0199	0.0136	0.0131	0.0123
Helicobacter Pylori	0.0037	0.0025	0.0020	0.0017	0.0015
Methanosarcina berkeri str. Fusaro	0.0872	0.0592	0.0554	0.0461	0.0403
Mycobacterium Tuberculosis	0.0884	0.0600	0.0530	0.0441	0.0385
Synechococcus elongatus	0.0184	0.0132	0.0100	0.0082	0.0071
Synechocystis	0.1171	0.0837	0.0628	0.0515	0.0446
CSP dataset	0.0024	0.0024	0.0017	0.0014	0.0012
NSP dataset	0.0189	0.0138	0.0103	0.0091	0.0085

‡the smallest value of η for which $\Delta_{\mathrm{AVE}} \leq \varepsilon$

TABLE VI. For directed hypergraphs, cores with their quality parameters (*cf.* Equations (8),(9),(11),(10)) as found by our algorithm. **The *p*-values for the core quality parameters (*cf.* Section II C 5) for all cores were found to be significantly less than 10^{-5}, so these values are not listed explicitly**.

Hypergraph (V, E, w)	$\lvert V\rvert$	$\mathrm{core}^{\dagger}_{\#}$	$\mathrm{core\_size}^{\ddagger}$	core quality parameters			
				$\mathscr{r}^{deg}_{\mathrm{in}}$	$\mathscr{r}^{deg}_{\mathrm{out}}$	$\mathscr{r}^{directed}_{\mathrm{dist\_stretch}}$	$\mathscr{r}^{\mathrm{ordered\_pairs}}_{\mathrm{disconnected}}$
Escherichia Coli	762	1	326	0.7259	0.7340	2.7043	0.1839
Homo Sapiens	343	1	144	0.6960	0.7400	2.7065	0.4303
Helicobacter Pylori	486	1	230	0.7859	0.7501	2.7065	0.3215
Methanosarcina berkeri str. Fusaro	629	1	254	0.7829	0.7590	2.7337	0.2463
Mycobacterium Tuberculosis	826	1	377	0.7993	0.7092	2.7236	0.2886
Synechococcus elongatus	769	1	305	0.8063	0.7907	2.7211	0.7345
Synechocystis	796	1	305	0.8058	0.7694	2.6035	0.1622

$^{\dagger}\mathrm{core}_{\#}$: number of cores
$^{\ddagger}\mathrm{core\_size}$: number of nodes in the cores

tional biology, researchers may focus on the molecules in the core and their associated biochemical reactions for further computational analysis if the original system was too computationally intensive to analyse because of its size.

a. Removing a core by biological experiments Perturbation experiments (*e.g.*, genetic knockouts) are well-established biological method for probing the importance of biochemical entities. The best way to eliminate a chemical reaction is to knock out the enzyme that catalyzes the reaction. We *briefly* comment here on *optimally* designing the biological experimental mechanism to remove a core $\mathcal{S}$, assuming that we have the data corresponding to the *enzymes* for each reaction (our datasets did *not* provide this information). Let $\mathcal{E}$ be the set of all biochemical reactions in which one or more members of $\mathcal{S}$ appear as either a reactant or a product (or both); thus, disabling the reactions in $\mathcal{E}$ will effectively disconnect the core $\mathcal{S}$. Let $\mathcal{C}$ be the set of enzymes catalyzing the reactions from $\mathcal{E}$. Then, a *minimal* set of enzyme knockouts that can be used to disable all the reactions in $\mathcal{E}$ can be determined by solving an appropriate *minimum hitting set* problem [14] defined such that the universe is $\mathcal{E}$ and corresponding to each enzyme $c \in \mathcal{C}$ we have a set $\{e \in \mathcal{E} \mid c \text{ is a catalyst for } e\}$. The reader is referred to standard literatures in computer algorithms such as [96]

TABLE VII. For undirected hypergraphs, cores with their quality parameters (*cf.* Equations (7),(13),(12)) as found by our algorithm. **The *p*-values for the core quality parameters (*cf.* Section II C 5) for all cores were found to be significantly less than 10^{-5}, so these values are not listed explicitly**.

Hypergraph (V, E, w)	$\lvert V\rvert$	$\text{core}^{\dagger}_{\#}$	$\text{core\_size}^{\ddagger}$	core quality parameters		
				$\curlyvee^{deg}$	$\curlyvee^{undirected}_{\text{dist\_stretch}}$	$\curlyvee^{\text{unordered\_pairs}}_{\text{disconnected}}$
Csp dataset	496	1	62[a]	0.6411	2.8964	0.0006
Nsp dataset	518	2	24[b]	0.7530	3.4574	0.0002
			23[c]	0.7600	1.6309	0.0001

$^{\dagger}\text{core}_{\#}$: number of cores
$^{\ddagger}\text{core\_size}$: number of nodes in the cores

[a] Authors in the core:
M. Bellare, R. Impagliazzo, M. Szegedy, L. Fortnow, M. Yung, B. Waters, D. Dolev, L. Babai, M. Luby, A. Lysyanskaya, A. Wigderson, A. Sahai, M. Sudan, J. Naor, T. Okamoto, L. Levin, Omer Reingold, N. Buchbinder, P. Rogaway, C. Dwork, H. Krawczyk, H. Buhrman, O. Goldreich, B. Barak, D. Boneh, N. Nisan, Y. Lindell, S. Halevi, S. Keelveedhi, A. Herzberg, S. Wolf, G. Segev, R. Rivest, S. Vadhan, C. Peikert, R. Santhanam, J. Stern, E. Fujisaki, V. Kabanets, S. Goldwasser, L. Trevisan, S. Jarecki, A. O'Neill, G. Rothblum, M. Kharitonov, R. Schwartz, D. Melkebeek, M. Naor, R. Canetti, A. Goldberg, S. Micali, A. Boldyreva, D. Pointcheval, S. Arora, K. Yang, N. Linial, J. Hastad, S. Rudich, E. Sweedyk, R. Ostrovsky, A. Desai, M. Feldman

[b] Authors in the core:
J. Zhuang, C. Bauch, M. Perc, Y. Tian, A. Sheikhahmadi, C. Hens, M. Duh, Y. Mu, Y. Xia, W. Lin, P. Ji, K. Skok, M. Milojevic, J. Ye, J. Sun, J. Kurths, Z. Cheng, A. Zareie, J. Cao, Y. Tang, L. Guerrini, M. S. K. Fasaei, L. Tang, M Gosak

[c] Authors in the core:
X. Wu, W. Han, D. Zhao, C. Lv, E. M. Ruiz, P. A. C. Sousa, L.-L. Jiang, A. Nicchi, S. Boccaletti, J. Gao, Z. Wang, D. Duan, E. Kubik, H. Stanley, S. Havlin, L. Wang, S. Li, Q. Su, A. Li, S. Si, D. Li, M. Zanin, D. Papo

for methods to solve a minimum hitting set problem efficiently.

2. *Interpretation and Usefulness of Cores for Undirected Hypergraphs (Co-author Relationships)*

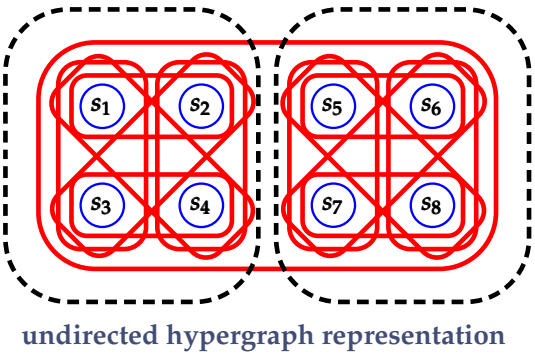
A core $\mathcal{S} \subset V$ of the undirected hypergraph $G = (V, E, w)$ corresponds to a set of authors. Below we explain what the core signifies in terms of its properties:

(I) The values of $\curlyvee^{deg}$ in Table VII suggest that the authors in $\mathcal{S}$ form a "close group" of collaborators in the sense that they wrote more papers with each other as compared to with authors outside the core.

We show by an illustrative example that the above need *not* be the case if cores are found by graph-theoretic approaches. For some large even $k > 2$, suppose that authors $a_1, \ldots, a_k$ wrote a paper, authors a_i, a_j wrote a paper for $1 \leq i < j \leq k/2$ and authors a_i, a_j wrote a paper for $k/2 + 1 \leq i < j \leq k$ (see Fig. 7 for a visual illustration when $k = 10$). The standard graph-theoretic approach (*cf.* Fig. 5) will group all the k nodes $a_1, \ldots, a_k$ in the same core since they form a large k-clique, whereas hypergraph-theoretic approach will correctly group them in two cores containing $a_1, \ldots, a_{k/2}$ and $a_{1+k/2}, \ldots, a_k$, respectively.

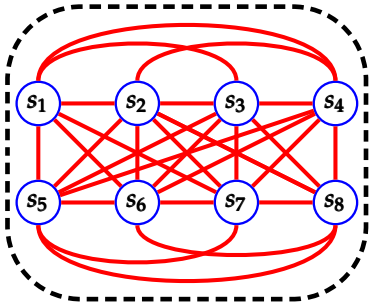
(II) The values of $\curlyvee^{undirected}_{\text{dist\_stretch}}$ and $\curlyvee^{\text{unordered\_pairs}}_{\text{disconnected}}$ signify the importance of the authors in $\mathcal{S}$ in *promoting* collaborations between other researchers in the following manner. Note that for a path $\mathscr{P}_{x,y} = (x = v_1, e_1, \ldots, v_k, e_k, v_{k+1} = y)$ between nodes x and y v_i and v_{i+1} are co-authors for $i = 1, \ldots, k$. Consider a pair of authors $\{x, y\}$ contributing towards the value of $\curlyvee^{\text{unordered\_pairs}}_{\text{disconnected}}$. This implies complete disruption of collaborations between any pairs of authors where one of them is in the set of authors reachable from u via chains of collaboration and the other one is in the set of authors reachable from v via chains of collaboration. If for a pair of authors $\{x, y\}$ $\frac{\text{dist}_{H\setminus\mathcal{S}}(u,v)}{\text{dist}_{H\setminus\mathcal{S}}(u,v)}$ was large enough to contribute significantly towards the value of $\curlyvee^{undirected}_{\text{dist\_stretch}}$ then, even though some collaborative chains connecting u and v are not completely disrupted, they are elongated leading to a decrease in productivity.

FIG. 7. A visual illustration of the example discussed in item **(I)** in Section III E 2 for $k = 8$. The cores are shown in dotted black lines. The hypergraph representation captures the cores correctly whereas the graph representation does not.



undirected hypergraph representation

graph representation



IV. CONCLUSION

In this article, we have designed and implemented an algorithmic paradigm for finding cores in edge-weighted directed and undirected hypergraphs using a hypergraph-curvature guided discrete time diffusion process, and have *successfully* applied our methods to seven metabolic hypergraphs and two social (co-authorship) hypergraphs. *En route*, we have also shown that an edge-weight renormalization procedure in a prior research work for Ricci flows has undesirable properties. Finding cores of hypergraphs is *still* a relatively new research topic that is growing rapidly, and we expect that our our work will provide further impetus and guidance to this burgeoning research area. We conclude by listing a few research questions in this direction that may be of interest to researchers:

▷ Our calculations of Ricci curvatures of directed hypergraphs in Section II A 1 and directed hypergraphs in Section II A 2 is a generalization of the corresponding calculations for undirected and directed graphs, respectively. However, it is possible to carry out these generalizations in different ways, and thus it will be of interest to know if other generalizations produce better qualities for cores of hypergraphs.

▷ It would be of interest to see if the discrete time diffusion process using Ricci curvatures *can* be combined with random walks in hypergraphs [54] for application domains such as modular decompositions of graphs.

▷ In spite of writing our code in Python (which runs slower than codes written in C or similar other programming languages) and using a single older (slower) laptop, we were able to finish our experimental work within reasonable time, and therefore we made no attempt to rewrite the code in a more suitable programming language, optimize the code and use a faster computational device. We believe this could be due to two reasons: **(*i*)** our hypergraphs were of moderate size (number of nodes) and moderate density (number and size of hyperedges), and **(*ii*)** our Ricci flow iterations converged within a few steps. However, computational speed could become an issue with larger hypergraphs or if Ricci flows required more steps to converge. Here we provide a few suggestions to researchers to overcome possible future computational bottlenecks. Note that the most computational intensive part of the curvature calculations involve the following two computational components: **(*a*)** computing the Earth Mover's Distance ($\text{EMD}_H(\mathbb{P}_{\text{left}}, \mathbb{P}_{\text{right}})$) for each directed hyperedge or for each pair of nodes within an undirected hyperedge for every iteration, and **(*b*)** the all-pairs distance calculations once for the entire hypergraph for every iteration.

For all-pairs distance calculations we either used a straightforward adoption of Floyd-Warshall's all-pair shortest path calculations with early terminations for graphs to our hypergraphs, or used a simple breadth-first-search based approach. However, the standard all-pairs-shortest-path problems for graphs have a long and rich algorithmic history [97] with strong connections to matrix multiplication algorithms [98, 99] and other problems [100], and future researchers could implement some of these advanced algorithmic methods for their computations. Note that in many applications it may even suffice to compute the distances *approximately* and in that case algorithms such as in [101] could be useful.

We have the following suggestions for future researchers regarding the EMD calculations:

▷ Note that within each iteration, the calculations of the EMD values for the hyperedges can be done in parallel, and thus a clustered computing could be used.

▷ If it suffices to compute the EMD values approximately, one could implement the ε-additive approximation algorithms in [102, 103].

Appendix: Details of Calculations of $\mathbb{P}_{\text{right}}$ for Weighted Directed Hypergraphs in Section II A 1

- ▷ Initially, $\mathbb{P}_{\text{right}}(u) = 0$ for all $u \in V$. In our subsequent steps, we will add to these values as appropriate.
- ▷ We divide the total probability 1 equally among the nodes in $\mathcal{H}\text{ead}_e$, thus "allocating" a value of $(|\mathcal{H}\text{ead}_e|)^{-1}$ to each node in question.
- ▷ For every node $x \in \mathcal{H}\text{ead}_e$ with $\mathsf{deg}_x^{out} = 0$, we add $(|\mathcal{H}\text{ead}_e|)^{-1}$ to $\mathbb{P}_{\text{left}}(x)$.
- ▷ For every node $x \in \mathcal{H}\text{ead}_e$ with $\mathsf{deg}_x^{out} > 0$, we perform the following:
  - ▷ We divide the probability $(|\mathcal{H}\text{ead}_e|)^{-1}$ equally among the hyperedges e' such that $x \in \mathcal{T}\text{ail}_{e'}$, thus "allocating" a value of $(|\mathcal{H}\text{ead}_e| \times \mathsf{deg}_x^{out})^{-1}$ to each hyperedge in question.
  - ▷ For each such hyperedge e' such that $x \in \mathcal{T}\text{ail}_{e'}$, we divide the allocated value equally among the nodes in $\mathcal{H}\text{ead}_{e'}$ and add this values to the probabilities of these nodes. In other words, for every node $y \in \mathcal{H}\text{ead}_{e'}$ we add $(|\mathcal{H}\text{ead}_e| \times \mathsf{deg}_x^{out} \times |\mathcal{H}\text{ead}_{e'}|)^{-1}$ to $\mathbb{P}_{\text{left}}(y)$.

Note that the final probability for each node is calculated by summing all the contributions from each bullet point.

ACKNOWLEDGMENTS

We thank Katie Kruzan for useful discussions and help in debugging our software.

[1] B. DasGupta and J. Liang, *Models and Algorithms for Biomolecules and Molecular Networks* (Wiley-IEEE Press, New Jersey, 2016).

[2] M. E. J. Newman, *Networks: An Introduction* (Oxford University Press, 2010).

[3] R. Albert and A.-L. Barabási, Statistical mechanics of complex networks, Reviews of Modern Physics **74**, 47 (2002).

[4] V. Colizza, A. Flammini, M. Serrano, and A. Vespignani, Detecting rich-club ordering in complex networks, Nature Physics **2**, 110 (2006).

[5] V. Latora and M. Marchiori, A measure of centrality based on network efficiency, New Journal of Physics **9**, 188 (2007).

[6] R. Albert, B. DasGupta, R. Hegde, G. S. Sivanathan, A. Gitter, G. Gürsoy, P. Paul, and E. Sontag, Computationally efficient measure of topological redundancy of biological and social networks, Physical Review E **84**, 036117 (2011).

[7] D. S. Bassett, N. F. Wymbs, M. A. Porter, P. J. Mucha, J. M. Carlson, and S. T. Grafton, Dynamic reconfiguration of human brain networks during learning, Proceedings of the National Academy of Sciences **118**, 7641 (2011).

[8] F. Battiston, G. Cencetti, I. Iacopini, V. Latora, M. Lucas, A. Patania, J.-G. Young, and G. Petri, Networks beyond pairwise interactions: Structure and dynamics, Physics Reports **874**, 1 (2020).

[9] F. Battiston, E. Amico, A. Barrat, G. Bianconi, G. F. de Arruda, B. Franceschiello, I. Iacopini, S. Kéfi, V. Latora, Y. Moreno, M. M. Murray, T. P. Peixoto, F. Vaccarino, and G. Petr, The physics of higher-order interactions in complex systems, Nature Physics **17**, 1093 (2021).

[10] L. Torres, A. S. Blevins, D. Bassett, and T. Eliassi-Rad, The why, how, and when of representations for complex systems, SIAM Review **63**, 435 (2021), https://doi.org/10.1137/20M1355896.

[11] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A.-L. Barabasi, The large-scale organization of metabolic networks, Nature **407**, 651 (2000).

[12] B. DasGupta, G. A. Enciso, E. Sontag, and Y. Zhang, Algorithmic and complexity results for decompositions of biological networks into monotone subsystems, Biosystems **90**, 161 (2007).

[13] C. Berge, *Hypergraphs: Combinatorics of Finite Sets*, 2nd ed. (Elsevier Science Publishers, New York, NY, 1989).

[14] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, 1st ed. (W. H. Freeman, 1979).

[15] M. R. Bridson and A. Häfliger, *Metric Spaces of Non-Positive Curvature*, 1st ed. (Springer-Verlag Berlin Heidelberg, 1999).

[16] M. Berger, *A Panoramic View of Riemannian Geometry*, 1st ed. (Springer-Verlag Berlin Heidelberg, 2003).

[17] R. Forman, Bochner's method for cell complexes and combinatorial ricci curvature, Discrete and Computational Geometry **29**, 323 (2003).

[18] R. P. Sreejith, K. Mohanraj, J. Jost, E. Saucan, and A. Samal, Forman curvature for complex networks, Journal of Statistical Mechanics: Theory and Experiment **2016**, 063206 (2016).

[19] R. P. Sreejith, J. Jost, E. Saucan, and A. Samal, Systematic evaluation of a new combinatorial curvature for complex networks, Chaos, Solitons and Fractals **101**, 50 (2017).

[20] M. Weber, E. Saucan, and J. Jost, Characterizing complex networks with forman-ricci curvature and associated geometric flows, Journal of Complex Networks **5**, 527 (2017).

[21] B. DasGupta, M. V. Janardhanan, and F. Yahyanejad, Why did the shape of your network change? (on detecting network anomalies via non-local curvatures), Algorithmica **82**, 1741 (2020).

[22] A. Samal, R. P. Sreejith, J. Gu, S. Liu, E. Saucan, and J. Jost, Comparative analysis of two discretizations of ricci curvature for complex networks, Scientific Reports

8, 8650 (2018).
[23] T. Chatterjee, R. Albert, S. Thapliyal, N. Azarhooshang, and B. DasGupta, Detecting network anomalies using forman-ricci curvature and a case study for human brain networks, Scientific Reports **11**, 10.1038/s41598-021-87587-z (2021).
[24] Y. Ollivier, A visual introduction to Riemannian curvatures and some discrete generalizations, in *Analysis and Geometry of Metric Measure Spaces: Lecture Notes of the 50th Séminaire de Mathématiques Supérieures (SMS), Montréal, 2011*, Vol. 56, edited by G. Dafni, R. J. McCann, and A. Stancu (American Mathematical Society, Providence, RI, USA, 2013) pp. 197–219.
[25] Y. Ollivier, Ricci curvature of markov chains on metric spaces, Journal of Functional Analysis **256**, 810 (2009).
[26] Y. Ollivier, A survey of ricci curvature for metric spaces and markov chains, in *Advanced Studies in Pure Mathematics*, Vol. 57, edited by M. Kotani, M. Hino, and T. Kumagai (Mathematical Society of Japan, 2010) pp. 343–381.
[27] Y. Ollivier, Ricci curvature of metric spaces, Comptes Rendus Mathematique **345**, 643 (2007).
[28] S. Asoodeh, T. Gao, and J. Evans, Curvature of hypergraphs via multi-marginal optimal transport, in *2018 IEEE Conference on Decision and Control* (2018) pp. 1180–1185.
[29] M. Eidi and J. Jost, Ollivier ricci curvature of directed hypergraphs, Scientific Reports **10**, 12466 (2020).
[30] C. Coupette, S. Dalleiger, and B. Rieck, Ollivier-ricci curvature for hypergraphs: A unified framework, in *The Eleventh International Conference on Learning Representations* (2023).
[31] T. Akamatsu, A new transport distance and its associated ricci curvature of hypergraphs, Analysis and Geometry in Metric Spaces **10**, 90 (2022).
[32] W. Leal, G. Restrepo, P. F. Stadler, and J. Jost, Forman-ricci curvature for hypergraphs, Advances in Complex Systems **24**, 2150003 (2021), https://doi.org/10.1142/S021952592150003X.
[33] R. S. Hamilton, Three-manifolds with positive Ricci curvature, Journal of Differential Geometry **17**, 255 (1982).
[34] G. Perelman, The entropy formula for the ricci flow and its geometric applications, arXiv preprint arXiv:math/0211159v1 10.48550/arXiv.math/0211159 (2002).
[35] C.-C. Ni, Y.-Y. Lin, F. Luo, and J. Gao, Community detection on networks with ricci flow, Scientific Reports **9**, 9984 (2019).
[36] J. Sia, E. Jonckheere, and P. Bogdan, Ollivier-ricci curvature-based method to community detection in complex networks, Scientific Reports **9**, 9800 (2019).
[37] X. Lai, S. Bai, and Y. Lin, Normalized discrete ricci flow used in community detection, Physica A: Statistical Mechanics and its Applications **597**, 127251 (2022).
[38] M. Weber, J. Jost, and E. Saucan, Forman-ricci flow for change detection in large dynamic data sets, Axioms **5**, 10.3390/axioms5040026 (2016).
[39] E. Cohen, Y. Nachshon, A. Maril, P. M. Naim, J. Jost, and E. Saucan, Object-based dynamics: Applying forman-ricci flow on a multigraph to assess the impact of an object on the network structure, Axioms **11**, 10.3390/axioms11090486 (2022).
[40] C.-C. Ni, Y.-Y. Lin, J. Gao, and X. Gu, Network alignment by discrete ollivier-ricci flow, in *Graph Drawing and Network Visualization*, edited by T. Biedl and A. Kerren (Springer International Publishing, 2018) pp. 447–462.
[41] J. Kim, H. J. Jeong, S. Lim, and J. Kim, Effective and efficient core computation in signed networks, Information Sciences **634**, 290 (2023).
[42] A. Fornito, A. Zalesky, and E. Bullmore, *Fundamentals of brain network analysis*, 1st ed. (Academic press, 2016).
[43] O. Sporns and R. F. Betzel, Modular brain networks, Annual Review of Psychology **67**, 613 (2016).
[44] L. Harriger, M. P. van den Heuvel, and O. Sporns, Rich club organization of macaque cerebral cortex and its role in network communication, PLoS ONE **7**, e46497 (2012).
[45] J. Kitazono, R. Kanai, and M. Oizumi, Efficient search for informational cores in complex systems: Application to brain networks, Neural Networks **132**, 232 (2020).
[46] M. E. J. Newman and M. Girvan, Finding and evaluating community structure in networks, Physical Review E **69**, 026113 (2004).
[47] E. A. Leicht and M. E. J. Newman, Community structure in directed networks, Physical Review Letters **100**, 118703 (2008).
[48] M. E. J. Newman, Modularity and community structure in networks, Proceedings of the National Academy of Sciences **103**, 8577 (2006), https://www.pnas.org/content/103/23/8577.full.pdf.
[49] B. DasGupta and D. Desai, On the complexity of Newman's community finding approach for biological and social networks, Journal of Computer and System Sciences **79**, 50 (2013).
[50] B. DasGupta, Computational complexities of optimization problems related to model based clustering of networks, in *Optimization in Science and Engineering*, edited by T. Rassias, C. Floudas, and S. Butenko (Springer, New York, NY, 2014) pp. 97–113.
[51] F. Tudisco and D. J. Higham, Core-periphery detection in hypergraphs, SIAM Journal on Mathematics of Data Science **5**, 1 (2023), https://doi.org/10.1137/22M1480926.
[52] I. Chien, C.-Y. Lin, and I.-H. Wang, Community detection in hypergraphs: Optimal statistical limit and efficient algorithms, in *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, Vol. 84, edited by A. Storkey and F. Perez-Cruz (PMLR, 2018) pp. 871–879.
[53] N. Ruggeri, M. Contisciani, F. Battiston, and C. D. Bacco, Community detection in large hypergraphs, Science Advances **9**, 10.1126/sciadv.adg9159 (2023), https://www.science.org/doi/pdf/10.1126/sciadv.adg9159.
[54] T. Carletti, D. Fanelli, and R. Lambiotte, Random walks and community detection in hypergraphs, Journal of Physics: Complexity **2**, 015011 (2021).
[55] A. Eriksson, D. Edler, A. Rojas, M. de Domenico, and M. Rosvall, How choosing random-walk model and network representation matters for flow-based community detection in hypergraphs, Communications Physics **4**, 1093 (2021).
[56] J. Kritschgau, D. Kaiser, O. A. Rodriguez, I. Amburg, J. Bolkema, T. Grubb, F. Lan, S. Maleki, P. Chodrow, and B. Kay, Community detection in hypergraphs via mutual information maximization., Scientific Reports

14, 10.1038/s41598-024-55934-5 (2024).
[57] Y. Zhen and J. Wang, Community detection in general hypergraph via graph embedding, Journal of the American Statistical Association **118**, 1620 (2023), https://doi.org/10.1080/01621459.2021.2002157.
[58] A. Eriksson, T. Carletti, R. Lambiotte, A. Rojas, and M. Rosvall, Flow-based community detection in hypergraphs, in *Higher-Order Systems*, edited by F. Battiston and G. Petri (Springer International Publishing, Cham, 2022) pp. 141–161.
[59] M. Mancastroppa, I. Iacopini, G. Petri, and A. Barrat, Hyper-cores promote localization and efficient seeding in higher-order processes, Nature communications **14**, 6223 (2023).
[60] M. Mancastroppa, I. Iacopini, G. Petri, and A. Barrat, The structural evolution of temporal hypergraphs through the lens of hyper-cores, EPJ Data Science **13**, 10.1140/epjds/s13688-024-00490-1 (2024).
[61] Y. Xu, F. Zhang, and B. Liu, The decomposition and maintenance of hypercores on edge-weighted hypergraphs, Mathematical Foundations of Computing (2023).
[62] D. Pretolani, Finding hypernetworks in directed hypergraphs, European Journal of Operational Research **230**, 226 (2013).
[63] A. P. Volpentesta, Hypernetworks in a directed hypergraph, European Journal of Operational Research **188**, 390 (2008).
[64] R. Albert, B. DasGupta, and N. Mobasheri, Topological implications of negative curvature for biological and social networks, Physical Review E **89**, 032811 (2014).
[65] R. S. Burt, *Structural Holes: The Social Structure of Competition* (Harvard University Press, Cambridge, MA, USA, 1992).
[66] C. L. Mallows, A Note on Asymptotic Joint Normality, The Annals of Mathematical Statistics **43**, 508 (1972).
[67] Y. Rubner, C. Tomasi, and L. J. Guibas, A metric for distributions with applications to image databases, in *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)* (1998) pp. 59–66.
[68] Y. Rubner, C. Tomasi, and L. J. Guibas, The earth mover's distance as a metric for image retrieval, International Journal of Computer Vision **40**, 99 (2000).
[69] C. Villani, Topics in optimal transportation, in *Graduate Studies in Mathematics*, Vol. 58 (American Mathematical Society, Providence, RI, USA, 2003) pp. 197–219.
[70] N. Azarhooshang, P. Sengupta, and B. DasGupta, A review of and some results for ollivier-ricci network curvature, Mathematics **8**, 10.3390/math8091416 (2020).
[71] M. Girvan and M. E. J. Newman, Community structure in social and biological networks, Proceedings of the National Academy of Sciences **99**, 7821 (2002), https://www.pnas.org/content/99/12/7821.full.pdf.
[72] S. Koujaku, I. Takigawa, M. Kudo, and H. Imai, Dense core model for cohesive subgraph discovery, Social Networks **44**, 143 (2016).
[73] F. Bonchi, D. García-Soriano, A. Miyauchi, and C. E. Tsourakakis, Finding densest k-connected subgraphs, Discrete Applied Mathematics **305**, 34 (2021).
[74] D. Boob, Y. Gao, R. Peng, S. Sawlani, C. Tsourakakis, D. Wang, and J. Wang, Flowless: Extracting densest subgraphs without flow computations, in *Proceedings of The Web Conference 2020*, WWW '20 (Association for Computing Machinery, New York, NY, USA, 2020) pp. 573–583.
[75] C. Chekuri, K. Quanrud, and M. R. Torres, Densest subgraph: Supermodularity, iterative peeling, and flow, in *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 1531–1555, https://epubs.siam.org/doi/pdf/10.1137/1.9781611977073.64.
[76] Y. Fang, W. Luo, and C. Ma, Densest subgraph discovery on large graphs: applications, challenges, and techniques, Proc. VLDB Endow. **15**, 3766 (2022).
[77] X. Liu, T. Ge, and Y. Wu, A stochastic approach to finding densest temporal subgraphs in dynamic graphs, IEEE Transactions on Knowledge and Data Engineering **34**, 3082 (2022).
[78] W. Luo, Z. Tang, Y. Fang, C. Ma, and X. Zhou, Scalable algorithms for densest subgraph discovery, in *2023 IEEE 39th International Conference on Data Engineering (ICDE)* (2023) pp. 287–300.
[79] C. Ma, R. Cheng, L. V. S. Lakshmanan, and X. Han, Finding locally densest subgraphs: a convex programming approach, Proc. VLDB Endow. **15**, 2719 (2022).
[80] C. Ma, Y. Fang, R. Cheng, L. V. S. Lakshmanan, W. Zhang, and X. Lin, On directed densest subgraph discovery, ACM Transaction Database Systems **46**, 10.1145/3483940 (2021).
[81] U. Feige, D. Peleg, and G. Kortsarz, The dense k-subgraph problem, Algorithmica **29**, 410 (2001).
[82] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan, Detecting high log-densities: an $O(n^{1/4})$ approximation for densest k-subgraph, in *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10 (Association for Computing Machinery, New York, NY, USA, 2010) pp. 201–210.
[83] S. K. Bera, S. Bhattacharya, J. Choudhari, and P. Ghosh, A new dynamic algorithm for densest subhypergraphs, in *Proceedings of the ACM Web Conference 2022*, WWW '22 (Association for Computing Machinery, New York, NY, USA, 2022) pp. 1093–1103.
[84] M. E. J. Newman, The structure and function of complex networks, SIAM Review **45**, 167 (2003).
[85] M. E. J. Newman, Detecting community structure in networks, European Physics Journal B **38**, 321 (2004).
[86] R. Kannan, P. Tetali, and S. Vempala, Simple markov-chain algorithms for generating bipartite graphs and tournaments, Random Structures & Algorithms **14**, 293 (1999).
[87] Z. A. King, J. Lu, A. Dräger, P. Miller, S. Federowicz, J. A. Lerman, A. Ebrahim, B. O. Palsson, and N. E. Lewis, Bigg models: A platform for integrating, standardizing and sharing genome-scale models, Nucleic acids research **44**, D515 (2016).
[88] R. Molontay and M. Nagy, Twenty years of network science: A bibliographic and co-authorship network analysis, in *Big Data and Social Media Analytics: Trending Applications*, edited by M. Çakırtaş and M. K. Ozdemir (Springer International Publishing, Cham, 2021) pp. 1–24.
[89] S. Goldwasser and S. Micali, Probabilistic encryption, Journal of Computer and System Sciences **28**, 270 (1984).
[90] R. M. Karp, Reducibility among combinatorial problems, in *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations*, edited by R. E. Miller, J. W. Thatcher,

and J. D. Bohlinger (Springer US, Boston, MA, 1972) pp. 85–103.
[91] N. Nisan and A. Wigderson, Hardness vs randomness, Journal of Computer and System Sciences **49**, 149 (1994).
[92] A.-L. Barabási and R. Albert, Emergence of scaling in random networks, Science **286**, 509 (1999).
[93] D. Watts and S. Strogatz, Collective dynamics of 'small-world' networks, Nature **393**, 440 (1998).
[94] *If the reaction times are known accurately then they can be used as the weights of the corresponding hyperedges*.
[95] B. DasGupta, E. Grigorescu, and T. Mukherjee, On computing discretized ricci curvatures of graphs: Local algorithms and (localized) fine-grained reductions, Theoretical Computer Science **975**, 114127 (2023).
[96] V. V. Vazirani, *Approximation Algorithms*, 1st ed. (Springer, Berlin, Heidelberg, 2010).
[97] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. (MIT Press and McGraw-Hill, 2009).
[98] R. Seidel, On the all-pairs-shortest-path problem in unweighted undirected graphs, Journal of Computer and System Sciences **51**, 400 (1995).
[99] U. Zwick, All pairs shortest paths using bridging sets and rectangular matrix multiplication, Journal of the ACM **49**, 289 (2002).
[100] V. V. Williams and R. R. Williams, Subcubic equivalences between path, matrix, and triangle problems, Journal of the ACM **65**, 1 (2018).
[101] D. Dor, S. Halperin, and U. Zwick, All-pairs almost shortest paths, SIAM Journal on Computing **29**, 1740 (2000).
[102] K. Quanrud, Approximating Optimal Transport With Linear Programs, in *2nd Symposium on Simplicity in Algorithms (SOSA 2019)*, OpenAccess Series in Informatics (OASIcs), Vol. 69, edited by J. T. Fineman and M. Mitzenmacher (Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2018) pp. 6:1—6:9.
[103] P. Dvurechensky, A. Gasnikov, and A. Kroshnin, Computational optimal transport: Complexity by accelerated gradient descent is better than by sinkhorn's algorithm, in *Proceedings of the 35th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 80, edited by J. Dy and A. Kraus (PMLR, 2018) pp. 1367—1376.