On the Current State of Sheaf Theoretic Networking

Robert Short and Jacob Cleveland
Communications and Intelligent Systems Division
NASA Glenn Research Center
Cleveland, OH, USA
{ robert.s.short, jacob.a.cleveland }@nasa.gov

Zoe Cooperband
Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, PA, USA
zcoop@seas.upenn.edu

Michael Moy
Department of Mathematics
Colorado State University
Fort Collins, CO, USA
michael.moy@colostate.edu

***Abstract*—Advancing the Delay Tolerant Networking (DTN) effort has proven to be a unique and difficult challenge. In the journey to implementing space networking, NASA has considered several strategies. From direct translations of Internet Protocol (IP) to satellite networks to pre-planned routing structures in Contact Graph Routing (CGR), each potential solution for DTN has come with its fair share of setbacks and challenges. Frequently, these challenges can be traced back to specific assumptions made in the development of each protocol that do not carry over from ground networks to space networks.**

To address these fundamental assumptions, a more general and foundational networking theory is required. In this paper, we survey a novel mathematical foundation for networking using the theory of cellular sheaves. Sheaves form a mathematical tool for modeling local phenomenon that leads to global effects. Sheaves have been introduced a number of times, and our goal here is to summarize the applications and point towards important future work that must be done to build a stronger, cohesive theory for networking.

***Index Terms*—Delay Tolerant Networking, sheaves, sheaf theory, networking, networking theory**

I. Introduction

When trying to transition the ideas from networking on Earth to networking in space, we encounter many difficulties. At a fundamental level, the assumptions we can form about connections on Earth are shattered when we move to space. Of particular note, the mobility of satellites and planets leads to connections changing over time and material differences in connection statistics, such as latency. To solve these problems, Delay Tolerant Networking (DTN) efforts have been proposed and studied by NASA and others.

The leading solutions proposed by NASA at the moment are Interplanetary Overlay Network (ION) [1], [2], DTN Marshall Enterprise (DTNME) [3], and High-rate Delay Tolerant Networking (HDTN) [4], [5]. Each of these software solutions finds practical workarounds to bridge the gaps left by these assumptions. One of the primary routing tools utilized is Contact Graph Routing (CGR) [6]–[9]. Some of these workarounds adapt to the needs of specific missions and operational situations, and so form point solutions that do not address general needs or capabilities. Many of these workarounds, especially CGR, are sufficient when it is possible to have a perfect understanding of the full network. However, with the number of satellites around the Earth increasing, and with plans to make humanity into a multi-planet species, it is unreasonable to think that we will always have perfect information about our network connections available to transmit to all satellites. So, the question remains, in a situation where we do not have a perfect understanding of the full network, what can we guarantee in terms of networking capabilities?

As a central example, we can think of the information needed to route data through a network. At the individual satellite level, routing has a straightforward structure. Locally, a routing system must take in data, determine what must be done with that data, and then transmit the data to the next destination. However, when pieced together, a route is a global set of decisions made across a network that is greater than the sum of its parts. Effective routing algorithms stem from the ability to make consistent routing decisions across the network so that data can reach its final destination.

But this "local to global" idea becomes complicated when implemented in real networks. On Earth, the Internet uses hierarchy, subnetworks, and fixed knowledge of connections to handle this issue. As data moves up the hierarchy, the routing protocol used changes to adapt to the needs at each level. So it is the combination and gluing of a variety of routing protocols that enables effective routing in the Internet.

Sheaves are a natural mathematical data structure for understanding local information and how it informs global information. They provide a flexible model for storing data over a space that lends itself to networking. In fact, sheaf theory has been applied to networking for several years now as part of investigations led by Michael Robinson [10]–[12], Rob Ghrist [13], Jakob Hansen [14], Justin Curry [15], and several researchers at NASA [16]–[18]. One aspect of the HDTN project at NASA Glenn Research Center is to lead the charge for researching sheaf theory as a foundational theory of networking; we survey some of the ideas this has produced in Section III.

By reframing networking theory in terms of sheaves, we seek to enable a more general understanding of the structure of networks in space and how they differ from those on Earth. As an example, a routing procedure determined locally by spacecraft must extend to produce global connections across a dynamic network. Sheaves can be used to describe such locally determined routing procedures, providing the right language to describe their global properties and analyze their effectiveness. Furthermore, sheaf theory has tools showing how protocols

Thanks to NASA Glenn Research Center SCaN Internship Program and the High-rate Delay Tolerant Networking project for sponsoring this research.

can be glued together to model hierarchical networking options – we explore these ideas in Section IV.

In this paper, we lay out the advances our group has made so far and outline future efforts into developing a coherent and consistent theory of networking using sheaves.

II. Sheaf theory background

Sheaves are mathematical structures for describing data distributed over a space and how this local data interacts and converges in a global data structure. While sheaves can be defined in general mathematical terms, we will focus on sheaves that record data over graphs, since graphs are commonly used to model networks. These are special cases of *cellular sheaves*; see [14], [15]. Here a graph $G = (V, E)$ will be a set V of vertices and a set E of edges, where each edge is a set of two distinct vertices. We write $v \leq e$ for a vertex v and an edge e if e is incident to v (formally, if e contains v).

Definition 1. A *sheaf* $\mathcal{F}$ over a graph $G = (V, E)$ consists of

- Sets $\mathcal{F}(v)$ and $\mathcal{F}(e)$ for each vertex v and edge e called the *stalks* of $\mathcal{F}$.
- Functions $\mathcal{F}(v \leq e)\colon \mathcal{F}(v) \to \mathcal{F}(e)$ between the stalks over vertices v and edges e if $v \leq e$[1] These functions are called *restriction maps*.

Stalks $\mathcal{F}(v)$ and $\mathcal{F}(e)$ are spaces of possible values that can be stored at given vertices and edges. The restriction maps specify how the values of stalks relate to one another. We choose stalks of sheaves to be sets in Definition 1 to allow for flexibility in modeling a wide variety of phenomena in graph networking. Depending on the application, additional structure on the stalks can be desirable. In this paper we will also consider the case where stalks are vector spaces and restriction maps are linear maps[2].

A. Sections

The modeling of data with sheaves is notable because sheaves come equipped with a notion of *consistency* of data. Specifically, data over a sheaf is consistent if it respects all of the relevant restriction maps. In the language of sheaf theory, an instance of consistent data is called a section.

Definition 2. A *local section* s over a subset[3] $U \subseteq G$ is a choice of elements $s_v \in \mathcal{F}(v)$ and $s_e \in \mathcal{F}(e)$ for each vertex and edge in U such that if $v \leq e$ we have $\mathcal{F}(v \leq e)(s_v) = s_e$, i.e. the choice of elements of stalks respects all restriction maps. A *global section* is a local section over the whole graph, setting $U = G$.

[1]Mathematically, the sheaf $\mathcal{F}$ is a set-valued functor on the poset category of G, ordered by inclusion. The category of sets may be replaced by other (complete) categories. This can be rephrased in topological terms by giving the poset the Alexandrov topology, and then $\mathcal{F}$ can be extended to all open sets in a way that satisfies the sheaf gluing axioms [15].

[2]Additionally, it is common to see sheaves with stalks of rings, groups, or modules in the literature [13], [15], [19]

[3]To match the topological definition of a sheaf, we can require U to be open in the Alexandrov topology.

Global sections are composed of individual points of data at vertices and edges that come together to form cohesive data over the entire graph. Thus, sheaves are often presented as a way to describe the gluing of local data into global data. This notion of data consistency is at the heart of sheaf theory. For instance, researchers in applied sheaf theory have studied methods of generating approximations to global sections and using the notion of data consistency to converge towards solutions; see [12].

While a section describes data over the entire graph, it is "determined locally," meaning it is given by the collection of data at all vertices and edges. This is a key feature of sheaves. In certain cases, the stalks may represent data structures expected to be used in the physical computers the vertices represent, in which case a section represents a consistent choice of instances of these data structures; we explore this perspective in Section IV.

B. Operations on Sheaves

It is often the case that data are related to other data and therefore sheaves are related to other sheaves. To make a mapping or transfer of data precise, we introduce the analog of a function that is used in sheaf theory, a *morphism* of sheaves. The key difference between sheaf morphisms and traditional functions is that sheaf morphisms are additionally required to respect the restriction maps.

Definition 3. For sheaves $\mathcal{F}$ and $\mathcal{G}$ over a graph G, a *sheaf morphism* $\varphi : \mathcal{F} \to \mathcal{G}$ is a collection of functions from the stalks of $\mathcal{F}$ to the stalks of $\mathcal{G}$ respecting restriction maps. Namely, φ consists of functions $\varphi_v : \mathcal{F}(v) \to \mathcal{G}(v)$ and $\varphi_e : \mathcal{F}(e) \to \mathcal{G}(e)$ for every vertex and edge such that $\varphi_e \circ \mathcal{F}(v \leq e) = \mathcal{G}(v \leq e) \circ \varphi_v$ whenever $v \leq e$[4].

This last condition of morphisms is often described as the commutativity of the following diagram for any $v \leq e$. "Commutativity" means that either path of functions taken from $\mathcal{F}(v)$ to $\mathcal{G}(e)$ gives the same result.

$$\begin{array}{ccc} \mathcal{F}(v) & \xrightarrow{\varphi_v} & \mathcal{G}(v) \\ {\scriptstyle \mathcal{F}(v\leq e)}\downarrow & & \downarrow{\scriptstyle \mathcal{G}(v\leq e)} \\ \mathcal{F}(e) & \xrightarrow[\varphi_e]{} & \mathcal{G}(e) \end{array}$$

The commutativity of these diagrams allows sheaf morphisms to transfer local sections and global sections from one sheaf to another. Effectively, sheaf morphisms are not only maps of data, but maps of compatible data. Sheaves and the morphisms between them form an appropriate model of functions on network data structures.

A sheaf morphism transforms the data assigned to a fixed graph. In contrast, the next two operations we describe, the direct image and the inverse image, characterize the transfer of data between separate graphs. First, we define a map between graphs.

[4]Viewing sheaves as functors, a sheaf morphism is a natural transformation.

Definition 4. A *map of graphs* $f\colon G \to H$ is a function from the set of vertices and edges of G to the set of vertices and edges of H satisfying the following condition. For any edge $e = \{v, w\}$ of G, if $f(v) = f(w)$ the edge collapses to $f(e) = f(v) = f(w)$, and otherwise $f(e) = \{f(v), f(w)\}$[5].

To a map of graphs $f : G \to H$, the direct image operation carries a sheaf over G forward along f to H. Oppositely, the inverse image operation pulls back a sheaf from H, backtracking along f to a sheaf over G. These two sheaf operations allow for the transfer of data between separate graphs.

Definition 5. Let $f\colon G \to H$ be a map of graphs and let $\mathcal{F}$ be a sheaf over G. The *direct image* $f_*\mathcal{F}$ is a sheaf over H such that:

- Over vertices, $f_*\mathcal{F}(v)$ is the set of local sections of $\mathcal{F}$ over the subgraph $f^{-1}(v)$[6].
- Over edges, $f_*\mathcal{F}(e)$ is the Cartesian product of stalks $\mathcal{F}(e')$ for all edges $e' \in f^{-1}(e)$ in G.
- Restriction maps $f_*\mathcal{F}(v \leq e)$ send a local section over $f^{-1}(v)$ to the corresponding choice of elements over edges.

Definition 6. Let $f\colon G \to H$ be a map of graphs and let $\mathcal{F}$ be a sheaf over H. The *inverse image* $f^*\mathcal{F}$ is a sheaf over G defined by setting $f^*\mathcal{F}(v) = \mathcal{F}(f(v))$ and $f^*\mathcal{F}(e) = \mathcal{F}(f(e))$ over vertices and edges. The restriction maps are defined as $f^*\mathcal{F}(v \leq e) = \mathcal{F}(f(v) \leq f(e))$.

These abstract operations form a blueprint for transferring, manipulating, and transforming data over graphs. There is a wide degree of freedom in how these methods are applied. In particular, in Section IV, we interpret sheaves as records of the data structures to be used in the computers of a network, and here these operations on sheaves will allow us to describe functions on such data structures and relationships between data structures in different graph models of a network.

III. Networking models using sheaves

Here we survey some of the ways sheaves have been used to model various aspects of networking. We begin with examples in which sections describe simple objects or attributes in a network.

A. The Weight Sheaf

It is common to assign weights to the edges of a graph $G = (V, E)$ to record attributes of the edges such as cost or capacity – this is frequently given by a function $w : E \to \mathbb{R}$. This data can be captured in a relatively simple sheaf. Take the sheaf $\mathcal{W}$ over G that assigns $\mathbb{R}$ to edges and assigns $\mathbb{R}^{\deg(v)}$ to each vertex, where $\deg(v)$ is the number of edges incident to v. The restriction maps simply communicate the expected weights to edges, so a section of $\mathcal{W}$ is locally determined by the weights at each edge and the weights recorded at each vertex.

B. The Path Sheaf

There is also a sheaf that captures the notion of a path through a graph. This is the *path sheaf* $\mathcal{P}$, described in [20], in which each global section indicates a path from a fixed source vertex to a fixed target vertex. Global sections can, however, also contain loops that are disjoint from the path. Thus, global sections of $\mathcal{P}$ indicate subsets of the network that *locally* resemble paths, which is not surprising, as global sections of a sheaf are described by local information at all vertices and edges.

A variation of this sheaf is the *distance path sheaf* $\mathcal{DP}$, which also records the distance from the source at each vertex and edge of the path; global sections of this sheaf also indicate paths. The recording of distance has the effect of eliminating loops as they can no longer maintain consistent distance record-keeping in $\mathcal{DP}$. To connect these, there is a sheaf morphism $\varphi\colon \mathcal{DP} \to \mathcal{P}$ that forgets the information of the distances. Thus, φ sends a global section of $\mathcal{DP}$ to the global section of $\mathcal{P}$ representing the same path.

C. The Multicast Sheaf

Another sheaf, described in [17], models multicast communications by letting vertices specify to which adjacent vertices they plan to send messages. Multicast communications allow a vertex to send to any number of its neighbors; this is more versatile than unicast communications, in which only one neighbor is chosen, or broadcast communications, in which all neighbors are chosen. This sheaf builds upon work in [11], which modeled wireless networks under an assumption of broadcast communications.

Locally, the global sections of a multicast sheaf look like the setup for routing we suggest in the introduction. The ingress takes in information from all incoming edges, the vertex displays a choice of which edges to send to, and then the egress passes those notifications to the outgoing edges. As such, a global section of the multicast sheaf represents a choice of edges to be transmitted along, agreed upon by all vertices.

D. The Delaunay Sheaf

While abstract graphs are good for simple examples, in networking it is often that the graph of interest is embedded in an ambient d-dimensional space, forming a *geometric graph* (typically with $d = 2, 3$). In such a case, each vertex is endowed with a position in $\mathbb{R}^d$, and each graph edge is realized as a straight line between two vertices. As an example, consider the *Delaunay Triangulation* of a set of points $\{v_i\}$ within $\mathbb{R}^d$, see [21]. This forms a geometric graph that is of considerable interest in networking [22].

Delaunay Triangulations are often constructed centrally [23], but one might consider the situation where each vertex v_i in the Delaunay Triangulation represents an *agent* engaging

[5]Viewing graphs as simplicial complexes, this is a simplicial map. This definition maintains that edges of $f(G)$ have distinct vertices. Note that this is not necessarily a graph homomorphism.

[6]Here $f^{-1}(v)$ is the preimage of the vertex v, the set of all vertices and edges of G that are mapped to v by f.

in networking tasks. Two interesting questions one can ask: what information must each agent know to reconstruct the network within a local neighborhood? Moreover, can these local networks be stitched together to accurately reconstruct a global network between all agents? These questions are best framed in the language of cellular sheaves and form a mathematical foundation of distributed Delaunay triangulation protocols [24], [25].

To construct the *Delaunay sheaf* $\mathcal{D}$, assign to each vertex the stalk $\mathbb{R}^d$ and to each edge $e_{ij} = \{v_i, v_j\}$ the stalk $\mathbb{R}^d/\langle v_i - v_j \rangle$, which quotients $\mathbb{R}^d$ by a one-dimensional subspace. The restriction maps $\mathcal{D}(v_i \leq e)$ send a vector of $\mathbb{R}^d$ to its equivalence class in the edge stalk. In a future paper, we will show that global sections of $\mathcal{D}$ provide sufficient information to reconstruct the possible Delaunay Triangulations over a set of vertices in $\mathbb{R}^d$.

At present, this model assumes that agents are stationary. If the agents are moving, the situation is substantially more complicated. Vertices connect and disconnect over time depending on well-formulated rules as seen in the case of the dual construction to the Delaunay Triangulation, the *Voronoi Diagram* [26]. In this setting, it would be fruitful to define a *time-parameterized sheaf* to model the information needed to construct global sections that persist over time. We hypothesize that an agent would have to have a broader knowledge of other vertices two edges away to do so. We further hypothesize that this sheaf framework is flexible enough to define conditions for *network discovery* where new vertices may be introduced to the triangulation. We will investigate this further in a future paper.

IV. Sheaves as network data structures

Mathematically, a sheaf provides a record of types of information over a space, and sections of the sheaf give specific instances of information over the space. This description closely matches the way data is stored in computers across a network, and with the right interpretation, we should expect that sheaves can be used to model the data structures in the computers of a network. This perspective, in which sheaves model data to be physically stored in the network, is more recent and diverges from some previous models such as those described in Section III. Here we introduce this recent perspective, beginning by translating some sheaf theoretic terms into the language of data structures.

Given a sheaf $\mathcal{F}$, we begin with the viewpoint that the stalk $\mathcal{F}(v)$ over each vertex v records a data structure used by the computer or network device represented by v. An instance of this data structure is then an element of the stalk. Data structures over the edges can be thought of as recording data that should agree between vertices, and restriction maps are hypothetical functions on the data structures of the vertices. A global section of $\mathcal{F}$ then consists of instances of the data structures over the vertices that are consistent with the restriction maps. Thus, a sheaf provides a notion of a "network data structure": it is a set of data structures at the vertices subject to consistency requirements across edges.

This perspective is general enough that any realistic description of data stored in the computers of a network could be modeled in this way. As a simple example, we might model a basic set of routing tables in the computers of a network as follows. Define a sheaf $\mathcal{F}$ on a graph G by letting $\mathcal{F}(v)$ be a set of all possible routing tables for each vertex v, where a routing table can be modeled, for instance, as a list of pairs $(\text{destination}, \text{first\_edge})$, with at most one entry per destination. For each edge e, let $\mathcal{F}(e) = \{*\}$, a set with one element, so that each restriction map $\mathcal{F}(v) \to \mathcal{F}(e)$ must be the unique function to $\{*\}$. These restriction maps thus impose no conditions on what routing tables may be stored, so a global section of $\mathcal{F}$ is simply an assignment of a routing table to each vertex. This technique of placing the set $\{*\}$ over each edge[7] can always be used to model data with no required relationships across edges.

Introducing nontrivial restriction maps imposes consistency conditions on the data structures. For instance, if we consider a time-varying network in which edges may only be available at certain times, we can modify the previous example by additionally requiring that each element (routing table) of $\mathcal{F}(v)$ also records a list of time intervals that each edge incident to the vertex v is available. Then for an edge e, let $\mathcal{F}(e)$ be a set of all possible lists of time intervals and let each restriction map $\mathcal{F}(v \leq e)$ send a table in $\mathcal{F}(v)$ to the list of times e is available according to the table. A global section of $\mathcal{F}$ is then an assignment of routing tables such that routing tables at adjacent vertices agree on when their common edge will be available. The construction of this sheaf is similar in spirit to the weight sheaf of Section III.

The power of this approach of modeling data structures with sheaves comes from its compatibility with operations on sheaves. To begin, if sheaves describe data structures on a network, a sheaf morphism describes a function between such data structures. A sheaf morphism will consist of functions at all computers (vertices) and edges, respecting the consistency conditions imposed by restriction maps. That is, a sheaf morphism must turn consistent data into consistent data.

A. Direct and inverse images manage subnetworks

In this interpretation of sheaves as network data structures, direct and inverse images serve as useful tools: they are especially well suited for describing subnetworks. Suppose we have a network modeled as a graph G and a subnetwork modeled as a subgraph $S \subseteq G$. There is an inclusion map $i\colon S \to G$ that sends vertices and edges to themselves in G, and we can consider both direct and inverse images with respect to this map. If $\mathcal{F}$ is a sheaf on S, then the direct image $i_*\mathcal{F}$ is a sheaf on G which takes the same values as $\mathcal{F}$ on those edges and vertices in S and gives a set $\{*\}$ with a single element on those vertices and edges not in S. Thus, $i_*\mathcal{F}$ views data structures over S as data structures over the entire network. On the other hand, if $\mathcal{H}$ is a sheaf on G, then

[7]More generally, we may place a terminal object in a sheaf's target category over an edge, so that the restriction maps impose no conditions across that edge.

we can form the inverse image $i^*\mathcal{H}$, which is a sheaf on S that will take the same values as $\mathcal{H}$ on each vertex and edge in S. Thus, given a data structure over the whole graph, $i^*\mathcal{H}$ restricts attention to the subnetwork S.

For another application of these operations, we can consider a graph G with its vertices partitioned into subnetworks and a second graph G' with a single vertex for each subnetwork and an edge between subnetworks whenever an edge exists between any vertices of those subnetworks in G. This comes with a map $q\colon G \to G'$ that sends a vertex in G to the vertex representing the subnetwork it belongs to in G'. Given a sheaf $\mathcal{H}$ on G', the inverse image $q^*\mathcal{H}$ assigns all vertices in a subnetwork in G to the value of $\mathcal{H}$ at the vertex representing that subnetwork in G'. This can be viewed as distributing the same information to all vertices of a subnetwork according to data over the simplified graph G'. On the other hand, given a sheaf $\mathcal{F}$ on G, the direct image $q_*\mathcal{F}$ is a sheaf on G'. For a vertex v in G' corresponding to a subnetwork $S \subseteq G$, the stalk $q_*\mathcal{F}(v)$ is the set of local sections of $\mathcal{F}$ over S. This means that the data structures described by $q_*\mathcal{F}$ are given by aggregate data over subnetworks of G.

B. Building larger data structures from smaller

Data structures in a single computer are often built from multiple smaller data structures. For instance, an ordered pair of objects is constructed from two objects. This idea extends in a natural way to sheaves representing network data structures. Given two sheaves $\mathcal{F}_1$ and $\mathcal{F}_2$ on a graph G, we can form a product sheaf $\mathcal{F}_1 \times \mathcal{F}_2$ over G, defined on vertices by $(\mathcal{F}_1 \times \mathcal{F}_2)(v) = \mathcal{F}_1(v) \times \mathcal{F}_2(v)$, where the right side is the usual Cartesian product of sets. The value on edges is defined similarly, and the restriction maps are defined as product maps of the restriction maps of $\mathcal{F}_1$ and $\mathcal{F}_2$, that is, they simply apply the restriction maps of $\mathcal{F}_1$ and $\mathcal{F}_2$ componentwise.

This is in fact a special case of a much more general technique for combining sheaves into a larger sheaf. The category theoretic notion of a *limit* of sheaves describes how to combine the data of multiple sheaves, possibly with imposed relationships between them, into a single sheaf containing all their data. This shows that our sheaf theoretic description of network data structures continues to apply as those data structures are combined into larger ones. See [18] for another example of a limit of sheaves in the context of networking and see [27] for the relevant concepts from category theory.

Limits of sheaves also behave well with respect to morphisms. For instance, given sheaves $\mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{H}_1$, and $\mathcal{H}_2$ on a graph G and sheaf morphisms $\varphi_1\colon \mathcal{F}_1 \to \mathcal{H}_1$ and $\varphi_2\colon \mathcal{F}_2 \to \mathcal{H}_2$, we get a morphism $\varphi_1 \times \varphi_2\colon \mathcal{F}_1 \times \mathcal{F}_2 \to \mathcal{H}_1 \times \mathcal{H}_2$. Thus, we can not only combine data structures to obtain a new data structure but can also combine functions on these data structures to obtain a new function[8].

[8]Specifically, given a natural transformation between two diagrams of sheaves, we get a sheaf morphism between the limits of the diagrams; see [27].

C. Modeling network behavior by sheaf morphisms

We now propose a framework for modeling networking protocols using sheaf morphisms, focusing on examples related to routing. The feature of networking that most resembles sheaves is the local decision making: that is, certain networking decisions made by a computer are based on information stored locally in the computer, and the local decisions made by all computers assemble into global behavior. The gluing of local decisions is modeled well by a sheaf, and the use of local information to make local decisions can be modeled by a sheaf morphism.

Thus, networking decisions meeting this description can be modeled as a sheaf morphism $\varphi\colon \mathcal{K} \to \mathcal{B}$, where $\mathcal{K}$ is a "knowledge" sheaf, recording data structures relevant to networking decisions, and $\mathcal{B}$ is a "behavior" sheaf, modeling the behavior of the network. In particular, to model routing, we can let $\mathcal{B}$ assign a vertex v to a set of functions, each of which takes as an input a packet to be sent across the network and outputs a decision of how and when the packet is sent out of v. The choice of function is made by the knowledge of the network, and this is modeled by the component of the sheaf morphism at v.

As an example, let G be a graph and let $\mathcal{K}$ be the sheaf described in Section IV-A, in which a global section was a choice of routing table for each vertex including the information of time intervals that each edge is available. Define a behavior sheaf $\mathcal{B}$ on each vertex v by letting $\mathcal{B}(v)$ be the set of functions taking a packet and its starting time to the first edge adjacent to v it will be sent along, plus the sending time. For each edge e, let $\mathcal{B}(e)$ be the set of possible sets of times the edge is available, and let each restriction map $\mathcal{B}(v \leq e)\colon \mathcal{B}(v) \to \mathcal{B}(e)$ send a function in $\mathcal{B}(v)$ to the set of possible times it may output as a sending time. Then we can define a sheaf morphism $\varphi\colon \mathcal{K} \to \mathcal{B}$ on each vertex v by letting $\varphi(v)$ take a routing table and edge times of $\mathcal{K}(v)$ to a particular function in $\mathcal{B}(v)$: the function that takes in a packet and its starting time and outputs the edge corresponding to that destination in the routing table and gives a sending time of the earliest time the edge is available in the table. On edges, let $\varphi(e)$ be the identity, sending a list of times an edge is available to the same list of times. These functions respect restriction maps and therefore define a sheaf morphism. This morphism defines a simple routing protocol, in which packets are sent according to a stored routing table at the earliest possible time.

We expect that the benefit of this approach is not in the modeling of a single algorithm or protocol, but in the fact that such models behave well under the operations on sheaves we have described. The direct and inverse images allow for modeling of data structures over subnetworks, and these operations in fact are compatible with sheaf morphisms[9]. Limits of sheaves also behave well with respect to morphisms, as described above. In the framework we have described with morphisms modeling network protocols or algorithms, this describes gluing of protocols or algorithms.

[9]Mathematically, the direct and inverse image operations are functors.

V. What comes next?

Given the flexibility of modeling data structures using sheaves, it is reasonable to expect that current network protocols used on Earth could be described in this language. In the near future, it will be helpful to see if the techniques we have described using direct and inverse images and limits of sheaves can describe some of the complex networking behavior that arises from gluing together various protocols across different parts of a network. This could lend insight into how different protocols could be combined in space networks. Different routing algorithms or protocols, such as Contact Graph Routing [8], [9], PRoPHET [28], [29], and the temporal Joswig algorithm described in [30], may each have their place in different parts of a space network, and it will be important to be able to use them in combination. The language of sheaves may also be appropriate to describe the function of DTN as an "overlay network" that interacts with other networks.

In addition, the more direct incorporation and connection of geometric information via the Delaunay sheaf signals a broadening of networking efforts. The Delaunay Triangulation is well understood among computational geometers, and there are significant improvements to the computational complexity of routing algorithms if they are restricted to Delaunay Triangulations. Encoding Delaunay Triangulations as a sheaf enables us to develop a stronger foundation for local algorithms and may even yield a way to prove computational efficiency in these cases. Future work will include studying properties of the Delaunay sheaf in conjunction with the behavior it can enable within routing algorithms.

Finally, a long-term goal of the authors is to apply sheaf theory to the full networking stack, modeling information from the physical layers all the way up to application layers. In the future, we hope to cast the full OSI model as a collection of sheaves with sheaf morphisms and other operations determining the interactions between layers. The details of an OSI stack of sheaves are highly non-trivial, but we believe such a model would demonstrate the full power of a theory of networking based on sheaves.

References

[1] S. Burleigh, *Interplanetary Overlay Network (ION) Design and Operation*, NASA Jet Propulsion Laboratory, 03 2016.

[2] NASA Jet Propulsion Laboratory, "Interplanetary Overlay Network (ION)." [Online]. Available: https://sourceforge.net/projects/ion-dtn/

[3] NASA Marshall Space Flight Center, "DTNME," 2020. [Online]. Available: https://github.com/nasa/DTNME

[4] NASA Glenn Research Center, "High Rate Delay Tolerant Networking (HDTN)," 2022. [Online]. Available: https://www1.grc.nasa.gov/space/scan/acs/tech-studies/dtn/

[5] ——, "HDTN," 2021. [Online]. Available: https://github.com/nasa/hdtn

[6] G. Araniti, N. Bezirgiannidis, E. Birrane, I. Bisio, S. Burleigh, C. Caini, M. Feldmann, M. Marchese, J. Segui, and K. Suzuki, "Contact graph routing in dtn space networks: overview, enhancements and performance," *IEEE Communications Magazine*, vol. 53, no. 3, pp. 38–46, March 2015.

[7] J. Segui, E. Jennings, and S. Burleigh, "Enhancing contact graph routing for delay tolerant space networking," in *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*, 2011, pp. 1–6.

[8] 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, January 2021.

[9] J. A. Fraire, P. Madoery, S. Burleigh, M. Feldmann, J. Finochietto, A. Charif, N. Zergainoh, and R. Velazco, "Assessing Contact Graph Routing Performance and Reliability in Distributed Satellite Constellations," Jul. 2017. [Online]. Available: https://www.hindawi.com/journals/jcnc/2017/2830542/

[10] M. Robinson, "Sheaves are the canonical data structure for sensor integration," *Information Fusion*, vol. 36, pp. 208–224, 2017.

[11] ——, "Modeling wireless network routing using sheaves," *arXiv*, 2016.

[12] ——, "Hunting for foxes with sheaves," *Notices of the American Mathematical Society*, vol. 66, pp. 661–676, May 2019.

[13] R. Ghrist and Y. Hiraoka, "Applications of sheaf cohomology and exact sequences to network coding," *Proc. NOLTA*, 2011.

[14] J. Hansen, "A gentle introduction to sheaves on graphs." [Online]. Available: https://www.jakobhansen.org/publications/gentleintroduction.pdf

[15] J. Curry, "Sheaves, cosheaves and applications," *arXiv*, 2013. [Online]. Available: http://arxiv.org/abs/1303.3255

[16] 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*, 2020, pp. 1–11.

[17] 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 (50100)*, 2021, pp. 1–9.

[18] 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, pp. 1–19.

[19] G. E. Bredon, *Sheaf theory*, 2nd ed., ser. Graduate Texts in Mathematics. Springer-Verlag, 1997, vol. 170.

[20] M. Moy, R. Cardona, R. Green, J. Cleveland, A. Hylton, and R. Short, "Path optimization sheaves," 2020. [Online]. Available: https://arxiv.org/abs/2012.05974

[21] B. Delaunay, "Sur la sph'ere vide," 1934.

[22] T. Midtbø, "Spatial modelling by delaunay networks of two and three dimensions," Ph.D. dissertation, 02 1993.

[23] D. T. Lee and B. J. Schachter, "Two algorithms for constructing a delaunay triangulation," *International Journal of Computer & Information Sciences*, vol. 9, no. 3, p. 219–242, 1980.

[24] G. Simon, M. Steiner, E. Biersack, and F. F. Germany, "Distributed dynamic delaunay triangulation in d-dimensional spaces," *Institut Eurocom Tech. Rep.*, 2005.

[25] D.-Y. Lee and S. S. Lam, "Efficient and accurate protocols for distributed delaunay triangulation under churn," in *2008 IEEE International Conference on Network Protocols*. IEEE, 2008, pp. 124–136.

[26] L. Gerhardalbers, Guibas, J. Mitchell, and Thomasroos, "Voronoi diagrams of moving points," *International Journal of Computational Geometry and Applications*, vol. 08, 11 2011.

[27] E. Riehl, *Category Theory in Context*, ser. Aurora: Dover Modern Math Originals. Dover Publications, 2017. [Online]. Available: https://books.google.com/books?id=6B9MDgAAQBAJ

[28] A. Lindgren, A. Doria, E. Davies, and S. Grasic, "RFC 6693: Probabilistic Routing Protocol for Intermittently Connected Networks," *IETF Network Working Group*, 2012. [Online]. Available: https://tools.ietf.org/html/rfc6693

[29] S. Grasic, E. Davies, A. Lindgren, and A. Doria, "The evolution of a dtn routing protocol - prophetv2," in *Proceedings of the 6th ACM Workshop on Challenged Networks*, ser. CHANTS '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 27–30. [Online]. Available: https://doi.org/10.1145/2030652.2030661

[30] J. Cleveland, A. Hylton, R. Short, B. Mallery, R. Green, J. Curry, D. V. Dabke, and O. Freides, "Introducing tropical geometric approaches to delay tolerant networking optimization," in *2022 IEEE Aerospace Conference*, 2022, pp. 1–11.