# On the Theory of Network Architectures in the Solar System Internet


**Alan Hylton**
**NASA Goddard**
**alan.g.hylton@nasa.gov**

**Oliver Chiriac**
**University of Oxford**
**oliver.chiriac@trinity.ox.ac.uk**

**Jacob Cleveland**
**NASA Glenn**
**jacob.a.cleveland@nasa.gov**

**Jihun Hwang**
**Purdue University**
**hwang102@purdue.edu**

**Daniel Koizumi**
**University of Texas Austin**
**daniel.koizumi@utexas.edu**

**Karuna Petwe**
**University of Washington**
**kpetwe@outlook.com**

**Tobias Timofeyev**
**University of Vermont**
**tobias.timofeyev@uvm.edu**



***Abstract*—Delay Tolerant Networking (DTN) is maturing into a viable enabling technology for the so-called Solar System Internet (SSI). The focus of SSI is shifting towards modern network architectures and scalability, which goes beyond the underlying protocol suite of DTN. Following a record-setting experiment campaign on the International Space Station (ISS), there is a wealth of operational experience and lessons-learned based on the advent of service-provider oriented architectures available to propel humanity's ability to network in space to new levels. However, deeper understandings of extending these architectures to the solar-system level are not fully explored. In this paper, we combine this new information with previous theoretical advances to open new doors in DTN network modeling with an eye on practical means to designing, creating, and operating future space networks.**

**The primary purpose of networking is scalability, however simply using DTN does not give this for free. Indeed, having a protocol suite does not inform the user on its best practices; in the case of DTN, best practices are not known. In particular, the ISS experiments illustrated the difficulty of uniting DTN network areas across project and programmatic boundaries. In traditional DTN routing, all nodes have the same schedule of contacts - known as a contact graph - and it is expected that these data are globally consistent. Approaches depending on omniscience neither scale nor generalize well, yet alternatives remain elusive as there is no standard temporospatial network modeling approach.**

**We investigate the capability of various mathematical models of dynamic heterogeneous networks to capture critical features such as routing, data flow optimization, and network hierarchy detection for the Near Space Network's upcoming real-mission deployment including LunaNet. To better encapsulate the multifaceted nature of space communications, we first explore sheaf constructions on hypergraphs and more accurately model time variation in our network using the theory of topos and moduli spaces. For algorithmic directions, we develop a framework for automated community and bottleneck detection using Ollivier-Ricci curvature and persistence homology, so we can either bypass or exploit the detected bottlenecks using network coding.**

**In establishing solid mathematical frameworks to model space communications, we will be able to better standardize more efficient and scalable network services for the upcoming Near-Space Network and design the architecture of the eventual Solar System Internet. Examples are given in the context of the ISS network experiment with a discussion on how these tools can be used in more general settings. Finally, we conclude with future research directions.**


## TABLE OF CONTENTS



## 1. INTRODUCTION

The recognized need for networked space communications has given rise to multiple projects, most notably Delay Tolerant Networking (DTN). DTN refers to a collection of protocols that work together to provide networking despite the mobility, disruption, latencies, and other difficulties inherent to space systems. The purpose of this paper is not to give an introduction to DTN (see e.g. [1, 2]), but rather to discuss what it means to architect DTN-enabled assets into a *true* network.

Typical DTN demonstrations have featured globally-distributed globally-consistent schedules between all network constituents. This approach enables route computations at any given moment in time there might never be end-to-end paths, but rather paths must be considered temporally. While this methodology does work, it has several limitations. One is that it defeats the real purpose of networking, which is scalability; indeed, updating one node's tables, trajectory, timing, or anything results in a network-wide update. A related limitation, which is perhaps a corollary to the first one, is that there is no addressing in DTNs - rather, there is a (topologically) flat name-space: nodes are identified with integers with no presumed hierarchy.

Focusing on the notion of a network-wide update, we note that updating configurations on a spacecraft can be a weeks-long process or more. Therefore, ad hoc events like ground-station handovers do not factor well into the procedural side of space communications. Recently, a DTN test campaign

was conducted on the International Space Station to see if a DTN could be architected in such a way - despite the addressing limitation - to enable flexible and transparent networking to the end user (in this case, the ISS) [3]). In this experiment, a laser relay satellite was used to connect the ISS to one of three ground stations, each of which had their own DTN nodes. The ISS was given one configuration for the test campaign, which lasted for several months, as a viable means was found to abstract the complexity of the satellite-to-ground communications system - and its inherent pointing decisions - away from the ISS. The architecture used for this test marked the first time DTN was able to contain complexity within the context of space network service providers, and it is this capability that will be necessary for the upcoming lunar network (LunaNet, [4]) and the future Solar System Internet.

The recent strides demark a long-overdue elevation of DTN operationally, but it is known that having multiple space network service providers (as in the case of LunaNet) treads on thin theoretical ice [2, 5–8]. The fundamental issue is that the foundational theory of DTN is under-developed, which these aforementioned papers have only begun to discover. Fortunately, the mathematical advances targeting DTN applications have begun to yield practical capabilities, e.g. time synchronization [9], but well-identified work remains. The ISS experiment campaign included lessons-learned about the value of extending DTN from unicast to broad/multi/anycast, as this was how the configuration issue was solved. Further means of reducing configuration complexity include introducing default routes for edge nodes, akin to terrestrial devices. Going beyond low Earth orbit, we see that different subdomains (e.g. Earth, Lunar, Martian) will have their own characteristics as will any cross-linking between these domains/communities. Progress in these areas can illuminate a path to temporospatial addressing for DTNs, among other benefits. At a high level, the contribution of this work is to both expand the theory and to provide germane examples so the reader might take ownership of these concepts. More concretely, we outline the four-fold focus areas:

**1.** (Section 2) We present the routing sheaves on hypergraphs by writing hypergraphs as posets, where one can define sheaves on by inducing Alexandrov topology [10, 11]. We also present another construction by reducing a hypergraph to a bipartite graph (incidence graph).

**2.** (Sections 3 and 4) We examine network curvature and dimensionality reduction techniques to make computations on network data more tractable and yield actionable insights into network health and design. We introduce a framework for embedding networks into low-dimensional Riemannian manifolds with heterogeneous geometry.

**3.** (Section 5) We show that techniques from persistent homology can be used to help detect poorly connected communities with a network. Additionally, we combine persistent homology and network embedding to incorporate both discrete and continuous perspectives for studying the emergent topology of time-varying networks.

**4.** (Section 6) We initiate the study of applied topos theory for network modeling by building a subobject classifier for topos of graphs. We introduce categories of directed hypergraphs, time-dependent directed hypergraphs, and time-dependent graphs as toposes with examples.

Additional mathematical quick-start material that may be needed or helpful for understanding the results of this paper are provided in the appendix.

- Appendix A provides an introduction to category theory needed to get started with Section 6 (and Appendix B, although it can be understood without it).
- Appendix B outlines the definition of sheaves, as well as simplicial and cell complexes, which are building blocks of cellular sheaves. It is also intended to assist the readers with reading Section 5.

## 2. From Graphs to Hypergraphs

The key to mathematical modeling is constructing a mathematical representation of a system with clarity as to the assumptions that the representation makes. This section considers the challenge of routing in the setting of space communications, in fact, any wireless communications. Traditionally, networks are mathematically modeled as a graph, the data structure that captures pairwise interactions between entities. Hypergraphs offer an alternative scaffolding to more accurately represent various communications structures; for example, graphs do not accurately capture multicast and broadcast in a flexible manner, whereas hypergraphs are proven to be capable of this [12–14]. In prior work the mathematical data structure known as *sheaves* proved useful for modeling general routing on graphs (and in particular, the path sheaf). Here we explore the implications of generalizing from graphs to hypergraphs and extensions of path finding sheaves to the hypergraph setting.

As said, networks are typically modeled mathematically as graphs, where nodes are represented as vertices and an edge between two nodes (vertices) represents the connection between the two. These (simple) graphs can represent interactions as directed or undirected, and weighted or unweighted. Let us define what we mean here.

**Definition 1.** A **directed graph** $G$ is a set of vertices $V$ along with a set of edges $E$, ordered pairs $E \subset V \times V$.

An **undirected** graph is defined as above, except that the edges are unordered vertex pairs instead of ordered vertex pairs. Additionally, we may equip a graph of any type with an assignment of weights to the edges $w : E \to \mathbb{R}^+$, to make it a **weighted** graph.

The vertices in a graph model of communication represent *nodes*, the objects that wish to communicate. Thus, implicit in these constructions, is the assumption that communication is done between pairs of vertices. In our pursuit of robust and scalable networking solutions, we must question what information we lose when modeling space communications with these assumptions.

There are communication protocols, like broadcast and multicast, which are both vital to space communications, and fundamentally not pairwise interactions. In order to better represent these kinds of interactions we must extend the definition of a graph to allow edges between multiple vertices. These generalizations are called hypergraphs.

**Definition 2.** A **hypergraph** $H$ is a set of vertices $V$ along with a set of edges $E \subseteq \mathbb{P}(V)$

The edges of a **directed** hypergraph are ordered pairs $(H, T)$ such that $H, T \in \mathbb{P}(V)$ and $H \cap T = \emptyset$. In other words, directed edges on a hypergraph are partitioned into the 'head' and 'tail' vertices which imply direction across the edge from the tail to the head. One can also assign weights to the edges

of a hypergraph, as described before, to make it a weighted hypergraph.

A hypergraph can be seen as a set system, with $V$ representing the base set of objects, and $E$ being sets of those objects. Viewed in this way, we may naturally build a partially ordered set (poset) on the containment structure of the hypergraph.

**Definition 3.** Let $H = (V, E)$ be a hypergraph. Consider the poset $(X, \leq)$, where $X = V \cup E$ and for $x_1, x_2 \in X$ $x_1 \leq x_2 \iff x_1 \subseteq x_2$. We call this the containment poset of $H$ and denote it $P(H)$.

This containment poset gives structure to the relations between vertices and the edges that contain them, but also the edges contained in each other. We now introduce a sheaf, which is a way of assigning information to this structure (Appendix B presents and explains the axiomatic definition of sheaf).

**Definition 4** (Sheaves on Partial Order [10, 11, 15, 16])**.** A sheaf on a partial ordering $(X, \leq)$ is an assignment of the following.

1. A set $\mathcal{F}(x)$ to each element $x \in X$, called the stalk of $x$.
2. For each $x \leq y \in X$ a map $\mathcal{F}(x \leq y) : \mathcal{F}(x) \to \mathcal{F}(y)$, called the restriction map from $x$ to $y$, such that
3. whenever $x \leq y \leq z$, it follows $\mathcal{F}(x \leq z) = \mathcal{F}(x \leq y) \circ \mathcal{F}(y \leq z)$.

We will be defining sheaves on the containment posets of hypergraphs. For ease of notation we will refer to the elements in this poset and the vertices and edges of the hypergraph interchangeably in our sheaf definitions.

Further, we can consider the following notion of locally consistent information on a sheaf.

**Definition 5** (Section of sheaf [10, 11])**.** Let $\mathcal{F}$ be a sheaf defined on the containment poset of a hypergraph $P(H)$. Let $H \subseteq V \cup E$. A **section** $s$ is a choice of element in the stalk of every $x \in H$ such that $x \leq y$ implies $\mathcal{F}(x \leq y)(s(x)) = s(y)$. We call $s$ a **global section** if $H = V \cup E$.

*Sheaves and network modeling*—The gluability condition of sheaf (see Definition 44 for its formal definition), intuitively, is called the *gluability* condition because, when pieces of *local information* ($f_i$ and $f_j$) are *consistent* in the sense that they are equal in the areas they intersect, then they can be *assembled* together into *global information* ($f$). With this in mind, a sheaf can be interpreted as a mathematical data structure that stores local information (over a topological space) that is subject to certain consistency requirements (across the space), basically *describing* or *informing* us of global property/information. This describes what is 'behind the scenes' of the services provided by the networks today. For example, the shortest path within an autonomous system (AS) or domain is computed using Open Shortest Path Finding (OSPF), which runs Dijkstra's shortest path finding algorithm, and the inter-AS routing is done with Border Gateway Protocol (BGP), essentially 'gluing' the 'local' paths that OSPF computed.

### *Directed Path Sheaf*

We begin with a discussion of our previous work in path sheaves on directed graphs [6, 7, 9]. With constructions similar to this on simple graph topologies, we have been able to formulate Dijkstra's algorithm in sheaf theoretic language, and consider diffusion across the network with sheaf Laplacians. Since we build sheaves on the containment posets of graphs, one might ask if the choice of directed or undirected graph changes this poset. In this case, because the only containments are from vertices to the edges, it does not change the poset. This is not so for directed hypergraphs, as there is no true way to define containment between two directed hyperedges.

We define the directed path sheaf, $\mathcal{P}$ on a directed graph, $G = (V, E)$. We choose one source vertex, $v_S$, and one target vertex, $v_T$.

We start by defining the stalks on the vertices, $v \in V$.

$$\mathcal{P}(v) = \begin{cases} \text{out}(v) & \text{if } v = v_S \\ \text{in}(v) & \text{if } v = v_T \\ (\text{in}(v) \times \text{out}(v)) \cup \{\bot\} & \text{otherwise} \end{cases} \tag{1}$$

where $\text{out}(v)$ is the set of outgoing edges of $v$ in $G$, and $\text{in}(v)$ is the incoming edges. These vertex stalks represent all the choices of out-edge a source vertex can have, as well as all the choices of in-edge a sink vertex can have. Then, for all other vertices, the possible pairings of in and out-edges and a symbol $\bot$ which would indicate the vertex is 'off.'

The edge stalks are much simpler in that they represent edges as either 'on' or 'off' with the symbols $\top$ and $\bot$ respectively. Given an edge $e \in E$, the edge stalk is

$$\mathcal{P}(e) = \{\top, \bot\}. \tag{2}$$

In the containment poset of a simple graph, the only comparable elements are the vertices with the edges that contain them. This means that the restriction maps we need to define are from the vertices to the edges incident to them. This also means that the restriction maps will automatically satisfy requirement 3 of Definition 4. Given a source or sink vertex $v_\circ = v_S, v_T$ incident to edge $e$, meaning that $v_\circ \leq e$, and $e' \in \mathcal{P}(v)$, we define

$$\mathcal{P}(v_\circ \leq e)(e') = \begin{cases} \top & \text{if } e = e' \\ \bot & \text{otherwise} \end{cases} \tag{3}$$

For other vertices, $v$, incident to edges $e$, with $(e', f') \in \mathcal{P}(v)$, we define

$$\mathcal{P}(v \leq e)((e', f')) = \begin{cases} \top & \text{if } e = e' \text{ or } e = f' \\ \bot & \text{otherwise} \end{cases} \tag{4}$$

In Figure 1, we walk through an example global section $s$ on directed graph $G$. In this global section, we see that $s(v_s) = e_2$, indicating a choice of $e_2$, from the stalk of $v_S$, as the out-edge for the source vertex. Following, we may notice that $s(e_2) = \top$, indicating a choice for $e_2$ to be 'on.' These pieces of information are seen to be consistent in the global section $s$ through the fact that $\mathcal{P}(v_S \leq e_2)(s(v_S)) = \top = s(e_2)$. The agreement of these values locally is indicative of a section, as seen in Definition 5. Similarly, we can see that $\mathcal{P}(v_S \leq e_1)(s(v_s)) = \bot = s(e_1)$ indicates agreement with the deactivated incident edge of $v_S$ as well. In this way we can go through the other vertices of the graph and their respective restriction maps. The activated vertex $v_3$ has a stalk assignment which affirms the activation of the incoming

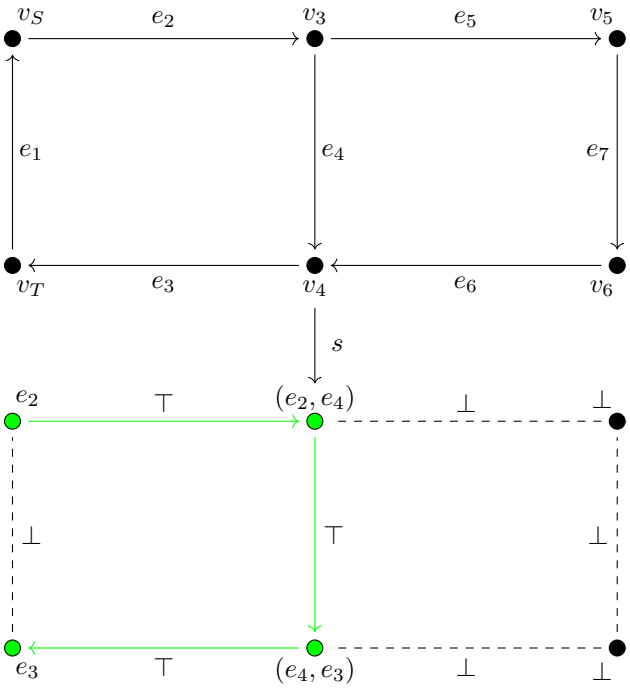



**Figure 1**: A graph $G$ and the values of a particular global section $s$.

edge $e_2$ and outgoing edge $e_4$. Similarly, the stalk assignment on $v_4$ affirms the activation of the incoming $e_4$ and outgoing $e_3$. Finally, the stalk assignment of $e_3$ simply affirms the incoming $e_3$. It goes on like this with the agreement for edges $e_5, e_6$, and $e_7$ to be off along with $v_5$ and $v_6$. Now, when looking at the big picture, we see that this assignment of local information has allowed a global section of consistent information, which outlines a directed path from $v_S$ to $v_T$.

We note that this formulation does allow for degenerate sections which have directed cycles outside of the path as well. We have explored these cases and alternate formulations that deal with these degeneracies in [9].

### *Bipartite Path Sheaf*

Now we can start thinking about path sheaves on hypergraphs. Hypergraphs are undoubtedly a strong generalization of graphs. Perhaps contrary to the resulting intuition, a hypergraph can be represented as a bipartite simple graph. The caveat is that both vertices and edges in the original hypergraph are represented as vertices. Similarly, a directed hypergraph can be represented as a bipartite directed graph.

**Definition 6.** Let $H = (V_H, E_H)$ be a directed hypergraph. The directed bipartite graph associated to $H$ is $G = (V, E)$ where $V = V_H \cup E_H$ and for all $v \in V_H$ and $e \in E_H$, we have $(v, e) \in E \iff v \in \text{tail}(e)$ and $(e, v) \in E \iff v \in \text{head}(e)$.

As an example we can consider the directed hypergraph and its bipartite representation in Figure 2. Note that in this example and in subsequent depictions of hypergraphs, we borrow some illustrative trends from simplicial complexes for ease of notation.

One straightforward way to understand path finding on hypergraphs with sheaves is by simply placing our path sheaf

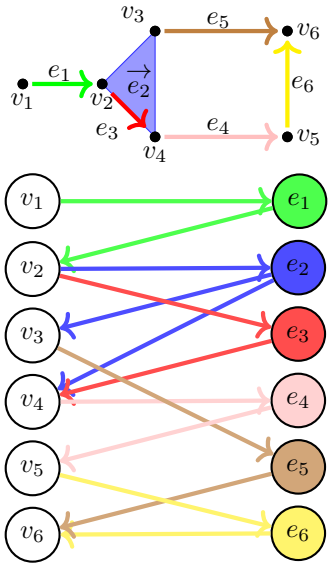



**Figure 2**: An example directed hypergraph (top) and the corresponding directed bipartite graph (bottom).

construction on the bipartite graph associated to a hypergraph. The global sections of this sheaf will correspond to an alternating sequence of hyperedges and vertices which lead from a chosen source vertex to a chosen target vertex. This corresponds with one commonly accepted notion of a hypergraph path, known as an $s-$path [17].

**Definition 7.** Let $H = (V, E)$ be a hypergraph. Consider a sequence of distinct edges, $e_1 \cdots e_k$, in $E$ such that $1 \leq |e_i \cap e_{i+1}| \leq s$ for $i = 2, \ldots, k-1$. Such a sequence of edges is called an $s-$path of length $k$.

The reason we start with this extension is because it gives us a familiar topology on which to put our sheaf. Global sections correspond to paths in the bipartite graph, which correspond to $s-$paths in the hypergraph.

In this formulation of our path finding problem, the length, $k$, of the path in the hypergraph is dictated by the number of edge vertices traversed in the bipartite graph. Meanwhile, the characterization of the $s-$path for a particular $s$ value is variable and not directly shown by the global section on the bipartite counterpart.

This gives us one notion of path finding in a directed hypergraph in a way that extends easily to weighted directed hypergraphs. We now present a slightly modified version of the distance path sheaf we have presented in our previous work [9].

Let $H$ be a hypergraph, and $B$ be the associated bipartite graph. In what follows we construct the hypergraph bipartite distance path sheaf $\mathcal{BDP}$. Given vertex $v \in V(B)$ (meaning that $v \in V(H) \cup E(H)$), we define the stalk

$$\mathcal{BDP}(v) = \begin{cases} \text{Out}(v) \times \{0\} & \text{if } v = v_S \\ \text{In}(v) \times \mathbb{R}^+ & \text{if } v = v_T \\ (\text{In}(v) \times \text{Out}(v) \times \mathbb{R}^+) \cup \{\bot\} & \text{otherwise} \end{cases}$$

For all edges, $e \in E(B)$, the stalk is defined as follows:

$$\mathcal{BDP}(e) = \mathbb{R}^+ \cup \{\bot\}.$$

The restriction maps are defined as follows:

$$\mathcal{BDP}(v_S \leq e)((e_i, 0)) = \begin{cases} w(v_S) & \text{if } e = e_i \\ \bot & \text{otherwise} \end{cases}$$

$$\mathcal{BDP}(v_T \leq e)((e_i, x)) = \begin{cases} x + w(v_T) & \text{if } e = e_i \\ \bot & \text{otherwise} \end{cases}$$

$$\mathcal{BDP}(v \leq e)((e_i, e_o, x)) = \begin{cases} x & \text{if } e = e_i \\ x + w(v) & \text{if } e = e_o \\ \bot & \text{otherwise} \end{cases}$$

Now the global sections will correspond to directed paths from the source to target vertices, along which real values indicate a distance from the source vertex $v_S$ as a sum of edge weights.

Since this bipartite graph is representing a hypergraph, the weight is added from a function on the vertices instead of on the edges. If these weights represent vertex induced delay, then the vertices from the original hypergraph may have time delays associated to processing and rerouting on them as well, and the vertices from the hyperedges of the original graph may have time delays associated to transmission time on them. As a consequence of this construction, with the weight added at the vertices, an edge weight from the hypergraph is considered constant no matter how the edge is routed through. In the broadcast or multicast modeling setting, this is equivalent to the assumption that a transmission delay from a (multi)broadcast is the same across all the corresponding receivers.

The convenience of the bipartite directed formulation is that it splits the hyperedge information up. In bipartite directed edges we have the directed information, and in the vertex weights, we have the weight information of the hyperedge.

***Undirected Hypergraph Path Sheaf***

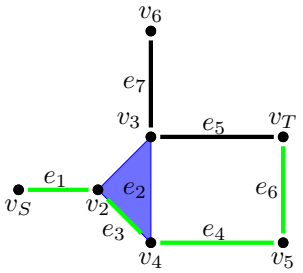
While a path sheaf on the bipartite graph does give us one notion of a path on a hypergraph, there are other types of path like structures we can explore for the purposes of routing. Additionally, the bipartite path sheaf does not really give us insight into the topology of the hypergraph we are on, just a way to traverse it. It treats hyperedges as though they are pairwise traversible.

This interpretation can be useful when traversing a network with mulitcast and broadcast relations. However, we do not necessarily benefit or gain as much insight from the more general hypergraph description if we treat its edges as pairwise traversible, through $s-$paths anyway. Because of these questions, We explore a path sheaf on the containment topology of the hypergraph itself, rather than that of the bipartite representation thereof.

In what follows we will define the undirected hypergraph path sheaf $\mathcal{UHP}$, which admits undirected global sections corresponding to paths on any simple subgraph of the hypergraph, as well as more general path like structures on the hypergraph as a whole.

Let $\mathsf{Ne}(x)$ be the set of edges incident to $x$ for $x \in V \cup E$. For $v_\circ = v_S$ or $v_T$ we define the stalks:

$$\mathcal{UHP}(v_\circ) = \mathbb{P}(\mathsf{Ne}(v)).$$




| Section $s_1$ on Vertices | Section $s_1$ on Edges |
|---|---|
| $v_S \to \{e_1\}$ | $e_1 \to \{e_1\}$ |
| $v_2 \to \{e_1, e_3\}$ | $e_2 \to \{\bot\}$ |
| $v_3 \to \{\bot\}$ | $e_3 \to \{e_3\}$ |
| $v_4 \to \{e_3, e_4\}$ | $e_4 \to \{e_4\}$ |
| $v_5 \to \{e_4, e_6\}$ | $e_5 \to \{\bot\}$ |
| $v_6 \to \{\bot\}$ | $e_6 \to \{e_6\}$ |
| $v_T \to \{e_6\}$ | $e_7 \to \{\bot\}$ |

**Figure 3**: A global section, $s_1$, of $\mathcal{UHP}$ on a hypergraph. This example shows how a regular path on the hypergraph can be expressed as a global section of this sheaf.

For all other vertices, $v$, and edges $e$, we define the following stalks:

$$\mathcal{UHP}(v) = (\mathbb{P}(\mathsf{Ne}(v)) \setminus \mathsf{Ne}(v)) \cup \{\bot\}$$
$$\mathcal{UHP}(e) = \mathsf{Ne}(e) \cup \{\bot\}.$$

In words, the stalk of a vertex is $\bot$ together with the set of collections of incident edges, minus the sets containing only one element. The stalk of an edge is the set of incident edges and $\bot$.

Now we move on to the restriction maps. For a vertex $v$ and edge $e$ such that $v \in e$, we define the following restriction map:

$$\mathcal{UHP}(v \leq e)(F) = \begin{cases} f' & e \subseteq f' \text{ for } f' \in F \\ \bot & \text{otherwise} \end{cases}$$

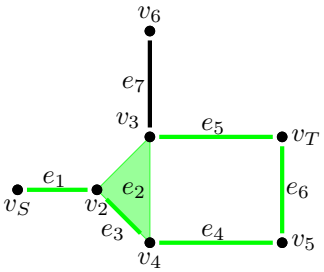
One key difference from our previous sheaves on regular graphs, is that the containment poset of a hypergraph displays containment between edges. That is, if $e_1 = \{v_1, v_2, v_3\}$ and $e_2 = \{v_1, v_2\}$, then $e_2 \leq e_1$. Thus in order for $\mathcal{UHP}$ to be a sheaf by Definition 4, we must define inter edge restriction maps. Thus given $e_1, e_2 \in E$ such that $e_1 \leq e_2$, let

$$\mathcal{UHP}(e_1 \leq e_2)(f) = \begin{cases} f & \text{if } e_2 \subseteq f \\ \bot & \text{otherwise} \end{cases}$$

The increased complexity of containment allows this sheaf to admit a more diverse space of global sections.

In Figure 3, we exemplify a familiar form of path, now on a hypergraph. Notice that the $s_1$ section chooses one edge for the source ($v_S$) and target ($v_T$) nodes, and pairs of edges for all others. This allows for a similar path structure to what we saw in the path sheaf before. Moreover, every enabled edge is a simple edge between two vertices. This generalizes in that a $s-$path in a section over $\mathcal{UHP}$ can only occur exclusively activating edges of size 2.

In Figure 4, we show what a global section, including a proper hyperedge, can look like. Notice that the section choices




| Section $s_2$ on Vertices | Section $s_2$ on Edges |
|---|---|
| $v_S \to \{e_1\}$ | $e_1 \to \{e_1\}$ |
| $v_2 \to \{e_1, e_2\}$ | $e_2 \to \{e_2\}$ |
| $v_3 \to \{e_2, e_5\}$ | $e_3 \to \{e_2\}$ |
| $v_4 \to \{e_2, e_4\}$ | $e_4 \to \{e_4\}$ |
| $v_5 \to \{e_4, e_6\}$ | $e_5 \to \{e_5\}$ |
| $v_6 \to \{\bot\}$ | $e_6 \to \{e_6\}$ |
| $v_T \to \{e_5, e_6\}$ | $e_7 \to \{\bot\}$ |

**Figure 4**: A global section, $s_2$, of $\mathcal{UHP}$ on a hypergraph. This example shows a more general path structure that emerges from the hypergraph topology.

are still between pairs of vertices, but now the activation of $e_2$ has required that $v_3$ also make a routing decision. Additionally, we can see that $s_2(e_2) = s_2(e_3) = \{e_2\}$. This allows agreement with the vertex restriction maps $\mathcal{UHP}(v_i \leq e_2)(s_2(v_i)) = \{e_2\} = s_2(e_2)$ and $\mathcal{UHP}(v_i \leq e_3)(s_2(v_i)) = \{e_2\} = s_2(e_3)$ for $i = 2, 3, 4$, as well as in the inter edge restriction map $\mathcal{UHP}(e_3 \leq e_2)(s_2(e_3)) = \{e_2\} = s_2(e_2)$. This shows consistency with Definition 5.

Here, we have seen that hyperedges with more than two vertices split the path in a global section. The key is that in order for this to be a section, the split paths must converge again at or before the target vertex. Thus instead of a regular path, we get a path that branches off at $e_2$ only for the branches to meet again at $v_T$. While these sections aren't as simple as the $s-$paths given by the bipartite graph, they are more true to the topology of the network, and could give insight into redundant routing schemes, and what robust network structure looks like.

To exemplify what we mean by being true to the topology of the network, we can briefly consider the hypergraph above without the edges $e_3$ and $e_5$. This hypergraph would still be connected in that there will exist $s-$paths between any two vertices, however, it will not have any global sections as defined, with source vertex $v_S$ and target vertex $v_T$. So while a chain of causality could reach $v_T$ from $v_S$, this sheaf would suggest that such a $s-$path would be using the edge $e_3$ without it being stated.

### *Directed Hypergraph Path Sheaf*

The undirected hypergraph path sheaf provides us with intuition on the structure of path like global sections on a hypergraph. We now explore how a hypergraph sheaf could capture a notion of direction. A structure which guarantees that every vertex in a global section knows in which direction the target is.

We define some sets for ease of notation. For all edges $e \in E$ we define the set of its directed edge partitions.

$$DE(e) = \{(H, T) \mid H \subseteq e \text{ and } V = e \setminus U\}$$

In words this is, given an edge, the set of all its partitions into a of head and tail vertices. Thus one element of this set descibes one scenario of flow across the edge.

For all vertices $v \in V$ we define two sets. These are sets of directed edge partitions on subsets of edges incident to $v$. The set of such partitions with $v$ in the head will be

$$DH(v) = \{(H, T) \in DE(e) | v \in H \text{ for some } e \in Ne(v)\}$$

and the set of such partitions with $v$ in the tail will be

$$DT(v) = \{(H, T) \in DE(e) | v \notin H \text{ for some } e \in Ne(v)\}$$

Now let's define the direct simplicial path sheaf, $\mathcal{DHP}$. Then with a choice of $v_S$, and $v_T$, we define the vertex stalks as follows.

$$\mathcal{DHP}(v) = \begin{cases} \mathbb{Pm}(DT(v)) & v = v_S \\ \mathbb{Pm}(DH(v)) & v = v_T \\ (\mathbb{Pm}(DH(v)) \times \mathbb{Pm}(DT(v))) \cup \{\bot\} & \text{otherwise} \end{cases}$$

Here let $\mathbb{Pm}(DT(v)) \subseteq \mathbb{P}(DT(v))$ and $\mathbb{Pm}(DH(v)) \subseteq \mathbb{P}(DH(v))$ denote the elements of the powerset of directed edge partitions whose elements have only $v$ in common. Thus a choice in the stalk of the source vertex corresponds to the directed partitions of some number of non overlapping incident edges, such that the source vertex is in the tail. A similar story with the target vertex, except that its stalk requires it to be in the head. Then, all other vertices have a stalk containing the product of these sets of directed edge partitions, along with the off symbol $\bot$.

Next, we define the stalks for all edges $e \in E$:

$$\mathcal{DHP}(e) = \{L \in DE(f) \mid \text{For } f \in \mathsf{Ne}(e) \text{ where } e \subseteq f\} \cup \{\bot\}.$$

This stalk is the set of directed partitions of edges that contain $e$ together with $\bot$.

We note an abuse of notation in what follows. Elements of the sets $DH(v), DT(v)$ and $DE(e)$ are ordered pairs of sets of vertices. For an element of one of these sets, $Y = (H, T)$ and an edge $f \in E$, we write $f \subseteq Y$ as a shorthand for $f \subseteq H \cup T$.

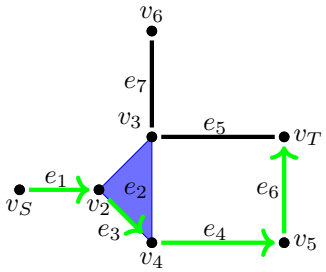
We proceed to defining the restriction maps. For the source and target vertices restricted onto any edge containing them we have the following.

$$\mathcal{DHP}(v_S \leq e)(O) = \begin{cases} O_i & e \subseteq O_i \text{ for some } O_i \in O \\ \bot & \text{otherwise} \end{cases}$$

Note that the choice of vertex stalks prohibits $e \subseteq O_i$ for more than one $i$, and the same is true for the next one:

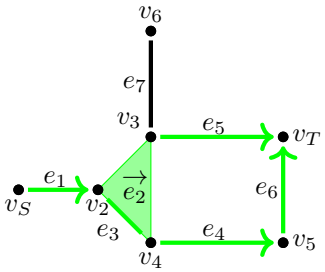
$$\mathcal{DHP}(v_T \leq e)(I) = \begin{cases} I_i & e \subseteq I_i \text{ for some } I_i \in I \\ \bot & \text{otherwise} \end{cases}$$

The restriction maps from the source vertex $v_S$ map to the directed partition of the superedge of $e$ if it exists in $O$. If $e$ is contained in no edge of $O$, then this maps to $\bot$. Similarly with restriction maps of $v_T$.




| Section $f_1$ on Vertices | Section $f_1$ on Edges |
|---|---|
| $v_S \to \{(v_S, v_2)\}$ | $e_1 \to \{(v_S, v_2)\}$ |
| $v_2 \to \{((v_S, v_2), (v_2, v_4))\}$ | $e_2 \to \{\perp\}$ |
| $v_3 \to \{\perp\}$ | $e_3 \to \{(v_2, v_4)\}$ |
| $v_4 \to \{((v_2, v_4), (v_4, v_5))\}$ | $e_4 \to \{(v_4, v_5)\}$ |
| $v_5 \to \{((v_4, v_5), (v_5, v_T))\}$ | $e_5 \to \{\perp\}$ |
| $v_6 \to \{\perp\}$ | $e_6 \to \{(v_5, v_T)\}$ |
| $v_T \to \{(v_5, v_T)\}$ | $e_7 \to \{\perp\}$ |

**Figure 5**: A global section, $f_1$, of $\mathcal{DHP}$ on a hypergraph. This example shows how a regular directed path on the hypergraph can be expressed as a global section of this sheaf.




| Section $f_2$ on Vertices | Section $f_2$ on Edges |
|---|---|
| $v_S \to \{(v_S, v_2)\}$ | $e_1 \to \{(v_S, v_2)\}$ |
| $v_2 \to \{((v_S, v_2), (v_2, \{v_3, v_4\}))\}$ | $e_2 \to \{(v_2, \{v_3, v_4\})\}$ |
| $v_3 \to \{((v_2, \{v_3, v_4\}), (v_3, v_T)\}$ | $e_3 \to \{(v_2, \{v_3, v_4\})\}$ |
| $v_4 \to \{((v_2, v_4), (v_4, v_5))\}$ | $e_4 \to \{(v_4, v_5)\}$ |
| $v_5 \to \{((v_4, v_5), (v_5, v_T))\}$ | $e_5 \to \{(v_3, v_T)\}$ |
| $v_6 \to \{\perp\}$ | $e_6 \to \{(v_5, v_T)\}$ |
| $v_T \to \{(v_5, v_T), (v_3, v_T)\}$ | $e_7 \to \{\perp\}$ |

**Figure 6**: A global section, $f_2$, of $\mathcal{DHP}$ on a hypergraph. This example shows a more general global section which can occur with this sheaf.

This same idea is used in the definition of restriction maps for the other vertices:

$$\mathcal{DHP}(v \le e)((I,O)) = \begin{cases} O_i & e \subseteq O_i \text{ for some } O_i \in O \\ I_i & e \subseteq I_i \text{ for some } I_i \in I \\ \perp & \text{otherwise} \end{cases}$$

Finally, for consistency we must define the inter edge restriction maps:

$$\mathcal{DHP}(e_1 \le e_2)(F) = \begin{cases} F & e_2 \subseteq F \\ \perp & \text{otherwise} \end{cases}$$

This simply reaffirms the containment assignment made by the vertices.

In Figures 5 and 6 we see two example global sections of the $\mathcal{DHP}$ sheaf on a hypergraph.

In Figure 5 we see how we can recover the notion of a directed path on a hypergraph, using the same undirected hypergraph containment poset as we used for $\mathcal{UHP}$. The direction is placed on the topology by the sheaf. This section will look similar to that of Figure 3, only instead of edges themselves, the section choices are directed partitions on the incident edges. Vertices $v_2$, $v_4$, and $v_5$ each have a designated incoming edge as well as outgoing edge.

If we look at Figure 6, we can see the direction giving more structure to the branched section we saw in Figure 4. In the global sections of this sheaf there is an indication of the flow from the source to the target vertex. Particularly, notice that $f_2(e_2) = (v_2, \{v_3, v_4\})$, assigns to $e_2$ the direction of transport from vertex $v_2$ to the vertices $v_3$ and $v_4$. Moreover, we can see that $e_3$ which is contained in $e_2$ has $f_2(e_3) = (v_2, \{v_3, v_4\})$ as well. Thus in this global section, the assignment on $e_3$ indicates the flow to vertices $v_3$ and $v_4$, despite $e_3$ having no direct connection to $v_4$.

## 3. Network Curvature

Graphs as mathematical objects are algebraic and combinatorial in nature (read: algorithmic and computational), but they are also geometric. Notions of curvature have been extended to the discrete graph setting as readily computable measurements that yield readily interpretable insight into graphs both locally and globally. Applied to networks, we can detect communities and bottlenecks, information that can help with network planning, troubleshooting, and architecture.

### *Ollivier-Ricci curvature on networks*

In the case of networks, we can define Ricci curvature by placing (discretely parameterized) probability measures on node neighborhoods and computing optimal transport plans between them.

**Definition 8.** A ***probability measure*** over a vertex set $V$ is a function $m : V \to [0,1]$ satisfying $\sum_{x \in V} m(x) = 1$.

Given a graph $G = (V, E)$, let $N(x) = \{y \mid (x, y) \in E\}$ be the neighborhood of a vertex $x \in V$. Let $\deg(x) = |N(x)|$ be the degree of $x$. We can define a *general family of probability distributions* on the graph, with two parameters $\alpha \in [0,1]$ and $p \ge 0$:

$$m_x^{\alpha,p}(y) = \begin{cases} \alpha & \text{if } y = x \\ \frac{1-\alpha}{C} \cdot \exp(-d(x,y)^p) & \text{if } y \in N(x) \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

Here $C = \sum_{x_i \in \pi(x)} \exp(-d(x, x_i)^p)$ is a normalization factor. The distance metric $d(x, y)$ is the length of the shortest path between $x$ and $y$ on the graph. The power parameter $p$ determines how much we want to discount the neighbor $x_i$ of $x$ with respect to the weight $d(x, x_i)$. The larger $p$ is, the less significant the distant neighbors of $x$ become. The Wasserstein distance $W\left(m_x^\alpha, m_y^\alpha\right)$ is defined as the best way to transport the distribution $m_x^\alpha$ to $m_y^\alpha$.

**Definition 9.** The ***Wasserstein distance*** $W\left(m_x^\alpha, m_y^\alpha\right)$ is the optimal transport distance between mass distributions $m_x^\alpha$ and $m_y^\alpha$, defined by:

$$W\left(m_x^\alpha, m_y^\alpha\right) = \inf_M \sum_{x_i, y_j \in V} d(x_i, y_j) M(x_i, y_j), \quad (6)$$

where $\sum_j M(x_i, y_j) = m_x^\alpha(x_i)$ and $\sum_i M(x_i, y_j) = m_y^\alpha(y_j)$.

Here $M(x_i, y_j)$ is a coupling that measures the amount of mass moved from $x_i$ to $y_j$ along the shortest path; it is also the best transportation plan (the one which minimizes the total transport distance).

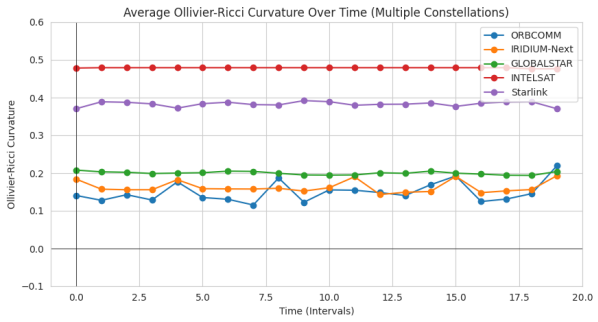
**Definition 10.** The discrete ***Ollivier-Ricci curvature*** $\kappa_\alpha(x, y)$ of a pair of nodes $x$ and $y$ is defined by the comparison of the Wasserstein distance to the shortest-path distance between the nodes:

$$\kappa_\alpha(x, y) = 1 - \frac{W(m_x^\alpha, m_y^\alpha)}{d(x, y)}. \tag{7}$$

We can define the Ollivier-Ricci curvature of a node to be the average of its incident edge curvatures:

$$\kappa(v_i) = \frac{1}{\deg(v_i)} \sum_{\{v_i, v_j\} \in E} \kappa(v_i, v_j). \tag{8}$$

**Figure 7**: Plot of average Ollivier-Ricci curvature over time for multiple satellite constellation networks.




### *Discrete Ricci Flow on Networks*

On manifolds, Ricci flow is a process that deforms the metric while the Ricci curvature evolves to be uniform everywhere. In his paper [18], Ollivier suggested that people should study the discrete Ricci flow in a metric space $(X, d)$ by evolving the distance $d(x, y)$ on $X$ according to the Ricci curvature $\kappa(x, y)$ between two points $x, y \in X$:

$$\frac{d}{dt} d_t(x, y) = -\kappa_t(x, y) d_t(x, y). \tag{9}$$

The discrete Ricci flow algorithm on a network is an evolving process similar to this.

**Definition 11** (Discrete Ricci Flow)**.** For any pair of adjacent nodes $x$ and $y$ on a graph $G = (V, E)$, ***discrete Ricci flow*** iteratively adjusts each edge weight $w(x, y)$ by the Ollivier-Ricci edge curvature $\kappa(x, y)$:

$$w^{(i+1)}(x, y) = d^{(i)}(x, y) - \kappa^{(i)}(x, y) \cdot d^{(i)}(x, y), \tag{10}$$

where $w^{(i)}(x, y)$ is the weight of the edge $xy$ at the $i$-th iteration, $\kappa^{(i)}(x, y)$ is the curvature of the edge $xy$ at the $i$-th iteration, and $d^{(i)}(x, y)$ is the shortest path distance on the graph induced by the weights $w^{(i)}(x, y)$. Initially, we set $w^{(0)}(x, y) = w(x, y)$ and $d^{(0)}(xy) = d(xy)$.

From Equations (9) and (10), we see that negative edge curvature $\kappa^{(i)}(x, y)$ induces an increase in both the distance and edge weight between nodes $x$ and $y$. This means that negatively curved edges expand, while positively curved edges shrink. Thus, we observe that intra-community edge curvatures converge to zero, while inter-community edge curvatures tend to infinity. After many iterations of Ricci flow, we may remove the most negatively curved edges to form a partitioning of the network into communities. Note that in each iteration we must update the edge weights in the network, so we must re-compute the shortest-path distances induced by the new weights, as well as the Ollivier-Ricci curvatures of the edges.

### *Community Detection with Discrete Ricci Flow*

In order to detect subnetworks with different curvatures within a network, we can use a Ricci flow-based method for community detection [19]. The algorithm is inspired by the observation that communities in networks resemble regions in Riemannian manifolds of positive curvature. As shown in [19], the discrete Ricci flow process simultaneously expands negatively curved edges and contracts positively curved edges. Over time, the network separates into communities of positive curvatures, with a few negatively curved edges connecting them. In effect, this allows us to more easily distinguish between clusters of nodes that are more densely connected. This is useful since we would like to partition the network into smaller subnetworks with varying curvatures. To do this, we can compute the average Ollivier-Ricci curvature of edges within each subnetwork to determine which model spaces to use in the product manifold.

## 4. Graph Embeddings into Riemannian Manifolds

As mentioned, the key motivator for networking is to achieve scalability - but of course this comes at a cost: complexity. At a glance, every node has a collection of neighbors, and hence choices in physical connections, protocols, policies, etc. As such, networks can appear to be very high-dimensional objects with an associated high overhead for any computation one might need. In traditional networking many of these details are naturally abstracted away, leaving exactly the necessary information (e.g. for routing). While it is not yet clear how to best identify excess data in our temporospatial networks, we can simplify matters by choosing better representations of these data. This motivates the notion of *graph embeddings*, where we take the geometrical graph object and find rigorous ways to embed it simplifying spaces.

While Euclidean spaces have long been the epicenter of network embedding methods, in recent years it has been shown that hyperbolic spaces and more general manifolds are advantageous in reflecting the power-law degree distributions and hierarchical structures of real-world networks. These spaces include matrix manifolds [20], elliptical spaces [21], rotationally-symmetric manifolds [22], and products of constant-curvature Riemannian manifolds [23]. Graph embeddings into Riemannian manifolds allow for the modeling of complex, heterogeneous network structures by embedding nodes into low-dimensional spaces with varying curvatures. This allows for better preservation of pairwise distances and curvature variations, which are critical for tasks such as routing, link prediction, and community detection.

### *Network Embeddings*

Often described in the context of dimensionality reduction, network embedding refers to the task of finding a low-dimensional representation of a graph in a vector space. Embedding graphs into a continuous space is at the heart of network representation learning, where the quality of embeddings crucially relies on the similarity between the geometry of the space and that of the network. One reason for doing so, among many others, is to reduce the computational complexity of downstream machine learning algorithms on graphs. It also allows us to approximate the network's geometry and topology in a more tractable space with additional mathematical structure.

**Definition 12.** Let $G = (V, E)$ be a graph. A ***network embedding*** is a map $\phi : V \to \mathbb{R}^n$ that assigns each node $v \in V$ to a corresponding vector in Euclidean space.

In other words, it is a low-dimensional representation of a graph in a vector space (that hopefully preserves its topology). As each node is represented by a vector containing relevant structural information, many combinatorial problems on graphs can be solved by computing distance metrics or operations on the embedding vectors to avoid high complexity. We refer the readers to [24] for an extensive survey of dynamic graph embeddings. In general, an ideal network embedding is an isometric embedding that preserves the distances between points exactly.

**Definition 13.** For metric spaces $(X, d_X)$ and $(Y, d_Y)$, an ***isometric embedding*** of $X$ into $Y$ is an injective map $f : X \to Y$ such that for every two points $x_1, x_2 \in X$,

$$d_Y(f(x_1), f(x_2)) = d_X(x_1, x_2).$$

Extending this definition to network embeddings in Riemannian manifolds, we wish to 'match' geodesic distances between embedded points on the manifold and shortest-path distances between nodes in the graph. Note that, in general, graph cannot always be isometrically embedded into a fixed space; the structure and the dimension of the space significantly affect the distortion [22]. Typically, increasing the dimension allows us to minimize the distortion, however, it is computationally costly and memory-inefficient. For this reason, we seek a lower-dimensional space with 'richer' structure. In Riemannian manifolds, this additional structure is manifested in their curvature and heterogeneous geometry.

### *Mixed-Curvature Network Embeddings*

In order to capture the heterogeneous geometry of a network, we require an embedding space with *non-uniform* curvature. We can solve this problem by learning embeddings of a network in a product manifolds in which each component has constant curvature, but the overall product space has *non-constant* curvature. As suggested in [23], we will consider embeddings into (Cartesian) products of the three component spaces: the $d$-dimensional sphere, $d$-dimensional Euclidean space, and the upper sheet of the $d$-dimensional hyperboloid. All of these Riemannian manifolds have analytic expressions for their exponential maps and gradient vectors, which make them particularly suitable for optimization.

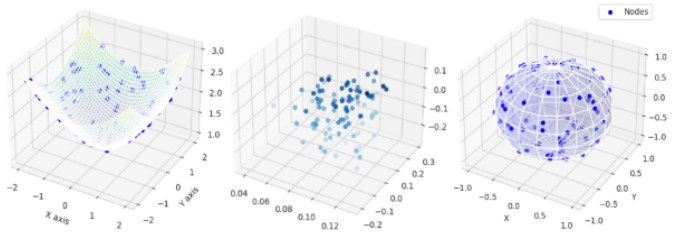
For a sequence of Riemannian manifolds $M_1, M_2, \ldots, M_k$, the *product manifold* refers to the Cartesian product $M = M_1 \times M_2 \times \ldots \times M_k$. If each component $M_i$ is equipped with a Riemannian metric $g_i$, then the product $M$ is also a Riemannian manifold with the metric $g(u, v) = \sum_{i=1}^{k} g_i(u_i, v_i)$. For our purposes, we now provide a more explicit definition of the product manifold suitable for mixed-curvature network embeddings.

**Definition 14** (The Product Manifold)**.** Let $\mathbb{H}^d$, $\mathbb{E}^d$, $\mathbb{S}^d$ denote the hyperbolic, Euclidean, and spherical model spaces of dimension $d$ and curvatures $-1$, $0$, and $1$, respectively. For numbers of copies of each space $h, e, s$, the ***product manifold*** is the Cartesian product space:

$$\mathcal{P} := (\mathbb{H}^d)^h \times (\mathbb{E}^d)^e \times (\mathbb{S}^d)^s \tag{11}$$

In the product space, we express the points $p \in M$ by their coordinates $p = (p_1, \ldots, p_k) : p_i \in M_i$, and similarly a tangent vector $v \in T_pM$ by $(v_1, \ldots, v_k) : v_i \in T_{p_i}M_i$. That is, the metric on the product manifold decomposes into a sum of constituent metrics on each component.

**Figure 8**: Visualization of a Starlink network embedding in product space $\mathbb{H}^2 \times \mathbb{E}^3 \times \mathbb{S}^2$.




### *Embedding and Optimization*

Our goal is to preserve the network structure given through its node-to-node shortest paths by minimizing a loss which encourages similar *relative* geodesic distances between embedded nodes. Let $G = (V, E)$ be a network with $n$ nodes and suppose $\phi\colon V \to \mathcal{P}$ is a node embedding map into a product manifold. Let $\{X_i \in V\}_{i=1}^n$ be the set of nodes and $\{x_i \in \mathcal{P}\colon \phi(X_i) = x_i\}_{i=1}^n$ be the corresponding embedded points in the product manifold. A *perfect* embedding would satisfy

$$d_{\mathcal{P}}(x_i, x_j) = \alpha \cdot d_G(X_i, X_j), \quad \text{for } \alpha > 0$$

for all $i, j \in [n]$. Ideally, we want $\alpha = 1$ so that geodesic distances on the manifold are *exactly* equal to the shortest-path distances on the graph. Thus, as done in [23], to compute embeddings in the product manifold we optimize the placement of points through the loss function

$$\mathcal{L}(x) = \sum_{i<j} \left| \left( \frac{d_{\mathcal{P}}(x_i, x_j)}{d_G(X_i, X_j)} \right)^2 - 1 \right|, \tag{12}$$

for graph distances $\{d_G(X_i, X_j)\}_{ij}$ and embedded points $x = \{x_1, ..., x_n\}$. Now we wish to minimize this loss, so the final embedding map $\phi : V \to \mathcal{P}$ is given by

$$\phi(X) = \arg\min_{x \subset \mathcal{P}} \mathcal{L}(x). \tag{13}$$

Optimization on the product manifold requires a notion of taking a 'step'. This step can be performed in the tangent space and transferred to the manifold via the exponential map $\mathrm{Exp}_p : T_p\mathcal{P} \to \mathcal{P}$. For $t \in \mathbb{R}$, the curves traced

out by $\text{Exp}_p(tv)$ are the geodesics of the product manifold. In a product manifold $\mathcal{P}$, the exponential map and squared distances between points simply decompose into each component simultaneously, making them suitable for use in optimization algorithms like Riemannian stochastic gradient descent (R-SGD). The main difference between gradient-based optimization of smooth functions $f$ on Euclidean versus Riemannian manifolds $(\mathcal{M}, g)$ is that for the latter, the isomorphism between the manifold $\mathcal{M}$ and its tangent space $T_p\mathcal{M}$ at a point $p \in \mathcal{M}$ no longer holds in general. In particular, the stochastic gradient descent update step

$$p' \leftarrow p - \eta \nabla f|_p$$

for learning rate $\eta$ and Euclidean gradient $\nabla f$, is generalized in two areas [25]. First, $\nabla f$ is converted to the Riemannian gradient $\text{grad}\, f$ by multiplying by the inverse of the Riemannian metric $g^{-1} : T_p^*\mathcal{M} \to T_p\mathcal{M}$:

$$\nabla f \to \text{grad}\, f := g^{-1} df,$$

where $df$ is the differential one-form. Second, the exponential map $\exp_p : T_p\mathcal{M} \to \mathcal{M}$ generalizes the vector space addition in the update equation. Note that for spherical and hyperbolic models (which have smaller dimension than the ambient space), we must project the Riemannian gradient onto their tangent spaces.

### *Measuring the Embedding Quality*

There are many ways to measure the quality of an embedding, depending on the given task at hand (i.e. node classification, community detection, link prediction). The quality of an embedding can be determined with various *fidelity metrics*. Here, we consider the metrics of *average distortion* and *mean average precision*. Average distortion measures how well an embedding preserves node distances, whereas mean average precision measures how well an embedding is able to reproduce the 1-hop neighbourhood of a node (disregarding the actual scale of distances).

In the following definitions, consider an undirected graph $G = (V, E)$ with $n$ nodes, and a node embedding $\phi : V \to X$ into a metric space $(X, d_X)$. For each node $a \in V$, let $\mathcal{N}_a$ denote the *neighborhood* of $a$, and let $\deg(a) = |\mathcal{N}_a|$ be the *degree* of $a$.

**Definition 15** (Average distortion)**.** The ***average distortion*** of the embedding $\phi$ (over all pairs of nodes $a, b$) is given by

$$D_{\text{avg}}(\phi) := \frac{2}{n(n-1)} \sum_{a,b \in V} \frac{(d_X(\phi(a), \phi(b)) - d_G(a, b))}{d_G(a, b)}, \tag{14}$$

where $d_G$ is the shortest-path distance on the graph and $d_X$ is the metric on $X$.

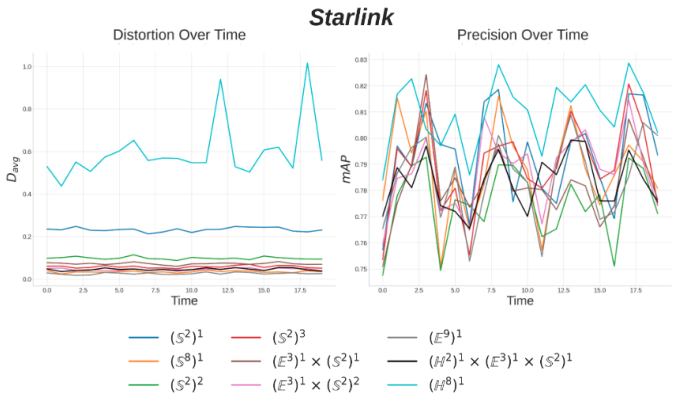
Distortion ($D_{avg}$) is a *global* metric; it considers the explicit value of all distances. At the other end of the spectrum of fidelity measures is the Mean Average Precision ($mAP$):

**Definition 16** (Mean average precision)**.** Let $\mathcal{N}_a = \{b_i\}_{i=1}^{\deg(a)}$ be the neighborhood of node $a \in V$. Define $R_{a,b_i}$ to be the set of nodes $c \in V$ such that $d_X(\phi(a), \phi(c)) \leq d_X(\phi(a), \phi(b_i))$. Then, the ***mean average precision*** is computed as

$$mAP(\phi) := \frac{1}{n} \sum_{a \in V} \frac{1}{\deg(a)} \sum_{i=1}^{|\mathcal{N}_a|} \frac{|\mathcal{N}_a \cap R_{a,b_i}|}{|R_{a,b_i}|}. \tag{15}$$

Observe that $mAP$ does not account for explicit distances; it is a ranking-based measure for reconstructing *local* neighborhoods. We note that $mAP \leq 1$ (higher is better) while $D_{\text{avg}} \geq 0$ (lower is better). In general, distortion is inevitable and tends to accumulate in representations of higher-order networks [22]. This makes it desirable to use other types of information than just pairwise distances when computing embeddings, such as discrete graph curvature [18, 26].

**Figure 9**: Distortion and mAP metrics for Starlink network embeddings in different product spaces over time.




### *Choosing the Optimal Product Space*

Selecting the appropriate curvatures, dimensions, and numbers of components in the product space is a challenging task. We can start by matching the geometry of each component in the product space with the average curvature of its corresponding community. To do this, we can first find the communities with the discrete Ricci flow process (as described in Section 3). Then, based on the average of the Ollivier-Ricci edge curvatures within each community, we can construct component spaces with corresponding Ricci curvatures and form the appropriate product manifold. In theory, the dimension, curvature, and number of copies of each component (hyperbolic, Euclidean, spherical) should have optimal values given the task at hand. In Figure 9, we see that the distortion of the Starlink network embedding is the highest in $\mathbb{H}^8$, slightly lower in $\mathbb{S}^2$, and the lowest in either $\mathbb{S}^8$, $\mathbb{E}^9$, or even $\mathbb{H}^2 \times \mathbb{E}^3 \times \mathbb{S}^2$. As Starlink is globally generally positively curved, it makes sense that a $2-$dimensional sphere preserves its topology better than an $8-$dimensional hyperboloid. However, it seems $\mathbb{H}^8$ has much higher overall precision ($mAP$ score) than the rest.

## 5. Persistent Homology of Networks

Persistent homology (PH) is a powerful tool in topological data analysis (TDA) for examining the topological features of a metric space across varying spatial resolutions. In other words, it is used to measure (and in fact detect) the *shape* of a data set across multiple dimensions. When applied to networks (which can be viewed as finite metric spaces), PH captures the evolution of topological characteristics such as connected components, loops, and voids, as a network is filtered through varying scales. This approach allows for the analysis of network structures beyond traditional graph-theoretic metrics like degree distributions or clustering coefficients.

In this work, we explore the persistent homology of network embeddings, where networks are embedded into low-dimensional manifolds that preserve geometric features such as curvature, rather than analyzing the homology of the networks themselves. Unlike traditional network analysis, where a network is treated as a finite metric space with the shortest-path distance defining the topology, network embeddings introduce a continuous structure that allows for the application of differential geometric techniques and provides a richer set of homological features. This distinction enables the study of multiscale geometric properties that are otherwise obscured in the discrete shortest-path metric, offering new insights into the global shape and connectivity of network data.

Our goal is to understand the topological structure of point clouds (network embeddings) in metric spaces. For this, we can use *homology*, which associates a vector space $H_p(X)$ to a metric space $X$ and a dimension $p$. The dimension of $H_p(X)$ gives us the number of $p$-dimensional holes in $X$. An important feature of these vector spaces is that they are robust to continuous deformations of the underlying metric space (i.e. *homotopy invariant*). However, computing the homology of an arbitrary metric space is extremely difficult. Thus, it necessary to approximate it with a structure that is both combinatorial and topological in nature. This structure is called a *simplicial complex*, which is a higher-dimensional generalization of a graph. The building blocks of this representation are *simplices*, which are the convex hulls of sets of points.

**Definition 17** (Simplicial complex)**.** A ***simplicial complex*** is a finite collection of simplices $K = \{\sigma_1, ..., \sigma_n\}$ such that:

- Every face of a simplex in $K$ also belongs to $K$.
- For any two simplices $\sigma_1, \sigma_2 \in K$, if $\sigma_1 \cap \sigma_2 \neq \emptyset$ then $\sigma_1 \cap \sigma_2$ is a common face of both $\sigma_1$ and $\sigma_2$.

For more detailed introduction to simplicial complexes, see Appendix B.

Using these definitions, we can define homology on simplicial complexes. For our purposes, we restrict ourselves to homology with coefficients in $\mathbb{Z}_2$. It is important to note that using a different field can affect the homology computation and result in the detection of different topological features [27].

**Definition 18** ($k$-chains)**.** Let $K$ be a finite simplicial complex, and $p$ a non-negative integer. The space of *$k$-chains* $C_p(K)$ is the $\mathbb{Z}_2$-vector space spanned by the $p$-simplices of $K$, where the sum of two $k$-chains is their symmetric difference.

**Definition 19.** The ***boundary*** of a $p$-simplex $\sigma$ is the $(p-1)$-chain

$$\partial_p(\sigma) := \sum_{\tau \in K_{p-1},\ \tau \subset \sigma} \tau,$$

where $K_{p-1}$ is the set of $(p-1)$-simplices of $K$.

Since the $p$-simplices form a basis of $C_p(K)$, we can extend $\partial_p$ to a linear map from $C_p(K)$ to $C_{p-1}(K)$, denoted the *boundary operator*. The kernel of this operator $\ker(\partial_p)$ consists of the *$p$-cycles* of $K$, and the image $\mathrm{im}(\partial_p)$ is the space of *$p$-boundaries* of $K$. It can be shown that $\mathrm{im}(\partial_{p+1}) \subset \ker(\partial_p)$.

**Definition 20** (Simplicial Homology)**.** For any $p \in \mathbb{N}$, the $p$-th ***simplicial homology*** group of a simplicial complex $K$ is the quotient vector space

$$H_p(K) := \ker(\partial_p)/\,\mathrm{im}(\partial_{p+1}).$$

The dimension $\beta_p(K)$ of $H_p(K)$ is called the *$p$-th* ***Betti number*** of $K$.

Intuitively, if we conceptualize a simplicial complex as a discretization of an arbitrary point cloud in a metric space, we notice the importance of specifying a *distance scale* at which it is constructed. Given a range of distance scales, we can construct a simplicial complex at each scale to form a nested sequence of complexes. This is the objective of filtered simplicial complexes, or *filtrations*.

**Definition 21** (Filtration)**.** A ***filtration*** $K$ is a nested sequence $\{K_i\}_{i=1}^n$ of simplicial complexes

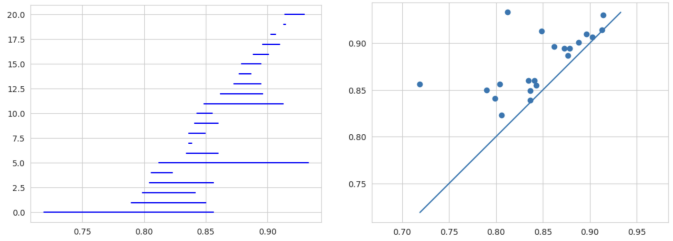
$$K_0 \subseteq K_1 \subseteq K_2 \subseteq \ldots \subseteq K_n, \quad \text{with} \quad \cup_{i \in I} K_i = K. \tag{16}$$

Given a filtration of simplicial complexes, *persistent homology* tracks the birth and death of homological features across distance scales as we move through the filtration.

**Definition 22** (Persistent homology)**.** Let $K = (K_i)_{i=1}^n$ be a filtration. The *$p$-th* ***persistent homology*** of a $K$ is the pairing $\left(\{H_p(K_i)\}_{i=1}^n, \{f_{i,j}\}_{i \leq j}\right)$, where each $f_{i,j} : H_p(K_i) \mapsto H_p(K_j)$ is a homomorphism induced by the inclusion map $K_i \mapsto K_j$.

Formally, for each $i$, we compute the $p$-dimensional homology $H_p(K_i)$ for each complex $K_i$ in the filtration. We say that $x \in H_p(K_i)$ is *born* in $H_p(K_i)$ if it is not in the image of $f_{i-1,i}$. We say that a feature $x$ *dies* in $H_p(K_j)$ if $j > i$ is the smallest index such that $f_{i,j}(x) = 0$. The half-open interval $[i, j)$ represents the lifetime of $x$. The *persistence* of a homological feature is the difference between its death and birth times. If $f_{i,j}(x) \neq 0$ for all $j > i$, its lifetime is the interval $[i, \infty)$.

**Figure 10**: Example of a 1-dimensional persistence barcode and diagram for the Starlink network embedding in $\mathbb{S}^9$.




### *Metrics for Persistence Diagrams*

In our analysis of temporal networks, we will obtain a sequence of persistence diagrams, one for each embedding of the network at each time step. We would like to compare the topological similarity of network embeddings across time. In order to do this, we need to measure the distance between persistence diagrams using a stable metric. A metric is *stable* if a small perturbation of the dataset creates only a small change in the persistence diagram (up to the metric). There are two commonly used metrics that are stable: the bottleneck distance and the Wasserstein distance metric. They are defined as follows.

**Definition 23.** Let $P$ and $Q$ be two persistence diagrams. The ***bottleneck distance*** between $P$ and $Q$ is defined as

$$d_B(P,Q) = \inf_{\gamma} \sup_{x \in P} \|x - \gamma(x)\|_\infty,$$

where $\gamma$ ranges over all matchings from $P$ to $Q$ and $\|p - q\|_\infty = \max(|p_1 - q_1|, |p_2 - q_2|)$ for $p = (p_1, p_2), q = (q_1, q_2) \in \overline{\mathbb{R}}^2$.

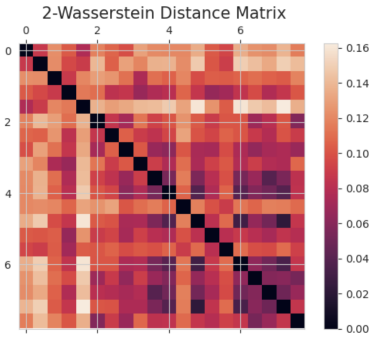
**Definition 24.** Let $P$ and $Q$ be two persistence diagrams. The *$p$-th* ***Wasserstein distance*** between $P$ and $Q$ is defined as

$$d_{W_p}(P,Q) = \inf_{\gamma} \left( \sum_{x \in P} \|x - \gamma(x)\|_p \right)^{1/p}$$

where $\gamma$ ranges over all matchings from $P$ to $Q$, and $\|p - q\|_p = (|p_1 - q_1|^p + |p_2 - q_2|^p)^{1/p}$ for points $p = (p_1, p_2), q = (q_1, q_2) \in \overline{\mathbb{R}}^2$.

The bottleneck distance measures the distance between two persistence diagrams $P$ and $Q$ by the maximum distance between two points in a matching $\gamma$ from $P$ to $Q$. In fact, the bottleneck distance gives the space of persistence diagrams a metric space structure [28]. However, it only outputs the distance between the greatest outliers, rather than the distance between all pairs of points. The Wasserstein distance considers the total distance between the matched pairs of points, and thus provides a more comprehensive measure for the similarity between persistence diagrams.

**Figure 11**: Wasserstein distance matrix for the Starlink network embedding in $\mathbb{S}^9$.




## 6. TOPOSES

Toposes provide a category-theoretic way to model temporospatial objects. We show different ways a time-varying graph can be modeled using toposes, which will provide more utility in understanding time-varying networks.

Due to the category-theoretic nature of topos theory, it is recommended that readers with no background in category theory read the introduction provided in Appendix A before proceeding with this section.

### *Category of Graphs*

A (directed) graph $G$ consists of two sets, $N$ (nodes, or vertices) and $A$ (arcs, or edges). It is accompanied with two functions, $s\colon N \to A$ (source) and $t\colon N \to A$ (target).

$$N \underset{t}{\overset{s}{\rightrightarrows}} A$$

Let $\Gamma$ be a category of two objects and two distinct non-identity morphisms.

Then, a graph $G$ can be written as a functor $G\colon \Gamma \to \mathbf{Set}$, and the category of graphs is a functor category

$$\mathbf{Graph} = \mathbf{Set}^{\Gamma}$$

where the homomorphism between two graphs $\varphi\colon G_1 \to G_2$ becomes a natural transformation between two functors $G_1$ and $G_2$. One can see that subobjects in $\mathbf{Graph}$ are subgraphs, and

**Proposition 1.** The terminal object of $\mathbf{Graph}$ is a graph with one node and one edge (cycle) going into itself.

**Proposition 2.** $\mathbf{Graph}$ is cartesian-closed.

### *Topos of Graphs*

In this subsection, we introduce toposes of graphs by constructing and demonstrating a subobject classifier.

**Definition 25** (Topos)**.** A **topos** is a cartesian-closed category $T$ with a **subobject classifier** $\Omega$ that admits all finite limits.

**Definition 26** (Topos of Graphs [29])**.** The presheaf category $\mathcal{G} = \mathbf{Set}^{\Gamma^{\mathrm{op}}}$ is called the *topos of graphs*.

"Presheaf category" here intuitively means the category behaves like a (pre)sheaf. Based on the definition above, a graph $G$ can be represented as a 'presheaf' in $\mathcal{G}$ in the sense that $G$ consists of:

- Set of nodes $G(N)$ and arcs $G(A)$.
- Maps $G(s)$ and $G(t)$ such that $G(A) \to G(N)$ which assigns each arcs their source and target.

**Definition 27.** A functor $F\colon \mathcal{C} \to \mathbf{Set}$ is a *representable functor* if it is naturally isomorphic[2] to $\mathrm{Hom}_{\mathcal{C}}(A, -)\colon \mathcal{C} \to \mathbf{Set}$ for some fixed object $A \in \mathcal{C}$.

Intuitively, a functor $F$ being representable means that there is an object $A \in \mathcal{C}$ that 'represents' $F$, in the sense that it encodes an action of an object on the category and 'classifies' the maps out of $A$. Representable functors are also called *representables* in short.

The representables for topos of graphs are (note the contravariance) the followings:

$$\Gamma(-, N) := \mathrm{Hom}_{\Gamma}(-, N) \text{ and } \Gamma(-, A) := \mathrm{Hom}_{\Gamma}(-, A)$$

[2]Natural isomorphism here roughly means a natural transformation that is isomorphic. A more precise definition exists, but for intuition purposes, leaving it at that should be sufficient.

such that (note that $\Gamma$ has two objects: $N$ and $A$)

$$\Gamma(N,N) = \{\mathbb{1}_N\} \text{ and } \Gamma(A,N) = \emptyset$$
$$\Gamma(N,A) = \{s,t\} \text{ and } \Gamma(A,A) = \{\mathrm{id}_A\}$$

Roughly speaking, $\Gamma(-,N)$ represents a node and $\Gamma(-,A)$ represents the arrow $A\colon s \to t$. The subobjects of those representables are then:

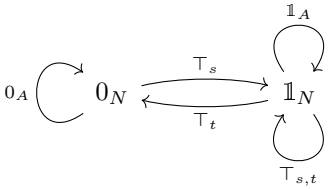
$$\Omega(N) = \{\text{Subobjects of } \Gamma(-,N)\} = \{0_N, \mathbb{1}_N\}$$

where $0_N$ represents 'the node is not in subgraph' and $\mathrm{id}_N$ represents 'the node is in subgraph,' and

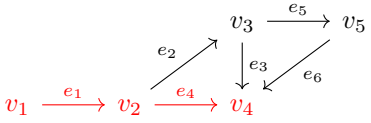
$$\begin{aligned}\Omega(A) &= \{\text{Subobjects of } \Gamma(-,A)\}\\ &= \{0_A, \mathbb{1}_A, \top_s, \top_t, \top_{s,t}\}\end{aligned}$$

where $\top_s$ and $\top_t$ are for the cases where the source/target is in the subgraph but target/source is not respectively, and $\top_{s,t}$ is for where the source and target are in the subgraph but the edge connecting them are not. The rest are self-explanatory.

Therefore, the subobject classifier $\Omega$ of **Graph** would look as follows as an object of **Graph** after being assembled altogether:



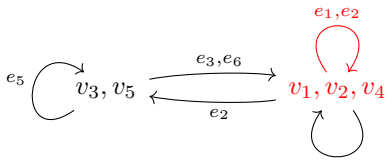

*Example.* Consider $G = (V,E)$ and its subgraph $H$ colored red below:




Per the definition of subobject classifier (Definition 40), our goal is to construct the following commutative diagram:

$$\begin{array}{ccc} H & \longrightarrow & 1 \\ {\scriptstyle m}\downarrow & & \downarrow{\scriptstyle t} \\ G & \xrightarrow{\chi_m} & \Omega \end{array}$$

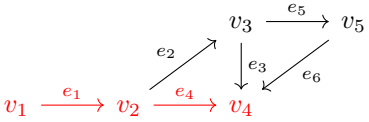
$\chi_m$ maps all nodes in $H$ (namely $v_1$, $v_2$, and $v_4$) to $\mathbb{1}_N$, and the rest ($v_3$ and $v_5$) to $0_N$. Similarly, it maps the edges in $H$ ($e_1$ and $e_2$) to $\mathbb{1}_A$, and the rest accordingly per their definition.




Roughly speaking, $\chi_m\colon G \to \Omega$ maps $H$ to $\{\mathbb{1}_A, \mathbb{1}_N\}$ and $G \setminus H$ to the rest. It is worthwhile to note that $(\mathbb{1}_A, \mathbb{1}_N)$ here, which contains what vertices and edges are in $H$, is a graph of one node and one edge looping around it, which is the terminal object of **Graph**. Therefore, intuitively, 'the truth classifier' $t\colon 1 \to \Omega$ decrees where in $\Omega$ should the 'correct answer' go into, and the classification map $\chi_m\colon G \to \Omega$ classifies the 'correct answer' encodes it into that location. This hence allows us to write $H$ as a pullback $H = G \times_\Omega 1$.

Based on this example, we can interpret $\Omega$ as an object that encodes the solution to a problem on a particular graph (that involves classifications of vertices and edges) such as the path-finding problem in a succinct, computable form. $\Omega$ also accurately encodes the relationship between the solution and the part that is not in the solution, as well as within the solution and the non-solution. This allows us to test if an alleged solution is precisely the solution, and if it is not, how close it is to the solution. The latter case can especially be handy in the case where the solution can, at best, be approximated. We elaborate on this below using routing (path-finding) as an example.

*Example.* Consider the previous example again with graph $G = (V,E)$ and its subgraph $H$ colored red:




Suppose that a packet has to be routed from $v_1$ to $v_4$. Assume that the shortest path is the path colored red $v_1 \xrightarrow{e_1} v_2 \xrightarrow{e_4} v_4$. By the time the packet was delivered to $v_2$ from $v_1$, the edge $v_2 \xrightarrow{e_4} v_4$ may not be available. After having the packet routed to $v_3$, instead of recomputing the path, with a subobject classifier, we know that we can eventually just take either $e_3$ or $e_6$ so we can get back to the original intended path. This by nature may not result in the shortest path deterministically, however; for example, it may not be clear to the packet (or even the nodes) whether $v_3 \xrightarrow{e_3} v_4$ is shorter than $v_3 \xrightarrow{e_5} v_5 \xrightarrow{e_6} v_4$ or not, because all subobject classifier informs is that the packet will eventually have to return to the original path by taking $e_3$ and $e_6$.

In the context of the celebrated contact graph routing [1], $\Omega$ can be thought of as a succinct 'pseudo-contact plan' (that can travel with the packet and) that itself is computable and where nodes can approximate the path for the packet or decide where to forward it to 'on the fly' without needing to store or update contact plans, as it already contains information on the intended path (computed as the shortest path, red vertices and edges) and edges that lead to the intended path ($e_3$ and $e_6$ in the diagram above). Similar to what we discussed above, it is indeed not always optimal; for example, if a packet is 'stranded' in the middle of the network far away from the red vertices and edges, then recomputing the shortest path is likely the best option.

As a future work, it might be interesting to see if there is a way to introduce a new type of subobject classifier where the elements of $0_N$ ($v_3$ and $v_5$) can be further classified by their distance to the elements in $\mathbb{1}_1$ ($v_1, v_2$, and $v_4$ – i.e. the path).

### *Topos of Time-Dependent Graphs and Hypergraphs*

We can extend the topos of graphs to the settings of time-dependent graphs and hypergraphs.

**Definition 28.** A time-dependent graph is defined as the topos of graph-valued presheaves on the category of natural numbers. In particular a time-dependent graph is a contravariant functor $\mathbb{N}$ to $\mathcal{G}$.

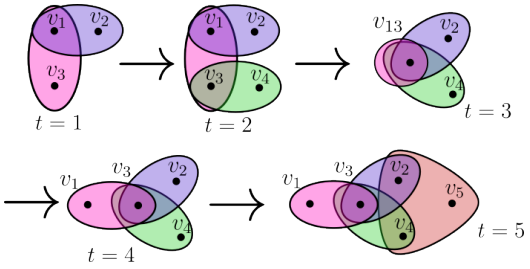



**Figure 12**: A time-dependent hypergraph. Note that the morphism of hypergraphs from $t = 2$ to $t = 3$ maps $v_1$ and $v_3$ to the same vertex $v_{13}$.

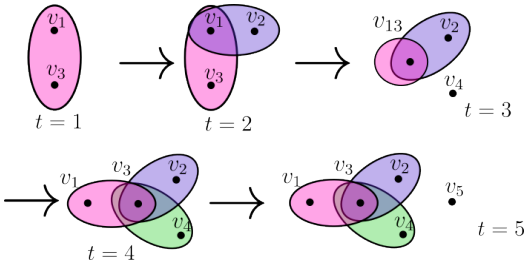



**Figure 13**: A subobject of the time-dependent hypergraph in Figure 12.

For a time-dependent graph, there are discrete time values given by the natural numbers $\mathbb{N}$, and for each $n \in \mathbb{N}$, the time-dependent graph assigns a graph $G_n$ to the time step. If $n \leq m$, we are provided a morphism $G_n \to G_m$. Thus for a time-dependent graph, the notions of time-dependence with communication networks coincide in the following ways. From a graph in one time step to another, we may gain new vertices, and we may gain edges between vertices. We may also have vertices merge with each other, which can model subgraphs becoming domains at a certain time step.

Using another diagram, we define a topos of directed hypergraphs. Consider a diagram with an object $N$ and objects $E_{i,j}$ for natural numbers $i, j$. For each such object, there are $i$ morphisms $s_1, \ldots, s_i : N \to E_{i,j}$ and $j$ morphisms $t_1, \ldots, t_j : N \to E_{i,j}$. Denote this diagram $\mathcal{H}$. Then the category of presheaves $\mathbf{DHypergraph} = \mathbf{Set}^{\mathcal{H}}$ forms a topos, which we call the topos of directed hypergraphs. As a word of caution, like graphs, this definition of hypergraphs allows for "self" hyperedges. In this definition, hyperedges can also take a vertex multiple times.

Likewise, we define a time-dependent hypergraph:

**Definition 29.** A time-dependent hypergraph is defined as the topos of hypergraph-valued presheaves on the category of natural numbers. In particular, the objects of the category are functors $\mathcal{F} : \mathbb{N} \to \mathbf{DHypergraph}$, and the morphisms are natural transformations between them.

*Example.* Consider the sequence of hypergraphs with maps between them described in Figure 12. A subobject is described in Figure 13. As in the example for graphs mentioned earlier, suppose we would like to deliver a packet from $v_1$ to $v_4$. In the subobject, a path may be described, but the data must be carried at vertex $v_3$ until $t = 4$ before being delivered to its destination.

# 7. Conclusion

In this paper, we explore several promising ways to detect and model networking architectures for DTNs, and provide worked examples to the reader. It is the authors' belief that these constructions will prove crucial to realizing the Solar System Internet, which will feature numerous space network service providers in addition to the expected myriad of space- and Earth-based nodes. As the theoretical constructs begin to give rise to germane examples, the next major step is to use these tools to model notional LunaNet architectures and prove that routing remains well-defined despite the novelty of the proposed approach. There are numerous other worthy follow-on research projects which are suggested below.

### *Future Works*

*Sheaves on Hypergraphs*— Hypergraphs have topological properties that are proven to be useful in application [30, 31]. As stated before, hypergraphs more accurately model interactions like broadcast and multicast in the space network setting. However, there remain open questions around what routing should look like on this topology. Some considerations for future work include the following.

1. Construction of a distance path sheaf on an undirected hypergraph topology should be possible. This could allow for a distributed sheaf framework to interrogate message passing times with redundancies or multi-route options.

2. The containment poset of a hypergraph is the most natural poset to build a sheaf on, but there are other definitions of containment we could consider. If we wanted to explore the more restricted space of global sections on a fully directed hypergraph, we could consider doing so. We would need a definition of directed edge containment, for the poset, motivated by the routing setting we are in.

3. There could be interesting applications of these path sheaves to the simplicial closure of a hypergraph instead of the hypergraph itself. This would simplify the poset and potentially allow for us to use some established tools with our sheaf constructions. For example, in the space of cellular sheaves (cellular meaning they're built on a cell complex) we may consider diffusion with something like the sheaf Laplacian. Generally, the regularity afforded by a simplicial complex could prove useful for some applications like diffusion across a hypergraph network.

4. Investigate ways to represent hypergraphs as schemes, and then experiment with topological refinements until one is found that naturally supports the path-sheaf construction. This should lead to a deterministic way to induce the "right" topology on a general hypergraph to support routing constructions.

*Network Embeddings*—In our work, the mixed-curvature embedding method we used can be easily modified by choosing the product space and the auxiliary loss function. Although we only included a term for distance-based loss in our objective function, as done in [23], one could incorporate curvature-based loss or proximity-based loss terms beyond the 1-hop neighborhoods, as suggested by Bronstein *et al.* [22]. This may be more costly to compute, but can more accurately represent the structure of the network. While we implicitly accounted for the discrete curvature of the network by embedding it into a product manifold with component spaces that reflects the geometry of communities detected with Ricci flow, we did not explicitly estimate the curvature of each component, as done in [23].

Further, in our embeddings of graph sequences, we fixed the total structure and dimension of the product space, which may have detracted from our ability to generate highly accurate representations. Ideally, at each time step, we would like to embed the network into a (possibly) different product manifold that represents the geometry of the network at that given time. However, this approach becomes especially difficult when faced with the problem of node alignment. It may be possible to define some sort of smooth map between consecutive Riemannian manifolds that minimizes distances between embedded nodes across time steps. Or, one could try to leverage the actions of Lie groups on the manifold to align their underlying coordinate systems.

*Topos Theory*—Toposes allow, in a sense, logic to become regional; something that is locally true might not extend to global truth. What makes this useful is that a topos formalizes the connections between locale, meaning that regions that will always have some differences can still have rigorous transformations between them. Combined with the application in this paper of modeling temporospatial networks, there are multiple paths ahead.

One example of regional truth is given by relativistic effects on clocks. Two otherwise similar clocks in different gravities, say Earth and Lunar orbits, will tick at different rates. This non-error source of drift can lead to difficulties with time-sensitive actions, such as satellite pointing, acquisition, and tracking. Moreover, it adds complexity to time synchronization in a DTN. The natural extension of the work done by the authors in time synchronization for DTNs can be extended to include relativistic effects using topos theory.

Roughly put, their structure enables us to model logic by allowing one to define logical operations that can record where and when a logical statement is true; as Section 6 showed, topos allowed us to represent classification-type problems on graphs in a readily computable form. Topos theory with higher-order category theory appears to be the most canonical method of modeling the behavior of systems and decision making process [32, 33]. Given these, we ambitiously speculate that toposes might be the natural setting within which to model DTNs.

## REFERENCES


[1] J. A. Fraire, O. De Jonckère, and S. C. Burleigh, "Routing in the space internet: A contact graph routing tutorial," *Journal of Network and Computer Applications*, vol. 174, p. 102884, 2021.

[2] A. Hylton, N. Tsuei, M. Ronnenberg, J. Hwang, B. Mallery, J. Quartin, C. Levaunt, and J. Quail, "Advances in Modeling Solar System Internet Structures and their Data Flows," in *2023 IEEE Aerospace Conference*, Big Sky, MT, USA, Mar. 2023, pp. 1–20.

[3] A. Hylton, D. Israel, M. Wennersten *et al.*, "The international space station, optical communications, and delay tolerant networking: Towards a solar system internet architecure," *41st International Communications Satellite Systems Conference (ICSSC 2024)*, 2024.

[4] D. J. Israel, K. D. Mauldin, C. J. Roberts, J. W. Mitchell, A. A. Pulkkinen, L. V. D. Cooper, M. A. Johnson, S. D. Christe, and C. J. Gramling, "Lunanet: a flexible and extensible lunar exploration communications and navigation infrastructure," in *2020 IEEE Aerospace Conference*, 2020, pp. 1–14.

[5] A. Hylton, R. Short, R. Green, and M. Toksoz-Exley, "A mathematical analysis of an example delay tolerant network using the theory of sheaves," in *2020 IEEE Aerospace Conference*. IEEE, 2020, pp. 1–11.

[6] R. Short, A. Hylton, R. Cardona, R. Green, G. Bainbridge, M. Moy, and J. Cleveland, "Towards sheaf theoretic analyses for delay tolerant networking," in *2021 IEEE Aerospace Conference*. IEEE, 2021, pp. 1–9.

[7] R. Short, A. Hylton, J. Cleveland, M. Moy, R. Cardona, R. Green, J. Curry, B. Mallery, G. Bainbridge, and Z. Memon, "Sheaf theoretic models for routing in delay tolerant networks," in *2022 IEEE Aerospace Conference*, 2022.

[8] A. Hylton, B. Mallery, J. Hwang, M. Ronnenberg, M. Lopez, O. Chiriac, S. Gopalakrishnan, and T. Rask, "Multi-domain routing in delay tolerant networks," in *2024 IEEE Aerospace Conference*, 2024, pp. 1–20.

[9] M. Moy, R. Cardona, R. Green, J. Cleveland, A. Hylton, and R. Short, "Path Optimization Sheaves," Dec. 2020, arXiv preprint arXiv:2012.05974 [cs.NI].

[10] M. Robinson, *Recent Applications of Harmonic Analysis to Function Spaces, Differential Equations, and Data Science*. Springer, 2017, ch. Sheaf and duality methods for analyzing multi-model systems, pp. 653–703.

[11] ——, "Hunting for Foxes with Sheaves," *Notices of the American Mathematical Society*, vol. 66, no. 05, p. 661, May 2019.

[12] Q. Li, G. Kim, and R. Negi, "Maximal scheduling in a hypergraph model for wireless networks," in *2008 IEEE International Conference on Communications*. IEEE, 2008, pp. 3853–3857.

[13] H. Zhang, L. Song, Y. Li, and G. Y. Li, "Hypergraph theory: Applications in 5g heterogeneous ultra-dense networks," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 70–76, 2017.

[14] T. Nyasulu and D. H. Crawford, "Comparison of graph-based and hypergraph-based models for wireless network coexistence," in *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE, 2021, pp. 203–208.

[15] S. Ladkani, "On derived equivalences of categories of sheaves over finite posets," *Journal of Pure and Applied Algebra*, vol. 212, no. 2, pp. 435–451, 2008.

[16] J. M. Curry, "Sheaves, cosheaves and applications," Ph.D. dissertation, University of Pennsylvania, 2014.

[17] S. G. Aksoy, C. Joslyn, C. Ortiz-Marrero, B. Praggastis, and E. Purvine, "Hypernetwork science via high-order hypergraph walks," *EPJ Data Science*, vol. 9, no. 1, p. 16, Dec. 2020.

[18] Y. Ollivier, "Ricci curvature of markov chains on metric spaces," *Journal of Functional Analysis*, vol. 256, no. 3, pp. 810–864, 2009.

[19] C.-C. Ni, Y.-Y. Lin, F. Luo, and J. Gao, "Community detection on networks with ricci flow," *Scientific Reports*, 2019.

[20] C. Cruceru, G. Bécigneul, and O.-E. Ganea, "Computationally tractable riemannian manifolds for graph embeddings," in *Proceedings of the 35th Conference on Artificial Intelligence (AAAI)*, vol. 35, no. 8, Feb. 2021, pp. 7133–7141.

[21] R. C. Wilson, E. R. Hancock, E. Pekalska, and R. P.

Duin, "Spherical and hyperbolic embeddings of data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2255–2269, 2014.

[22] F. D. Giovanni, G. Luise, and M. M. Bronstein, "Heterogeneous manifolds for curvature-aware graph embedding," in *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022.

[23] A. Gu, F. Sala, B. Gunel, and C. Ré, "Learning mixed-curvature representations in product spaces," in *International Conference on Learning Representations (ICLR)*, vol. 5, 2019.

[24] C. D. T. Barros, M. R. F. Mendonça, A. B. Vieira, and A. Ziviani, "A survey on embedding dynamic graphs," *ACM Comput. Surv.*, vol. 55, no. 1, nov 2021.

[25] S. Bonnabel, "Stochastic gradient descent on riemannian manifolds," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2217–2229, 2013.

[26] R. Forman, "Bochner's method for cell complexes and combinatorial ricci curvature," *Discrete and Computational Geometry*, vol. 29, pp. 323–374, 01 2003.

[27] G. Carlsson, G. Singh, and A. Zomorodian, "Computing multidimensional persistence," in *Algorithms and Computation*, Y. Dong, D.-Z. Du, and O. Ibarra, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 730–739.

[28] M. E. Aktas, E. Akbas, and A. E. Fatmaoui, "Persistence homology of networks: methods and applications," *Applied Network Science*, vol. 4, 2019.

[29] S. Vigna, "A guided tour in the topos of graphs," *arXiv preprint arXiv:math/0306394 [math.CT]*, 2003.

[30] C. A. Joslyn, S. G. Aksoy, T. J. Callahan, L. E. Hunter, B. Jefferson, B. Praggastis, E. Purvine, and I. J. Tripodi, "Hypernetwork science: from multidimensional networks to computational topology," in *International conference on complex systems*, 2020, pp. 377–392.

[31] A. Myers, C. Joslyn, B. Kay, E. Purvine, G. Roek, and M. Shapiro, "Topological analysis of temporal hypergraphs," in *International Workshop on Algorithms and Models for the Web-Graph*, 2023, pp. 127–146.

[32] D. I. Spivak, "Higher-dimensional models of networks," *arXiv preprint arXiv:0909.4314 [cs.NI]*, 2009.

[33] A. Speranzon, D. I. Spivak, and S. Varadarajan, "Abstraction, composition and contracts: A sheaf theoretic approach," *arXiv preprint arXiv:1802.03080 [cs.LO]*, 2018.

[34] S. Awodey, *Category theory*, 2nd ed., ser. Oxford Logic Guides. Oxford University Press, 2010, vol. 52.

[35] G. E. Bredon, *Sheaf theory*, ser. Graduate Texts in Mathematics. Springer New York, 1997, vol. 170.

[36] I. R. Shafarevich, *Basic algebraic geometry 2: Schemes and complex manifolds*, 3rd ed. Springer Science & Business Media, 2013.

[37] Stacks Project Authors, "*Stacks Project*," https://stacks.math.columbia.edu, 2018.

[38] A. D. Shepard, "A cellular description of the derived category of a stratified space," Ph.D. dissertation, Brown University, 05 1985.

[39] J. Friedman, *Sheaves on graphs, their homological invariants, and a proof of the Hanna Neumann conjecture: with an appendix by Warren Dicks*. American Mathematical Society, 2015, vol. 233, no. 1100.

# APPENDICES

## A. CATEGORY THEORY

This section is primarily based on [34].

### *Basic Notions*

**Definition 30** (Category)**.** A **category** $\mathcal{C}$ consists of *objects* and *morphisms* (also called '*arrows*' or *maps*) between the objects, where morphisms should satisfy the following (here, $A, B, C$, and $D$ denote objects in $\mathcal{C}$):

- Given two morphisms $f\colon A \to B$ and $g\colon B \to C$, there exists a morphism $f \circ g\colon A \to C$ referred as the *composition* of $f$ and $g$.
- The *identity* morphism $\mathbf{1}_A\colon A \to A$ satisfies $f \circ \mathbf{1}_A = \mathbf{1}_B \circ f = f$ for all morphisms $f\colon A \to B$.
- *Associativity* $h \circ (g \circ f) = (h \circ g) \circ f$ holds for all morphisms $f, g$, and $h$.

In addition, a morphism $f\colon A \to B$ is referred to as:

- *Monomorphism*: for any $g_1, g_2\colon C \to A$, if $fg_1 = fg_2$ then $g_1 = g_2$.
- *Epimorphism*: for any $h_1, h_2\colon B \to D$, if $h_1 f = h_2 f$ then $h_1 = h_2$.
- *Isomorphism*: there exists a morphism $g\colon B \to A$ such that $g \circ f = \mathbf{1}_A$ and $f \circ g = \mathbf{1}_B$. We write $g = f^{-1}$ and $A \cong B$.

*Example.* $\mathbf{Set}$ is the category whose objects are sets and morphisms are functions.

**Definition 31** (Partial Order)**.** A partial order on a set $X$ is a relation $\leq$ on $X$ such that, for all $x, y, z \in X$:

- Reflexive: $x \leq x$.
- Anti-symmetric: if $x \leq y$ and $y \leq x$ then $x = y$.
- Transitive: if $x \leq y$ and $y \leq z$ then $x \leq z$.

And the pair $(X, \leq)$ is called a *partially ordered set*, or *poset*.

*Example.* A poset $P$ equipped with the relation $\leq$ denoted as $(P, \leq)$ is itself a category where $x \leq y$ is the morphism between two objects $x, y \in P$.

**Definition 32** (Functors)**.** A *functor* $F\colon \mathcal{C} \to \mathcal{D}$ between categories $\mathcal{C}$ and $\mathcal{D}$ is a mapping of objects to objects and morphisms to morphisms that preserves:

- domains & codomains: $F(f\colon A \to B) = F(A) \to F(B)$,
- identity morphisms: $F(\mathbf{1}_A) = \mathbf{1}_{F(A)}$, and
- composition of morphisms: $F(g \circ f) = F(g) \circ F(f)$.

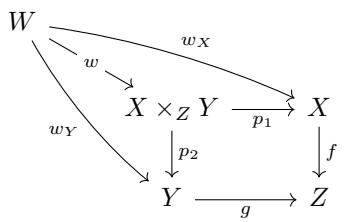
$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ {\scriptstyle g}\downarrow & \nearrow_{h} & \\ C & & \end{array} \quad \Longrightarrow \quad \begin{array}{ccc} F(A) & \xrightarrow{F(f)} & F(B) \\ {\scriptstyle F(g)}\downarrow & \nearrow_{F(h)} & \\ C & & \end{array}$$

*Example.* We can consider a group $\mathbb{G}$ as a one-object category (the group $\mathbb{G}$ would be a category with a single object $*$, where each $g \in G$ is a morphism $g\colon * \to *$) and define a functor $F\colon \mathbb{G} \to \mathbf{Set}$. We remark that $F$ is a group action.

**Definition 33** (Pullback)**.** Let $\mathcal{C}$ be a category with objects $X, Y, Z \in \mathcal{C}$ and morphisms $f\colon X \to Z$ and $g\colon Y \to Z$. The *pullback* of $X$ and $Y$, along with morphisms $f$ and $g$, is an object $P := X \times_Z Y$ with morphisms $p_1\colon X \times_Z Y \to X$ and $p_2\colon X \times_Z Y \to Y$ such that the diagram below commutes.

$$\begin{array}{ccc} X \times_Z Y & \xrightarrow{p_1} & X \\ {\scriptstyle p_2}\downarrow & & \downarrow{\scriptstyle f} \\ Y & \xrightarrow[g]{} & Z \end{array}$$

**Proposition 3** (Universal Property of Pullback)**.** The pullback construction above is *universal*; that is, given any $w_X : W \to X$ and $w_Y : W \to Y$ such that $w_X \circ f = w_Y \circ Y$ (i.e., the diagram below commutes), there exists a unique isomorphism $w := \langle w_X, w_Y \rangle : Z \to X \times_Z Y$.




*Example.* Let $\mathbb{N}$ be a poset of natural numbers where divisibility is the partial order (i.e., $m \mid n$ then $m \leq n$). Then the pullback of $m$ and $n$ is $\gcd(m, n)$.

**Definition 34** (Initial and Terminal Objects)**.** An object $0 \in \mathcal{C}$ is an *initial object* if for every $c \in \mathcal{C}$, there exists a unique morphism $f : 0 \to c$. An object $1 \in \mathcal{C}$ is a *terminal object* if for every $c \in \mathcal{C}$, there exists a unique morphism $c \to 1$.

*Example.* Let $X$ and $Y$ be objects of category $\mathcal{C}$, and $1 := 1_{\mathcal{C}}$ be a terminal object of $\mathcal{C}$. Then the pullback $X \times_1 Y$ is the product $X \times Y$.

*Remark.* We can observe that an element $a$ in a set $A \in \mathbf{Set}$ can be represented as a global element $a : 1 \to A$ where 1 is a terminal object of $\mathbf{Set}$. In fact, global elements can be seen as a way to denote elements from $\mathbf{Set}$. That is, there is an isomorphism:

$$A \cong \mathrm{Hom}_{\mathbf{Set}}(1, A)$$

Moreover, in $\mathbf{Set}$, the empty set $\emptyset \in \mathbf{Set}$ is the unique initial object, and singletons are terminal objects. This is, in fact, the reason 1 and 0 are usually used to denote a terminal and initial object, respectively (also as they are the unit and zero of the cartesian product, respectively). By a similar logic, in $\mathbf{Grp}$, any trivial group is a zero object (an object that is both an initial and terminal object).

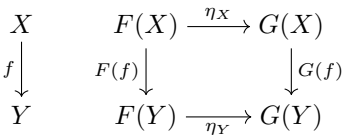
With the remark above in mind, one can observe that there is an 'equivalence' relationship between an object and a functor in the sense that any object $C \in \mathcal{C}$ can be seen as a 'constant' functor $C : \mathcal{I} \to \mathcal{C}$ where $\mathcal{I}$ is an 'indexing' category (or the 'category of indices').

Now, consider a functor $X : \mathcal{I} \to \mathcal{C}$. As both $C$ and $X$ have the same domain and codomain, we can transform from one to the other in a way that it preserves the 'category-ness' of the domain and codomain. This is called natural transformation.

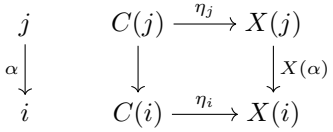
**Definition 35** (Natural Transformation)**.** A natural transformation $\eta$ is an operation between functors $F, G : \mathcal{C} \to \mathcal{D}$ from category $\mathcal{C}$ to $\mathcal{D}$ such that:

1. for all object $X \in \mathcal{C}$, $\eta_X : F(X) \to G(X)$, and
2. for all $f : X \to Y$ in $\mathcal{C}$, $\eta_Y \circ F(f) = G(f) \circ \eta_X$.

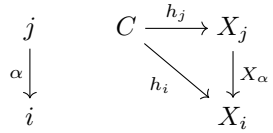
In other words, the following diagram commutes:




Let $i, j \in \mathcal{I}$ be objects and $\alpha : j \to i$ be a morphism. Then we can consider the following natural transformation:

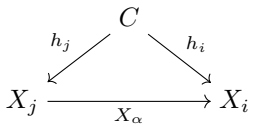



But in reality, $C$ is an object of $\mathcal{C}$. That is, what we have drawn above is this:



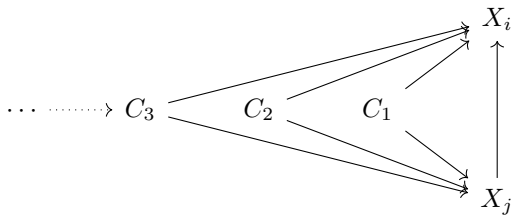

where $X_i$ and $X_j$ are notations for $X(i)$ and $X(j)$ respectively. Writing this more formally, we reach a new collection of objects called **cones**.

**Definition 36** (Cone)**.** Let $\mathcal{I}$ (called *index category*) and $\mathcal{C}$ be categories and $X : \mathcal{I} \to \mathcal{C}$ be a functor (called *diagram of type* $X$). A *cone* of $X$, denoted $\mathbf{Cone}(\mathrm{X})$, is a collection of objects $C \in \mathcal{C}$ and maps $h_i : C \to X_i$ for all $i \in \mathcal{I}$ such that, for any morphism $\alpha : j \to i$ in $\mathcal{I}$, the following diagram commutes:




and $\mathbf{Cone}(\mathrm{X})$ itself is a category.

For visualization purposes, suppose that the objects in $\mathcal{C}$ are 'ordered' in a way that $C_1 \in \mathcal{C}$ is 'below' $C_2 \in \mathcal{C}$ if there is a map $C_2 \to C_1$, and so on. The diagram in the definition would then look as follows:

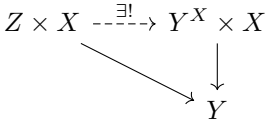



There is some sequence-like structure here, where the object where the sequence 'terminates' gives the smallest cone. This, finally, motivates and defines **limits**.

**Definition 37** (Limit)**.** A *limit* of $X : \mathcal{I} \to \mathcal{C}$ denoted $\varprojlim_i X_i$ is the terminal object of $\mathbf{Cone}(\mathrm{X})$. We say the limit is *finite* if the index category $\mathcal{I}$ is finite.

Given two sets $X, Y$, we have a set $Z$ which consists of all functions $X \to Y$. This is called an **exponential object** $Y^X$, which also satisfies a universal property. This object can thus be defined on categories other than the category of sets.

**Definition 38.** Let $\mathcal{C}$ be a category. Given two objects $X$ and $Y$ in the category, an exponential object $Y^X$ is an object in $\mathcal{C}$ with map $Y^X \times X \to Y$ satisfying the following universal property: given any object $Z$ with morphism $Z \times X \to Y$, there exists a unique map $Z \to Y^X$ such that the following diagram commutes:



***Subobject Classifiers***

Roughly speaking, if $X$ be an object in a category $\mathcal{C}$ then we can have a notion of subobjects. To say that $S$ is a 'subobject' of $X$, let $m\colon S \to X$ be the 'inclusion map.' Then $m$ is injective, and $m$ can be used to 'represent' $S$ and its relationship with $X$. Formulating this in a category-theoretic language:

**Definition 39** (Subobject)**.** Let $X$ be an object of a category $\mathcal{C}$. A subobject of $X$ is a monomorphism $m\colon S \to X$.

*Example.* As described, in $\mathbf{Set}$, subobjects are subsets. For example, in $\mathbf{Set}$, the inclusion map from $\{1,2\}$ to $\{1,2,3\}$ is a subobject.

**Definition 40** (Subobject Classifiers)**.** Let $\mathcal{C}$ be a category with all finite limits. A *subobject classifier* in $\mathcal{C}$ consists of an object $\Omega$ with a morphism $t\colon 1 \to \Omega$ that is a **universal subobject** in the following sense: Given any object $C$ and any subobject $U \to C$, there is a unique arrow $\chi\colon C \to \Omega$ making the following diagram a **pullback** (see Definition 33):

$$\begin{array}{ccc} U & \longrightarrow & 1 \\ \downarrow & & \downarrow t \\ C & \xrightarrow{\chi} & \Omega \end{array}$$

where $\chi$ is called the *classifying arrow* of the subobject $U \to C$ that takes $U \subset C$ to the *point* $t$ of $\Omega$.

Here, $1$ denotes the terminal object of $\mathcal{C}$, as denoted in Definition 34. For intuition-building purposes, one can consider subobject classifiers as: for every subobject $U$ of $C$, there is a 'predicate' on $C$ that 'cuts' $U$ out from $C$. This should get clearer after the following example:

*Example.* Let $\mathcal{C} = \mathbf{Set}$ where $\Omega = \{0,1\}$, $1 = \{1\}$, and $U \subseteq C \in \mathcal{C}$. Then $U$ would 'induce' a 'characteristic function' $u\colon C \to \Omega$ that classifies whether an element in $C$ should be in $U$ or not, and $U$ can be represented as a 'product' of $C$ and $1$ in a way that $U$ consists of elements selected from $C$.

## B. SHEAVES ON GRAPHS

In this section, we introduce the basic notions of sheaves and simplicial complexes, leading to the definition of cellular sheaves and sheaves on graphs. Additionally, we also recommend that readers with no prior exposure to algebraic topology read this section before reading Section 5.

Given a set $X$, a topology on $X$ gives a sense of whether two points on $X$ are 'close' or not, in the sense that a topology is a collection of open sets called *neighborhoods*, which contain points that are close to each other.

**Definition 41** (Topology)**.** Let $X$ be a set. Then a topology on $X$, $\tau$ is a subset of the power set $\mathcal{P}(\mathcal{X})$ which satisfies the following properties:

- $\varnothing, X \in \tau$.
- $\tau$ is closed under arbitrary unions. Given any collection $U_\alpha$ of sets in $\tau$ indexed by $J$ ( $\alpha \in J$ ), $\bigcup_\alpha U_\alpha \in \tau$.
- $\tau$ is closed under finite intersections. Given open sets $U_1, \ldots, U_n, \bigcap_{j=1}^n U_j \in \tau$.

A set with topology $(X, \tau)$ is called a *topological space*.

In literature, a symbol for the topology is rarely used, and open sets are discussed under the assumption that the topology is known. We will follow this convention.

*Example.* Trivially, on $\mathbb{R}$, the usual topology is the topology given by open intervals $(a,b) \subset \mathbb{R}$. From this, one can also see that metric spaces are topological spaces (but the converse is not true).

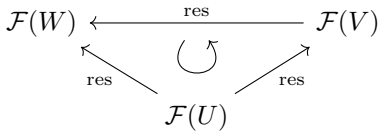
***Basic Notions***

The main references for this subsection are [35, 36], and [37, Chapter 6].

Let $\mathbf{Open}(X)$ be the category that has open subsets of $X$ as objects and inclusions as morphisms; that is, if $V \subseteq U$ for some open sets $V$ and $U$ of $X$ then $V \to U$.

**Definition 42** (Presheaf)**.** Let $X$ be a topological space and $\mathcal{C}$ be a category. A *presheaf* $\mathcal{F}$ on $X$ is a contravariant functor $\mathcal{F}\colon \mathbf{Open}(X) \to \mathcal{C}$ such that

- If $V \to U$ (i.e., $V \subseteq U$) then we have a morphism $\mathrm{res}_{U,V}\colon \mathcal{F}(U) \to \mathcal{F}(V)$.
- For all $U$, $\mathrm{res}_{U,U} = \mathrm{id}_{\mathcal{F}(U)}$.
- If $W \to V \to U$ ($W \subseteq V \subseteq U$) then $\mathrm{res}_{U,W} = \mathrm{res}_{U,V} \circ \mathrm{res}_{V,W}$ and hence the following diagram commutes.




For brevity, two notations $\mathrm{res}_{U,V}(f)$ and $f|_V$ may be used interchangeably throughout this section. Also, a *contravariant functor* $\mathcal{F}\colon \mathcal{C} \to \mathcal{D}$ means it is a functor $\mathcal{F}\colon \mathcal{C}^{\mathsf{op}} \to \mathcal{D}$, where $\mathcal{C}^{\mathsf{op}}$ is the *opposite* category of $\mathcal{C}$: same objects but with morphisms in the reversed direction.

**Definition 43** (Section)**.** Let $\mathcal{F}$ be a presheaf over a topological space $X$. An element of $\mathcal{F}(U)$ is called a *section* of $\mathcal{F}$ over $U$, or more generally, a *local section* of $\mathcal{F}$. An element of $\mathcal{F}(X)$ is called a *global section* of $\mathcal{F}$.

**Definition 44** (Sheaf)**.** A presheaf $\mathcal{F}$ on $X$ is a *sheaf* if for all open $U \subseteq X$ and every open cover $\{U_i\}_{i \in I}$ of $U$, it satisfies the followings:

- *Gluability*: If $f_i \in \mathcal{F}(U_i)$ where $f_i|_{U_i \cap U_j} = f_j|_{U_i \cap U_j}$ for all $i$ and $j$, then there exists $f \in \mathcal{F}(U)$ such that $f|_{U_i} = f_i$ for all $i$.
- *Uniqueness*: If $s_1, s_2 \in \mathcal{F}(U)$ such that $s_1|_{U_i} = s_2|_{U_i}$ for all $i \in I$, then $s_1 = s_2$.

***Simplicial Complex***

Sheaves are axiomatically defined over topological spaces. A natural starting point in topology is simplicial complexes, which *resemble* simple networks. Let us start by restating Definition 17 (with slightly different wordings).

**Definition 45** (Simplicial Complex)**.** A *simplicial complex* $K$ is a set of simplices where:

- If $\sigma \in K$ then $K$ contains every face of $\sigma$.
- For all $\sigma, \tau \in K$ such that $\sigma \cap \tau \neq \emptyset$, $\sigma \cap \tau$ is a face of both $\sigma$ and $\tau$.

We say $K$ is a *simplicial $k$-complex* if the largest dimension among all simplices in $K$ is $k$.

Here, a **$k$-simplex** (dimension-$k$ simplex) is a convex hull of $k+1$ vertices in $k$-dimensional space that is a $k$-dimensional

polytope $\Delta^k$; they are $k$-dimensional objects that exist in a $k$-dimensional space, but not in smaller dimensions.

$$\Delta^k := \left\{ (x_0, x_1, \cdots, x_k) \;\middle|\; \sum_{i=0}^{k} x_i = 1, 0 \le x_i \le 1 \right\}$$

They are essentially the $k$-dimensional version of a triangle: 0-simplex is a vertex, 1-simplex is an edge, 2-simplex is a triangle, 3-simplex is a tetrahedron, 4-simplex is a pentatope, and so on. A **face** of a simplex is a simplex generated by a subset of the vertices of that simplex; similarly, a $d$-simplex of a $k$-simplex (the convex hull of $d+1$ vertices out of the $k+1$ vertices defining the $k$-simplex) is called a $d$-**face** of the $k$-simplex.

Simplicial complexes, although interesting and easy to visualize, are rather geometric. We 'abstract' out its geometric aspects into combinatorial forms, make it more computationally tractable and easier to algorithmize.

**Definition 46** (Abstract Simplicial Complex)**.** An *abstract simplicial complex* (ASC) $X$ is a collection of finite sets such that if $\sigma \in X$, then $\tau \in X$ for any $T \subseteq S$. More precisely, given a set $A$, an abstract simplicial complex $X$ is a subset of the power set $\mathcal{P}(A)$ such that

- $\{a\} \in X$ for all $a \in A$.
- If $\sigma \in X$ and $\tau \subseteq \sigma$, then $\tau \in X$.

The terms $k$-*simplex* and *face* can be defined for abstract simplicial complexes analogously.

*Remark.* By definition, abstract simplicial complexes are equivalent to independent systems, which are augmentation property away from being matroids. From this, we can also see that hypergraphs are generalizations of (abstract) simplicial complexes, as independent systems are set families with downward-closedness (Condition 2 of Definition 46).

Simplicial complexes have simplicies as its 'cells' intuitively. For now, let $k$-**cell** be a space homeomorphic to an $k$-dimensional ball $\mathbb{B}^k$ (also called $k$-**balls**), which is the interior of the $k$-dimensional disk:

$$\mathbb{D}^k := \left\{ x \in \mathbb{R}^k \;\middle|\; \|x\| \le 1 \right\}$$
$$\mathbb{B}^k := \mathrm{Int}(\mathbb{D}^k) = \left\{ x \in \mathbb{R}^k \;\middle|\; \|x\| < 1 \right\}$$

and its boundary is the $(k-1)$-sphere:

$$\mathbb{S}^{k-1} := \partial \mathbb{B}^k = \left\{ x \in \mathbb{R}^k \;\middle|\; \|x\| = 1 \right\}$$

Unless specified, $\|\cdot\| \colon \mathbb{R}^n \to [0, \infty)$ denotes the standard Euclidean norm in $\mathbb{R}^n$.

The following result gives the big picture of the construction we aim to achieve.

**Proposition 4.** For all $k \in \mathbb{N}$, every $k$-simplex is homeomorphic to the $k$-ball, and its boundary is homeomorphic to the $(k-1)$-sphere. Moreover, for all $k \ge 1$, $\mathbb{D}^k \big/ \mathbb{S}^{k-1} \cong \mathbb{S}^k$.

Let $X$ be a topological space where there is a nested sequence of topological spaces (this can be seen as a sort of filtration, described in Definition 21 in Section 5):

$$\emptyset \subseteq X^{(0)} \subseteq X^{(1)} \subseteq X^{(2)} \subseteq \cdots \subseteq X^{(d)} = X$$

where $X^{(k)}$ is called $k$-**skeleton**, such that:

(0) $X^{(0)}$ is 0-cells: set of points.
(1) $X^{(1)}$ is created by *attaching* 1-balls to $X^{(0)}$.
$\vdots$
($d$) $X^{(d)}$ is created by *attaching* $d$-balls to $X^{(d-1)}$.

Here, the $k$-balls attached to $X^{(k-1)}$ are called the $k$-**cells** of $X$. Also, **attaching** in this context means the map $\sigma_\alpha^{(k)} \colon \partial e_\alpha^{(k)} \to X^{(k-1)}$ and $X^{(k)}$ created by gluing the boundaries of $k$-cells $e_\alpha^{(k)}$ to $X^{(k-1)}$; more precisely, $X^{(k)}$ is the disjoint union

$$X^{(k)} = X^{(k-1)} \sqcup_\alpha e_\alpha^{(k)}$$

where $\sqcup_\alpha$ is the disjoint union that also identifies $x \in \partial e_\alpha^{(k)}$ and $\sigma_\alpha^{(k)}(x) \in X^{(k-1)}$ for all $x \in \partial e_\alpha^{(k)}$.

Extending this to infinite dimensions and defining the topologies gives us the definition of CW complex (cell complex). Before moving into the definition, let us take a look at an example:

*Example.* The 1-sphere $\mathbb{S}^1$ (circle) can be constructed via:

(0) $X^{(0)}$ is a single point.

(1) $X^{(1)}$ is created by gluing both ends of an interval to $X^{(0)}$, and this is $\mathbb{S}^1$.

Therefore, one can write $\mathbb{S}^1 = e^{(0)} \cup e^{(1)}$. More concretely, in this case, we can 'decompose' $\mathbb{S}^1$ as $e^{(0)} = \{1\}$ and $e^{(1)} = \mathbb{S}^1 \setminus \{1\} \cong (0,1)$ such that $\sigma^{(1)}(x) = \exp(2\pi i x)$.

Generalizing these, we reach the following definition:

**Definition 47** (CW-Complex)**.** A CW-complex $X$ is a topological space constructed via a filtration of topological spaces

$$\emptyset = X^{(-1)} \subseteq X^{(0)} \subseteq X^{(2)} \subseteq \cdots$$

where $X^{(k)}$ is constructed via some specified attaching maps $\sigma_\alpha^{(k)} \colon \partial e_\alpha^{(k)} \to X^{(k-1)}$ inducing the disjoint union $X^{(k)} = X^{(k-1)} \sqcup_\alpha e_\alpha^{(k)}$ for all $k$-cells $(e_\alpha^{(k)})_{\alpha \in I_k}$ that also identifies $x \in \partial e_\alpha^{(k)}$ and $\sigma_\alpha^{(k)}(x) \in X^{(k-1)}$ for all $x \in \partial e_\alpha^{(k)}$.

### *Cellular Sheaves*

While sheaves are axiomatically defined over topological spaces (see Appendix B for introductions), sheaves can be defined on a partially ordered set (Definition 31) as well. While a poset itself is not a topological space on its own, one can induce a topology on it.

**Definition 48** (Alexandrov Topology)**.** Let $(X, \le)$ be a poset, then the following collection of subsets of $X$:

$$\mathcal{U} := \{ \mathcal{U}_x \subseteq X \mid \mathcal{U}_x = \{ y \in X \mid x \le y \} \}$$

is a topology on $X$, called *Alexandrov topology*.

In other words, $\mathcal{U}_x$ is a neighborhood of $x \in X$ that includes all points $y$ that are adjacent to $x$ in the sense that there is a partial order $x \le y$; that is, $x \in \mathcal{U}_x$ and if $x \le y$ then $y \in \mathcal{U}_x$ as well. In Alexandrov topology, an arbitrary intersection of open sets is open.

We can now formally define cellular sheaves, introduced by Shepard in his thesis [38], by including cell complexes Alexandrov topology, where the partial order is defined in terms of incidence: if a cell $\sigma \in X$ is incident to another cell $\tau \in X$, then $\sigma \le \tau$. The open sets (neighborhood of $\sigma$) in this topology would be $\mathcal{U}_\sigma := \{\tau \in X \mid \sigma \le \tau\}$ for all $\sigma$.

**Definition 49** (Cellular Sheaf [38])**.** Let $\mathcal{C}$ be a category and $X$ be a cell complex. A *cellular sheaf* on $X$ is defined as a covariant functor $\mathcal{F}\colon X \to \mathcal{C}$ that assigns an object $\mathcal{F}(\sigma)$ to each cell $\sigma \in X$ and with the restriction map $\mathcal{F}_{\sigma\to\tau}\colon \mathcal{F}(\sigma) \to \mathcal{F}(\tau)$ for each incident pair of cells $\sigma \subseteq \tau$.

Finally, by interpreting graphs as one-dimensional cell complexes where 0-cells are vertices and 1-cells are edges, we yield sheaves on graphs, which was first introduced and extensively studied by Friedman [39]. For simplicity's sake, we will present the definition of vector-space valued sheaves on undirected graphs below.

**Definition 50** (Sheaves on Graph)**.** A sheaf $\mathcal{F}$ on graph $G = (V, E)$ is a functor that equips each vertex $v \in V$ and edge $e \in E$ with a vector space $\mathcal{F}(v)$ and $\mathcal{F}(e)$ respectively, such that there is a linear map $\mathcal{F}_{v\leq e}\colon \mathcal{F}(v) \to \mathcal{F}(e)$ for all $v$ and $e$ such that $v \leq e$ (i.e. $v \in e$)

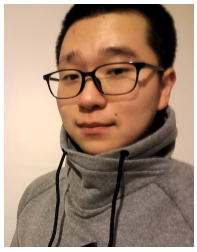
However, sheaves on graphs do not necessarily have to be vector-space-valued to be found useful. Path sheaves (defined in Equations (1) to (4) in Section 2) are an example. While this path sheaf can be 'vector-spacified,' a careful analysis is required as its global sections may then correspond to 'degenerate paths' which are cycles or disjoint unions of cycles. This was shown by Hylton et al. in [8].

## BIOGRAPHY

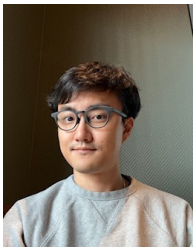
***Alan Hylton*** *would rather work in tube audio, but instead directs Delay Tolerant Networking (DTN) research at NASA GSFC. After studying mathematics at Cleveland State University and Lehigh University, he now advocates for his students and is humbled to work with his powerful and multidisciplinary team by creating venues for mathematicians to work on applied problems.*

***Oliver Chiriac*** *is a Master's student in mathematics at the University of Oxford, with a Bachelor's degree in pure mathematics from the University of Toronto. His primary interests lie at the interface of differential geometry, topology, and network theory. While he spends most of his time listening to music, he also enjoy*

***Jacob Cleveland*** *is a PhD student studying mathematics at Colorado State University. They have a bachelors degree in mathematics from the University of Nebraska at Omaha and a bachelors degree in computer engineering from the University of Nebraska - Lincoln. They joined the Secure Networks, System Integration and Test Branch as a Pathways Intern at NASA Glenn Research Center in 2020. Since joining, they have contributed to several research projects applying pure mathematics to engineering problems in space networking, star tracking, and artificial neural networks.*



***Jihun Hwang*** *(Jimmy) is a Ph.D. student in computer science at Purdue University. He is primarily interested in information-theoretic cryptography and secure multi-party computations, but he ultimately likes to talk about any topics in or related to theoretical computer science, computer networks, and information security. Before Purdue, he studied mathematics and computer science at the University of Massachusetts Amherst.*



***Daniel Koizumi*** *is a first-year Ph.D. student at the University of Texas at Austin and recently received his H.B.S. in Mathematics at the University of Utah. This is his second internship with NASA studying applications of pure mathematics to space communications at Goddard. His interests include approaches involving analysis and algebraic geometry. On a lazy weekend, he enjoys rock climbing, spending time with loved ones, cooking, and, on rare occasions, scuba diving.*

***Karuna Petwe*** *is a fourth year undergraduate student majoring in applied and computational math sciences at the University of Washington. She is broadly interested in understanding the role that mathematical structures and models play in solving current issues in computer and computational sciences. She is excited to participate in ongoing investigations into vital networking capabilities for Delay Tolerant Networks.*

***Tobias Timofeyev*** *is a 4th year Ph.D. student in applied mathematics at the University of Vermont (UVM). His interests are varied, but he particularly enjoys to work at the intersection of network science and dynamical systems theory. He finds it often important to ask how the structure of a system and its function interact. Prior to UVM, he received his B.A. in Mathematics from Bard College. In his free time, he enjoys practicing the cello, and playing boardgames with friends.*