Sheaves in Machine Learning

Grégoire Sergeant-Perthuis*

January 15, 2024

Abstract

In these notes, we begin with an overview of how data with graph structures are processed in deep learning, introducing graph neural networks. We discuss their limitations and present how extending graphs to sheaves, processed by sheaf neural networks, can address these limitations. We then discuss different ways to extend graphs to higher-order interactions and the associated sheaves and invariants (cohomology groups). Finally, we show how these ideas can be extended to inference on graphical models to improve interaction modeling under constraints and to provide novel theoretical insights into classical message-passing algorithms.

1 Signals on Graphs and Graph Neural Networks

The presentation of this section is taken from the course slides [12] and Chapter 5 of [9].

Graphs, adjacency matrix, Laplacian, and signals on graphs. Graphs are a flexible and expressive language for representing entities and the relations or interactions between them. Many real-world domains naturally give rise to graph-structured data, for example: knowledge graphs, regulatory and biological networks, scene graphs in vision, code graphs in program analysis, molecules in chemistry, and 3D shapes in computer graphics.

Definition 1 (Graph)**.** A *graph* is a pair $G = (V, E)$ where V is a set of vertices (nodes) and E is a set of edges describing relations between vertices.

- In a *directed graph*, each edge $e \in E$ is an ordered pair $e = (u, v)$ indicating direction, with $u, v \in V$.
- In an *undirected graph*, each edge $e \in E$ is an unordered pair $\{u, v\}$.

*CQSB, Sorbonne Université

A *signal* with the same *feature* space ($\mathbb{R}^n$) on a graph is a collection of vectors $\{ x_v \in \mathbb{R}^n : v \in V \}$, where each x_v represents the feature vector associated with node v.

For this presentation, we will focus on undirected graphs in order to have a graph Laplacian and normalized Laplacian that can be diagonalized. Recall that the adjacency matrix of an undirected graph $G = (V, E)$ is defined as

$$A_{u,v} = \begin{cases} 1 & \text{if } \{u, v\} \in E, \\ 0 & \text{otherwise,} \end{cases}$$

where A is a symmetric $|V| \times |V|$ matrix with entries in $\{0, 1\}$ indicating the presence or absence of an edge between nodes u and v. Since A is symmetric for undirected graphs, the combinatorial Laplacian $\mathcal{L} = D - A$ and the normalized Laplacian $\Delta = I - D^{-1/2}AD^{-1/2}$ are also symmetric and therefore admit real eigenvalues and an orthogonal eigenbasis [9].

Let us denote by $\mathcal{N}(u)$ the set of neighbors of u in the graph (V, E), i.e.

$$\mathcal{N}(u) = \{ v \in V : A_{u,v} \neq 0\}.$$

Graph Neural Networks. Tasks on graphs can take place at different levels of granularity, including: node-level prediction, edge-level prediction, subgraph and community-level tasks, graph-level prediction, graph generation. A typical machine learning workflow on graphs first extracts features at the node, edge, and graph levels, and then learns a model (e.g., SVM, neural network) that maps these features to target labels. Graph representation learning alleviates the need for manual feature engineering by learning informative representations directly from the graph structure and node/edge attributes.

What corresponds to neural networks in deep learning are Graph Neural Networks (GNN). They work as follows, a signal on a graph G, denoted $h^{(0)}$, which, is a collection $(h_v^{(0)} \in \mathbb{R}^{d^{(0)}}, v \in V)$ is the input of the GNN. A GNN consists of stacking $L \in \mathbb{N}$ layers $\text{GNNLay}^{(\ell)}$ for $\ell = 1, \dots, L$. Typically, each layer $\text{GNNLay}^{(\ell)} : \mathbb{R}^{d^{(\ell-1)} \times |V|} \to \mathbb{R}^{d^{(\ell)} \times |V|}$ sends a signal $h^{(\ell-1)}$ on the graph G to an other signal $h^{(\ell)}$ on the same graph G. Each layer is made of two main operations: generating messages that each node sends to its neighbors and aggregating those messages at each node before updating its representation.

The message functions are differentiable and parameterized, allowing them to be learned from data during training. In general, there are two message functions: one for the node sending a message to itself, denoted by MSG_{self}, and another for its neighbouring nodes, denoted by $\text{MSG}_{\text{neigh}}$. These functions do not depend on the identities of the nodes: for each node, the appropriate function is used depending on whether the message originates from the node itself or from one of its neighbours.

The sent messages take the following form:

$$m_u^{(\ell)} = \mathrm{MSG}_{\mathrm{neigh}}\left(h_u^{(\ell-1)}\right) \quad \forall u \in \mathcal{N}(v),$$
$$m_v^{(\ell)} = \mathrm{MSG}_{\mathrm{self}}\left(h_v^{(\ell-1)}\right)$$

Messages from all neighbors and optionally from the node itself are aggregated using a permutation-invariant function:

$$h_v^{(\ell)} = \mathrm{AGG}\left(\left\{ m_u^{(\ell)} : u \in \mathcal{N}(v) \right\},\ m_v^{(\ell)}\right),$$

The aggregation function AGG can also include learnable parameters, as is the case for example when using attention coefficients $\alpha_{u,v}$ to weight the contributions of neighbors in the aggregation. The total layer is

$$\mathrm{GNNLay}^{(\ell)}(h^{(\ell-1)}) = \mathrm{AGG}\left(\left\{ \mathrm{MSG}_{\mathrm{neigh}}\left(h_u^{(\ell-1)}\right) : u \in \mathcal{N}(v) \right\},\ \mathrm{MSG}_{\mathrm{self}}\left(h_v^{(\ell-1)}\right)\right).$$

The basic message-passing update for a Graph Neural Network layer (see [9, Chapter 5.1.3]) can be written as

$$h_v^{(\ell)} = \sigma\left(W_{\mathrm{self}}^{(\ell)} h_v^{(\ell-1)} + W_{\mathrm{neigh}}^{(\ell)} \sum_{u \in \mathcal{N}(v)} h_u^{(\ell-1)} + b^{(\ell)}\right),$$

In this case:

$$m_u^{(\ell)} = \mathrm{MSG}_{\mathrm{neigh}}\left(h_u^{(\ell-1)}\right) = W_{self}^{(\ell)} h_u^{(\ell-1)}, \quad \forall u \in \mathcal{N}(v),$$
$$m_v^{(\ell)} = \mathrm{MSG}_{\mathrm{self}}\left(h_v^{(\ell-1)}\right) = W_{neigh}^{(\ell)} h_v^{(\ell-1)},$$

and,

$$h_v^{(\ell)} = \mathrm{AGG}\left(\left\{ m_u^{(\ell)} : u \in \mathcal{N}(v) \right\},\ m_v^{(\ell)}\right) = \sigma\left(\sum_{u \in \mathcal{N}(v) \cup u} m_u^{(\ell)} + b^{(\ell)}\right)$$

where $W_{self}^{(\ell)}, W_{neigh}^{(\ell)} \in \mathbb{R}^{d^{(\ell)} \times d^{(\ell-1)}}$ and $b^{(\ell)} \in \mathbb{R}^{d^{(\ell)}}$ are respectively the learnable weight matrices and bias term at layer ℓ.

Graph Convolutional Network. One of the most widely used baseline graph neural network models is the *Graph Convolutional Network (GCN)*[9, 13]. The layer-wise message passing update can be written as

$$h_v^{(\ell)} \ = \ \sigma\left(W^{(\ell)} \sum_{u \in \mathcal{N}(v) \cup \{v\}} \frac{h_u^{(\ell-1)}}{\sqrt{|\mathcal{N}(v)|\,|\mathcal{N}(u)|}}\right), \tag{1}$$

where

- $\mathbf{h}_u^{(\ell)}$ is the embedding of node u at layer k,
- $\mathcal{N}(u)$ denotes the set of neighbors of u,
- self-loops $\{u\}$ are included explicitly in the neighborhood,
- the symmetric normalization $\sqrt{|\mathcal{N}(u)|\,|\mathcal{N}(v)|}$ down-weights messages by node degrees,
- $W^{(\ell)} \in \mathbb{R}^{d^{(\ell)} \times d^{(\ell-1)}}$ is a trainable weight matrix,
- $\sigma(\cdot)$ is a pointwise nonlinearity (e.g., ReLU).

In matrix form, the layer-wise message passing update can be rewritten as,

$$H^{(\ell+1)} = \sigma\Big(H^{(\ell)} + D^{-\frac{1}{2}}\, A\, D^{-\frac{1}{2}}\, H^{(\ell)}\, W^{(\ell)}\Big)\,, \tag{2}$$

Let $\tilde{A} = A + \mathbb{I}$ denote the adjacency matrix of the undirected graph G with added self-connections, where $\mathbb{I}$ is the identity matrix. The degree matrix $\tilde{D}$ is the diagonal matrix with

$$\tilde{D}_{ii} = \sum_j \tilde{A}_{ij},$$

i.e., its diagonal entries are given by the row sums of $\tilde{A}$.

To avoid signal amplification, [13] proposed to replace the first propagation rule given by $\mathbb{I} + D^{-\frac{1}{2}}\, A\, D^{-\frac{1}{2}}$ with a renormalized version defined by $\tilde{D}^{-\frac{1}{2}}\, \tilde{A}\, \tilde{D}^{-\frac{1}{2}}$. The total update rule is then;

$$H^{(\ell+1)} = \sigma\Big(\tilde{D}^{-\frac{1}{2}}\, \tilde{A}\, \tilde{D}^{-\frac{1}{2}}\, H^{(\ell)}\, W^{(\ell)}\Big)\,. \tag{3}$$

The issue of stacking many GNN layers. One of the central challenges in designing deep Graph Neural Networks (GNNs) is the phenomenon known as *over-smoothing*.

- Over-smoothing problem: As the number of GNN layers increases, the node embeddings tend to become increasingly similar and ultimately converge to nearly identical values across the graph. This results in representations that are indistinguishable from one another, even for nodes with different structural or feature contexts.
- This is detrimental because in most downstream tasks (e.g., node classification, link prediction) we rely on the ability of embeddings to differentiate nodes based on their local and global context — which the over-smoothing effect undermines.
- Why does over-smoothing happen? At a high level, standard GNNs rely on iterative message passing and aggregation. Each layer updates a node's representation by combining its previous embedding with information from its neighbors. After many such aggregations, the repeated

mixing of signals across the graph causes the representations of nodes, especially within the same connected component, to become homogenized.

- Formally, this effect has been characterized as the exponential convergence of similarity measures between node features as depth increases. Under typical message-passing schemes, the repeated application of local averaging (or low-pass filtering on the graph) drives the node features toward a consensus state, reducing their variance and expressive diversity.

We can understand the phenomenon of over-smoothing in GNNs through the notion of the *receptive field* of a node.

- In a GNN, the embedding of a node at layer k is determined by information aggregated from its *receptive field*, i.e., the set of nodes whose features can influence that node after k message-passing layers, which typically corresponds to its k-hop neighborhood. Each message-passing layer expands this receptive field by one hop.
- If two nodes have highly overlapping receptive fields, then they will aggregate similar information during message passing, and thus their learned embeddings will tend to be highly similar. This similarity increases with depth because more and more neighbors are included in each node's receptive field.
- As we stack many GNN layers, the receptive fields of nodes within a connected component increasingly overlap, eventually encompassing most of the graph. Consequently, the node embeddings become nearly indistinguishable. This is precisely the over-smoothing problem, where representations collapse and lose discriminative power.

2 Sheaf Neural Networks

This section will be base on [17]. Celebrated references are [10, 5, 1].

Traditional GNNs treat node features as vectors in a single vector space and aggregate them using simple operations (sum, mean, max). However, this approach has limitations:

- *Over-smoothing:* Deep GNNs tend to make node representations increasingly similar, losing discriminative power.
- *Heterophily:* When connected nodes have dissimilar features (heterophilic graphs), simple aggregation performs poorly.

Sheaf neural networks address the previously mentioned issues by introducing functors and sheaves over cell complexes, as we will see in the next section. For now, the sheaves that are most amenable are those constructed over graphs. For this section we will limit our attention to sheaves on graphs.

Let us first recall the link between heat diffusion and GCNs, recasting a remark in [5]. Consider a graph with adjacency matrix A, diagonal degree matrix D, normalised graph Laplacian $\Delta := \mathbb{I} - D^{-1/2}AD^{-1/2}$, and an $n \times f$ feature matrix X. We can define the heat diffusion equation and its Euler discretisation with a unit step as follows:

$$\dot{X}(t) = -\Delta X(t)$$
$$X(t+1) = X(t) - \Delta X(t) = (\mathbb{I} - \Delta)X(t).$$

Comparing this with the Graph Convolutional Network model, we observe that GCN is an augmented heat diffusion process with an additional $f \times f$ weight matrix W and a nonlinearity σ:

$$\mathrm{GCN}(X) = \sigma(D^{-1/2}AD^{-1/2}XW) = \sigma((\mathbb{I} - \Delta_0)XW). \tag{4}$$

From this perspective, it is perhaps not surprising that GCN is particularly affected by heterophily and oversmoothing since heat diffusion makes the features of neighbouring nodes increasingly smooth. In what follows, we consider a much more general and powerful family of (sheaf) diffusion processes leading to more expressive sheaf convolutions.

2.1 Cellular sheaves over graphs

In a cellular sheaf, data is associated with both the nodes and edges of a graph, effectively representing local and global structures.

Given an undirected graph $G = (V, E)$, for each node $v \in V$, we associate a vector space or algebraic structure $\mathcal{F}(v)$. Similarly, for each edge $e \in E$, we associate a vector space or algebraic structure, $\mathcal{F}(e)$. $\mathcal{F}(v)$ and $\mathcal{F}(e)$ the local data on the graph.

A restriction map $\mathcal{F}_{v \trianglelefteq e} : \mathcal{F}(v) \to \mathcal{F}(e)$ represents the relationship between local data on nodes and edges for each incident node-edge pair $v \trianglelefteq e$. In this context, the vector spaces $\mathcal{F}(v)$ and $\mathcal{F}(e)$, corresponding to nodes and edges, are called stalks.

For a given sheaf $(G, \mathcal{F})$, the space of 0-cochains $C^0(G, \mathcal{F})$ is defined as the direct sum over the vertex stalks:

$$C^0(G, \mathcal{F}) = \bigoplus_{v \in V} \mathcal{F}(v)$$

By assigning an arbitrary orientation to each edge e, the co-boundary map serves as a linear map associating 0-cochains to 1-cochains by capturing the difference between the data associated with the vertices connected by an edge. Formally, the co-boundary map is defined as:

$$\delta : C^0(G, \mathcal{F}) \to C^1(G, \mathcal{F}) = \delta(x)_e = \mathcal{F}_{v \trianglelefteq e} x_v - \mathcal{F}_{u \trianglelefteq e} x_u$$

The sheaf Laplacian is an operator that maps 0-cochains to 0-cochains:

$$L_{\mathcal{F}} = \delta^T \delta = \sum_{v, u \trianglelefteq e} \mathcal{F}^T_{v \trianglelefteq e}(\mathcal{F}_{v \trianglelefteq e} x_v - \mathcal{F}_{u \trianglelefteq e} x_u) \tag{5}$$

The sheaf Laplacian is a positive semi-definite block matrix. The diagonal blocks are $L_{\mathcal{F}vv} = \sum_{v \trianglelefteq e} \mathcal{F}_{v \trianglelefteq e}^{\top} \mathcal{F}_{v \trianglelefteq e}$, while the non-diagonal blocks $L_{\mathcal{F}vu} = -\mathcal{F}_{v \trianglelefteq e}^{\top} \mathcal{F}_{u \trianglelefteq e}$. Denoting by D the block-diagonal of $L_{\mathcal{F}}$, the normalised sheaf Laplacian is given by $\Delta_{\mathcal{F}} := D^{-1/2} L_{\mathcal{F}} D^{-1/2}$.

For simplicity, we assume that all the stalks have a fixed dimension d. In that case, the sheaf Laplacian is a $nd \times nd$ real matrix, where n is the number of nodes of G. When the vector spaces are set to $\mathbb{R}$ (i.e., $d = 1$) and the linear maps to the identity map over $\mathbb{R}$, the underlying sheaf is trivial and one recovers the well-known $n \times n$ graph Laplacian matrix and its normalised version Δ_0. In general, $\Delta_{\mathcal{F}}$ is preferred to $L_{\mathcal{F}}$ for most practical purposes due to its bounded spectrum and, therefore, we focus on the former. A cochain x is called *harmonic* if $L_{\mathcal{F}} x = 0$ or, equivalently, if $x \in \ker(L_{\mathcal{F}})$. This means harmonic cochains are characterised by zero disagreements along all the edges of the graph, and it is not difficult to see that, in fact, $H^0(G; \mathcal{F})$ and $\ker(L_{\mathcal{F}})$ are isomorphic as vector spaces.

If the maps $\mathcal{F}_{u \trianglelefteq e}$ are invertible, the sheaf Laplacian is called a *connection Laplacian*; this is the case considered in [1], where the maps are orthogonal.

2.2 Neural Sheaf Diffusion

Considering a graph $G = (V, E)$, each individual node $v \in V$ is associated with a d-dimensional feature vector $x_v \in \mathcal{F}(v)$. The individual vectors x_v are column-stacked to create an nd-dimensional vector $x \in C^0(G, \mathcal{F})$. The vectors belonging to $C^0(G, \mathcal{F})$ form the columns of the feature matrix $X \in \mathbb{R}^{(nd) \times f}$.

Sheaf diffusion can then be described as a process that operates on $(G, \mathcal{F})$, controlled, at time t, by the differential equation:

$$X(0) = X, \quad \dot{X}(t) = -\Delta_{\mathcal{F}} X(t)$$

This equation is discretized using the explicit Euler scheme, which employs a unit step size:

$$X(t+1) = X(t) - \Delta_{\mathcal{F}} X(t) = (I_{nd} - \Delta_{\mathcal{F}}) X(t)$$

In the model proposed in [4], the discretization of the above equation is carried out as follows:

$$X(t+1) = X(t) - \sigma(\Delta_{\mathcal{F}(t)} (I_n \otimes W_1^t) X_t W_2^t) \tag{6}$$

In this case, the sheaf $\mathcal{F}(t)$ and the weights $W_1^t \in \mathbb{R}^{d \times d}$ and $W_2^t \in \mathbb{R}^{f_1 \times f_2}$ are time-dependent. This implies that the underlying geometric structure of the graph changes over time.

3 Around Topological Structures

We now discuss two different combinatorial structures: cell complexes and posets, and the associated cohomology groups for their representations.

Definition 2 (Regular Cell Complex [7, Def. 4.1.1])**.** A *regular cell complex* X is a topological space equipped with a partition into pieces $\{X_\sigma\}_{\sigma \in P_X}$, called *cells*, such that the following properties are satisfied:

1. Locally finite: Each point $x \in X$ has an open neighborhood U that intersects only finitely many X_σ.
2. For each $\sigma \in P_X$, the cell X_σ is homeomorphic to $\mathbb{R}^k$ for some k (where $\mathbb{R}^0$ is a point).
3. Axiom of the Frontier: If $\overline{X}_\tau \cap X_\sigma \neq \emptyset$, then $X_\sigma \subseteq \overline{X}_\tau$. In this case, we say that the pair (σ, τ) are *incident* or that X_σ is a *face* of X_τ. The face relation makes the indexing set P_X a poset by declaring $\sigma \leq \tau$ whenever $X_\sigma \subseteq \overline{X}_\tau$.
4. For each $\sigma \in P_X$, the pair $X_\sigma \subseteq \overline{X}_\sigma$ is homeomorphic to the pair $\mathrm{int}(B^k) \subseteq B^k$, i.e. there is a homeomorphism $\varphi : B^k \to \overline{X}_\sigma$ that sends the interior $\mathrm{int}(B^k)$ homeomorphically onto X_σ.

Definition 3 (Cell Complex [7, Def. 4.1.3])**.** A *cell complex* is a topological space X equipped with a partition into pieces $\{X_\sigma\}$ that satisfies the first three axioms of a regular cell complex.

Moreover, we require that when we take the one-point compactification of X, then the cells $\{X_\sigma\} \cup \{\infty\}$ are the cells of a regular cell complex structure on $X \cup \{\infty\}$.

Definition 4 (Cell category [7, Def. 4.1.5])**.** To a cell complex $(X, \{X_\sigma\}_{\sigma \in P_X})$ we can associate a category $\mathbf{Cell}(X; \{X_\sigma\})$, which is the indexing poset P_X viewed as a category. This means that there is one object σ for each X_σ and a unique morphism $\sigma \to \tau$ for each incident pair $X_\sigma \subseteq \overline{X}_\tau$. When there is no risk of confusion, or a cell structure is specified at the beginning, then we will suppress the extra notation and just use $\mathbf{Cell}(X)$ or X.

Definition 5 (Cellular Sheaves)**.** A *cellular sheaf* F valued in a category D on a cell complex X is a functor $F : \mathrm{Cell}(X) \to \mathsf{D}$, i.e. it consists of

- an assignment to each cell X_σ in X of an object $F(\sigma)$ in D,
- and to every pair of incident cells $X_\sigma \subseteq \overline{X}_\tau$ a restriction map $\rho^F_{\sigma,\tau} : F(\sigma) \to F(\tau)$ such that, whenever $\sigma \leq \tau \leq \gamma$,

$$\rho^F_{\sigma,\tau} \circ \rho^F_{\tau,\gamma} = \rho^F_{\sigma,\gamma}$$

Definition 6 ([7, Def. 6.1.7])**.** We write $\sigma \leq_i \tau$ if the difference in the dimensions of the cells σ and τ is i, i.e.

$$\dim(\sigma) - \dim(\tau) = i.$$

Definition 7 (Signed Incidence Relation [7, Def. 6.1.9])**.** A *signed incidence relation* is an assignment to any pair of cells $\sigma, \tau \in X$ of a number $[\sigma : \tau] \in \{0, \pm 1\}$ such that

- if $[\sigma : \tau] \neq 0$, then $\sigma \leq_1 \tau$ (i.e. τ covers σ in the face poset),
- and if γ and τ are any pair of cells, then

$$\sum_{\sigma} [\gamma : \sigma]\,[\sigma : \tau] = 0.$$

Definition 8 ([7, Def. 6.2.1])**.** Given a cellular sheaf $F\colon X \to \mathrm{Vect}$, the category of vector spaces, we define its *compactly supported k–cochains* to be the product of the vector spaces residing over all the k–dimensional cells:

$$C_c^k(X;F) \;=\; \bigoplus_{\sigma_k} F(\sigma_k).$$

These vector spaces are graded components in a complex of vector spaces $C_c^\bullet(X;F)$. The differentials are defined by

$$\delta_c^k \;=\; \sum_{\sigma_k \leq \tau_{k+1}} [\sigma_k : \tau_{k+1}]\,\rho_{\tau,\sigma}.$$

Let us now discuss Partially ordered sets

Definition 9 (Partially Ordered Set (poset))**.** A *partially ordered set*, or *poset*, is a pair $(P, \leq)$ where P is a set and $\leq$ is a binary relation on P that is

1. *reflexive*: for all $x \in P$, $x \leq x$,
2. *antisymmetric*: for all $x, y \in P$, if $x \leq y$ and $y \leq x$ then $x = y$,
3. *transitive*: for all $x, y, z \in P$, if $x \leq y$ and $y \leq z$ then $x \leq z$.

The relation $\leq$ is called a *partial order* on P, and not every pair of elements in P must be comparable under $\leq$.

Let $\mathcal{C}(\mathcal{A})$ denotes the category whose arrows go in the same direction to the relation $\leq$, i.e. we have an arrow $\beta \to \alpha$ whenever $\beta \leq \alpha$. Functors from $\mathcal{C}(\mathcal{A})^{op}$ to Vect, i.e. contravaraint functors, correspond to sheaves on $X_{\mathcal{A}}$ for the lower topology on $\mathcal{A}$ as we will define now.

Definition 10 (Alexandroff's topologies on a poset)**.** P. S. Alexandroff introduced a natural topology on a poset $\mathcal{A}$, given by a basis of open sets called *down-sets* $U_\alpha = \{\beta :\ \beta \leq \alpha\}$, indexed by $\alpha \in \mathcal{A}$; there are sometime denoted as $\alpha \downarrow$ in the poset litterature. The open subsets for the associated topology are the sets $U \subseteq \mathcal{A}$ such that whenever $b \leq a$ with $a \in U$ then $b \in U$; those opens subsets are called the *lower-sets* of $\mathcal{A}$. We will name this topology the lower Alexandroff topology (A-topology) of $\mathcal{A}$, and denote $X_{\mathcal{A}}$ as the topological space obtained in this way.

The Čech cohomology on $X_{\mathcal{A}}$ is one way to define cohomology groups for functors on $\mathcal{A}$ [2]. Another approach is given by the derived cohomology of the limit functor lim , the cohomology groups $H^\bullet(\mathcal{C}(\mathcal{A}); F)$ are defined as the right derived functors $\mathrm{R}^\bullet\mathrm{lim}$ of the section functor of F [8, Definition 11.17, Definition 13.2].

Definition 11 $(H^0(\mathcal{C}(\mathcal{A}); F))$**.** Let $\mathcal{A}$ be a finite poset, and $F : \mathcal{C}(\mathcal{A}) \to \text{Vect}$ a contravariant functor over $\mathcal{C}(\mathcal{A})$. The following short exact sequence defines $\lim F = H^0(\mathcal{C}(\mathcal{A}); F)$,

$$0 \longrightarrow \lim F \longrightarrow \prod_{a \in A} F_a \xrightarrow{\delta_F} \prod_{\substack{a,b \in A \\ a \geq b}} F_b$$

where for any

$$v \in \prod_{\substack{a,b \in A \\ a \geq b}} F_b \quad \text{and} \quad a, b \in A \text{ with } b \leq a,$$

we define

$$\big(\delta_F(v)\big)(a, b) = F_b^a(v_a) - v_b.$$

4 Inference on diagrams in the category of Markov kernels

This is the work presented at ACT 7[19], see also the more general setting [21]. Graphical models are widely used families of probability distributions that capture conditional independence relations between a collection of variables $X_i, i \in I$; celebrated examples are Hidden Markov models, Bayesian Networks [14]. Graphical models are built from directed and undirected graphs $G = (I, A)$ where nodes $i \in I$ are uniquely identified with the variables X_i. Inference in graphical models ultimately boils down to inference for an undirected graphical model, achieved through the Belief Propagation algorithm [25]. Such Inference constitutes a specific instance of variational inference as it revolves around a free energy termed the Bethe free energy [25].Adopting a variational inference perspective for graphical models has facilitated the extension of the Belief Propagation algorithm to encompass broader classes of probability distributions, enabling the accommodation of interactions among more that 2 variables in contrast to traditional graphical models (factor graphs [3]); this is achieved through the introduction of the Kikuchi free energies [24]. Let us denote $\mathbf{Mes}^f, \mathbf{Kern}^f$, the categories with objects finite measurable spaces and respectively with morphisms measurable maps and the second Markov Kernels (stochastic matrices). $\mathbf{Mes}^f$ can be seen as a subcategory of $\mathbf{Kern}^f$. As exhibited in [26, 16, 15], what underlies variational inference for those classes of probability distributions are presheaves from a finite poset to $\mathbf{Mes}^f$ which morphisms are epimorphisms. We will call them the 'graphical' presheaves. Our contribution is to extend the Generalized Belief Propagation [26] to any presheaf from a finite poset to $\mathbf{Kern}^f$. This work is contained in Chapter 9 of [20] and Appendix 1 of unpublished [22], where we consider the more general problem of optimizing a collection of cost functions over a presheaf of signals.

Consider a collection of agents represented by vertices $i \in I$ that can communicate their beliefs to neighboring vertices ∂i through undirected edges $e \in A$.

Each agent has its own representation of its environment, denoted by E_i. They can share their beliefs with neighboring nodes $j \in \partial i$ through a measurable map $f_e^i : E_i \to E_e$. Graphical models and their extensions do not allow us to account for such heterogeneity in the way each agent models their environment. Such setting is better captured by cellular sheaves [6] and applications [11], an important example of which are Sheaf Neural Networks [5], are limited to functors from the poset associated to a graph ($i \leq e \iff i \in e$) to the category of finite vector spaces $\mathbf{Vect}^f$. We are interested in the more general case where beliefs transfer through a hierarchy, i.e. a poset, and we provide an algorithm for inference in such case where Sheaf Neural Networks can't be used; by convention we consider presheaves instead of functors: 'orders' are given top-down. Furthermore, cellular sheaves are restricted to the face poset of a cell complexe and hence don't apply to all hierarchies.

Definition 12 (Graphical presheaves)**.** Let I be a finite set and $\mathcal{A} \subseteq \mathcal{P}(I)$ be a sub-poset of the powerset of I. Let $E_i, i \in I$ are finite sets. For $a \in \mathcal{A}$ $E_a := \prod_{i \in a} E_i$, let $F_a := E_a$, and for $b \subseteq a$, let $F_b^a : E_a \to E_b$ be the projection map from $\prod_{i \in a} E_i$ to $\prod_{i \in b} E_i$. F is called a graphical presheaf from $\mathcal{A}$ to $\mathbf{Mes}^f$.

For $\mathcal{A}$ a finite poset, the 'zeta-operator' of $\mathcal{A}$, denoted ζ, from $\bigoplus_{a \in \mathcal{A}} \mathbb{R}$ to $\bigoplus_{a \in \mathcal{A}} \mathbb{R}$ is defined as, for any $\lambda \in \bigoplus_{a \in \mathcal{A}} \mathbb{R}$ and any $a \in \mathcal{A}$, $\zeta(\lambda)(a) = \sum\limits_{b \leq a} \lambda_b$. ζ is invertible [18], we denote μ its inverse; its matrix expression $(\mu(a,b), b \leq a)$ defines the Möbius function of $\mathcal{A}$. Let F be a presheaf from $\mathcal{A}$ to $\mathbf{Kern}^f$. It induces a presheaf $\tilde{F}$ from $\mathcal{A}$ to $\mathbf{Vect}^f$, where $\tilde{F}_b^a : \mathbb{P}(F_a) \to \mathbb{P}(F_b)$ is the linear map that sends probability distributions $p \in \mathbb{P}(F_a)$ to $F_b^a \circ p$. Following [26], we introduce a free energy $\mathcal{F}(Q) = \sum_{a \in \mathcal{A}} c(a) \left(\mathbb{E}_{Q_a}[H_a] - S(Q_a)\right)$; $c(a) = \sum_{b \geq a} \mu(b,a)$ is the generalization of the inclusion-exclusion formula associated to $\mathcal{A}$. $S(Q_a) = -\sum_{x_a \in F_a} Q_a(x_a) \ln Q_a(x_a)$ is the entropy of Q_a. We propose to solve $\inf_{Q \in \lim \tilde{F}} F_{\text{Bethe}}(Q)$. $\tilde{F}^*$ is the functor obtained by dualizing the morphisms $\tilde{F}_b^a$, i.e. $\tilde{F}_a^{*,b} : \tilde{F}_b^* \to \tilde{F}_a^*$ sends linear maps $l_b : \tilde{F}_b^* \to \mathbb{R}$ to $l_b \circ \tilde{F}_b^a$. We denote its transpose as $\tilde{F}_a^{\dagger,b}$ for the l^2 scalar product on $\mathbb{R}^{E_a}$ and $\mathbb{R}^{E_b}$.

For a functor G from $\mathcal{A}$ to $\mathbb{R}$-vector spaces, we define μ_G as, for any $a \in \mathcal{A}$ and $v \in \bigoplus_{a \in \mathcal{A}} G(a)$, $\mu_G(v)(a) = \sum_{b \leq a} \mu(a,b) G_a^b(v_b)$. Let us define the function $FE(Q) : \prod_{a \in \mathcal{A}} \mathbb{P}(E_a) \to \prod_{a \in \mathcal{A}} \mathbb{R}$ as $FE(Q) = (\mathbb{E}_{Q_a}[H_a] - S_a(Q_a),\ a \in \mathcal{A})$, which sends a collection of probability measures over $\mathcal{A}$ to their Gibbs free energies. For any $Q \in \prod_{a \in \mathcal{A}} \mathbb{P}(E_a)$, let us denote $\mathrm{d}_Q FE$ as the differential of FE at the point Q.

Theorem 1. *Let $\mathcal{A}$ be a finite poset, let F be a presheaf from $\mathcal{A}$ to $\boldsymbol{Kern}^f$. Let $H_a : F_a \to \mathbb{R}$ be a collection of (measurable) functions. The critical points of $\mathcal{F}$ are the $Q \in lim\ \tilde{F}$ such that,*

$$\mu_{\tilde{F}^\dagger} \mathrm{d}_Q FE|_{lim\ \tilde{F}} = 0 \tag{7}$$

The message-passing algorithm we consider is Algorithm 1; it specializes to the General Belief Propagation for Graphical Presheaves. For two elements of

$a, b \in \mathcal{A}$, such that $b \leq a$, two types of messages are considered: top-down messages $m_{a\to b} \in \mathbb{R}^{F_b}$ and bottom-up messages $n_{b\to a} \in \mathbb{R}^{F_a}$.

Algorithm 1: Message passage algorithm for presheaves from $\mathcal{A}$ to $\mathbf{Kern}^f$

Data: Initialization: $(m^0_{a\to b} \in \tilde{F}_b, b, a \in \mathcal{A}$ s.t. $b \leq a)$, poset $\mathcal{A}$, a presheaf $F : \mathcal{A} \to \mathbf{Kern}^f$;

for $t \leq T$ **do**
  for $a \in \mathcal{A}, b \in \mathcal{A}$ *such that* $b \leq a$ **do**
    $\forall \omega_a \in F_a$
      $n_{b\to a}(\omega_a) \leftarrow \prod_{\substack{c: b\leq c \\ c \not\leq a}} \exp\left(\sum_{\omega'_b \in F_b} \ln m_{c\to b}(\omega'_b) \cdot F^a_b(\omega'_b|\omega_a)\right)$
  end
  for $a \in \mathcal{A}, b \in \mathcal{A}$ *such that* $b \leq a$ **do**
    $b_a = e^{-H_a} \prod_{\substack{b\in\mathcal{A}: \\ b\leq a}} n_{b\to a}$
    $p_a = \frac{b_a}{\sum_{\omega_a} b_a(\omega_a)}$
    $m_{a\to b} \leftarrow m_{a\to b} \cdot \frac{\tilde{F}^b_a(p_b)}{p_a}$
  end
end

A criterion to stop the algorithm is when the beliefs do not change, i.e., when $p^{t+1}_a \approx p^t_a$. The fixed points of the previous message-passing algorithm correspond to critical points of $\mathcal{F}$ over $\lim F$ (Corollary of Theorem 2.2 [22]v2).

References

[1] Federico Barbero, Cristian Bodnar, Haitz Sáez de Ocáriz Borde, Michael Bronstein, Petar Veličković, and Pietro Liò. Sheaf neural networks with connection laplacians. In *Topological, Algebraic and Geometric Learning Workshops 2022*, pages 28–36. PMLR, 2022.

[2] Daniel Bennequin, Olivier Peltre, Grégoire Sergeant-Perthuis, and Juan Pablo Vigneaux. Extra-fine sheaves and interaction decompositions, 2020.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[4] Cristian Bodnar, Francesco Di Giovanni, Benjamin Paul Chamberlain, Pietro Liò, and Michael M. Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns, 2022.

[5] Cristian Bodnar, Francesco Di Giovanni, Benjamin Paul Chamberlain, Pietro Lio, and Michael M. Bronstein. Neural sheaf diffusion: A topological

perspective on heterophily and oversmoothing in GNNs. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[6] Justin Curry. *Sheaves, cosheaves and applications*. PhD thesis, The University of Pennsylvania, 2013. arXiv:1303.3255.

[7] Justin Michael Curry. *Sheaves, cosheaves and applications*. University of Pennsylvania, 2014.

[8] Jean Gallier and Jocelyn Quaintance. *Homology, cohomology, and sheaf cohomology for algebraic topology, algebraic geometry, and differential geometry*. World Scientific, 2022.

[9] William L Hamilton. *Graph representation learning*. Morgan & Claypool Publishers, 2020.

[10] Jakob Hansen and Thomas Gebhart. Sheaf neural networks. *arXiv preprint arXiv:2012.06333*, 2020.

[11] Jakob Hansen and Robert Ghrist. Distributed optimization with sheaf homological constraints. In *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 565–571, 2019.

[12] Jure Leskovec. CS224W: Machine Learning with Graphs. `https://web.stanford.edu/class/cs224w/`, 2025. Course website, Stanford University.

[13] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[14] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.

[15] Olivier Peltre. A homological approach to belief propagation and Bethe approximations. In *International Conference on Geometric Science of Information*, pages 218–227. Springer, 2019.

[16] Olivier Peltre. Message passing algorithms and homology, 2020. Ph.D. thesis, Link.

[17] Antonio Purificato, Giulia Cassarà, Federico Siciliano, Pietro Liò, and Fabrizio Silvestri. Sheaf4rec: Sheaf neural networks for graph-based recommender systems. *ACM Transactions on Recommender Systems*, 4(2):1–26, 2025.

[18] Gian-Carlo Rota. On the foundations of combinatorial theory I. Theory of Möbius functions. *Probability theory and related fields*, 2(4):340–368, 1964.

[19] Grégoire Sergeant-Perthuis and Nils Ruet. Inference on diagrams in the category of Markov kernels (Extended abstract). In *7th International Conference on Applied Category Theory (ACT 7, 2024)*, Oxford (UK), United Kingdom, June 2024. David Jaz Myers and Michael Johnso.

[20] Grégoire Sergeant-Perthuis. *Intersection property, interaction decomposition, regionalized optimization and applications.* PhD thesis, Université de Paris, 2021. Link.

[21] Grégoire Sergeant-Perthuis. Regionalized optimization, 2022.

[22] Grégoire Sergeant-Perthuis. Regionalized optimization, 2022. `https://arxiv.org/abs/2201.11876`.

[23] Grégoire Sergeant-Perthuis, Toby St Clere Smithe, and Léo Boitel. On the functoriality of belief propagation algorithms on finite partially ordered sets, 2025.

[24] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'00, page 668–674, Cambridge, MA, USA, 2000. MIT Press.

[25] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. *Understanding belief propagation and its generalizations*, page 239–269. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[26] J.S. Yedidia, W.T. Freeman, and Y. Weiss. Constructing Free-Energy Approximations and Generalized Belief Propagation Algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312, July 2005.

A Sheaves and Belief Propagation

The next section is taken from [23, Section 2.3].

The General Belief Propagation algorithm is used to find the critical points of the Generalized Bethe Free Energy. A classical result states that the fixed points of this algorithm correspond to the critical points of that free energy, which we state in Proposition 1. Let us now recall the expression of this algorithm following [26, 16].

Let I be a finite set that serves as index for variables $(X_i \in E_i; i \in I)$, each of which takes values in a finite set E_i; let $\mathcal{A} \subseteq \mathcal{P}(I)$ be a collection of subsets of I and denote F the associated graphical presheaf. Let $(H_a : E_a \to \mathbb{R}, a \in \mathcal{A})$ be a collection of Hamiltonians. Let us denote the update rule of the General Belief Propagation algorithm as BP. BP acts on messages that we will now define. In the classical presentation of the algorithm, there are two types of messages at each time $t \in \mathbb{N}^*$. For elements $a, b \in \mathcal{A}$ such that $b \subseteq a$, we have top-down messages $m_{a\to b} \in \mathbb{R}_{>0}^{E_b}$ and bottom-up messages $n_{b\to a} \in \mathbb{R}_{>0}^{E_b}$.

These messages are related as follows:

$$\forall a, b \in \mathcal{A}, \text{s.t. } b \subseteq a, \quad n_{b\to a}^t = \prod_{\substack{c: b\subseteq c \\ c\not\subseteq a}} m_{c\to b}^t \tag{8}$$

Beliefs, which are interpreted as probability distributions up to a multiplicative constant and correspond to candidate sections of F, are defined as follows:

$$\forall a \in \mathcal{A}, \forall x_a \in E_a \quad b_a^t(x_a) \propto e^{-H_a(x_a)} \prod_{\substack{b\in\mathcal{A}: \\ b\leq a}} n_{b\to a}^t(x_b) \tag{9}$$

where $\propto$ stands for proportional to. The multiplication of function $n_{b\to a}$ that have different domains is made possible because there is an the embedding of $\mathbb{R}^{E_b}$ into $\mathbb{R}^{E_a}$ implicitly implied in the last equation; indeed, for $x \in E_a$ and $f \in E_b$, $f : x \mapsto f(x_b)$ defines a function from E_a to $\mathbb{R}$.

To clarify the presentation, we require that b_a be a probability distribution and normalize it accordingly. The update rule is given by,

$$\forall a, b \in \mathcal{A}, \text{s.t.} b \leq a \quad m_{a\to b}^{t+1}(x_b) = m_{a\to b}^t(x_b) \frac{\sum_{y_a : F_b^a(y_a)=x_b} b_a^t(y_a)}{b_b^t(x_b)} \tag{10}$$

One observes that in the previous Equation 10, any normalization of beliefs does not change the update rule. The update rule can be rewritten in a more condensed manner, updating only the top-down messages, for all $a, b \in \mathcal{A}$, such that $b \leq a$,

$$m_{a\to b}^{t+1}(x_b) = m_{a\to b}^t(x_b) \frac{\sum_{y_a : F_b^a(y_a)=x_b} e^{-H_a(y_a)} \prod_{\substack{c\in\mathcal{A}: \\ c\subseteq a}} \prod_{\substack{d: c\subseteq d \\ d\not\subseteq a}} m_{d\to c}^t(x_c)}{e^{-H_b(x_b)} \prod_{\substack{c\in\mathcal{A}: \\ c\subseteq b}} \prod_{\substack{d: c\subseteq d \\ d\not\subseteq b}} m_{d\to c}^t(x_c)} \tag{11}$$

We denote the collection $(m_{a\to b}; a, b \in \mathcal{A}, b \leq a)$ as m. We denote BP : $\bigoplus_{a,b:b\subseteq a} \mathbb{R}^{E_b} \to \bigoplus_{a,b:b\subseteq a} \mathbb{R}^{E_b}$ as the operator underlying the update rule of Equation 11, i.e., we define $\mathrm{BP}(m^t) = m^{t+1}$.

Consider the collection $(C_{a\to b}m_{a\to b}; a, b \in \mathcal{A}, b \leq a)$, where $C_{a\to b}$ is a strictly positive constant, i.e., it does not depend on $x_b \in F(b)$. Then, there is a collection of constants $(C'_{a\to b} > 0; a, b \in \mathcal{A}, b \leq a)$ such that

$$\mathrm{BP}(C_{a\to b}m_{a\to b}) = C'_{a\to b}\mathrm{BP}(m_{a\to b}).$$

Furthermore, the associated beliefs defined by Equation 9 remain unchanged under multiplication of $m_{a\to b}$ by a constant $C_{a\to b}$ for all $a, b \in \mathcal{A}$ such that $b \leq a$.

Therefore, BP is an algorithm that preserves the equivalence classes $\{C \cdot m\}$, i.e., it is defined by the relation $m \sim m'$ whenever there is a collection of scalars $(C_{a\to b} \neq 0; a, b \in \mathcal{A}, b \leq a)$ such that

$$m_{a\to b} = C_{a\to b}m'_{a\to b} \quad \text{for any } a, b \in \mathcal{A} \text{ with } b \leq a.$$

We shall denote the equivalence class of m as $[m]$. The action of BP on the equivalence classes of messages is denoted by [BP] and defined as $[\mathrm{BP}]([m]) = [\mathrm{BP}(m)]$.

Proposition 1 (Yedidia, Freeman, Weiss, Peltre)**.** *Let I be an finite set, and $\mathcal{A} \subseteq \mathcal{P}(I)$ a collection of subsets of I; let F be a graphical presheaf. Let $(m_{a\to b} \in \mathbb{R}^{E_b}_{>0}, a, b \in \mathcal{A} s.t.\ b \subseteq a)$ be a fix point of the Generalized Belief Propagation up to a multiplicative constant, i.e. $[m] = [BP]([m])$. Let $(b_a, a \in \mathcal{A})$ be the associated beliefs normalized so that each $b_a \in \mathbb{P}(E_a)$ (Equation 9). Then $(b_a, a \in \mathcal{A})$ is a critical point of F_{Bethe} under the constraint that $p \in \lim \mathbb{P}F$. Furthermore, any critical point of F_{Bethe} in $\lim \mathbb{P}F$ is a belief associated to a fixed point of BP.*

Proof. See Theorem 5.15 in [16] or Theorem 5 in [26]. This result is also a corollary of Theorem 2.2. [22]. □

An extension of the Belief Propagation algorithm to presheaves from a finite poset taking values in finite sets can be found in [22]. In their work, for each presheaf F, they propose a message-passing algorithm, that we will denote as MP, whose set of fixed points corresponds to the critical points of the Generalized Bethe Free Energy. When considering graphical presheaves, these message-passing algorithms slightly differ from the Belief Propagation algorithm but have the same fixed points, see for a description of the shared properties of the two algorithms. However, as we will show, MP behaves well under natural transformations $\phi : F \to G$, whereas BP does not. Let us now recall their message-passing algorithm, MP. To do so, following [22], we need to introduce the elementary operators from which MP is built.

For a functor G from $\mathcal{A}$ to $\mathbb{R}$-vector spaces, we define μ_G as, for any $a \in \mathcal{A}$ and $v \in \bigoplus_{a\in\mathcal{A}} G_a$, $\mu_G(v)(a) = \sum_{b\leq a} \mu(a, b)G^b_a(v_b)$. Let F be a presheaf from a finite poset $\mathcal{A}$ to finite sets **FinSet**. Let us call $\mathrm{FE} : \prod_{a\in\mathcal{A}} \mathbb{R}^{E_a} \to \prod_{a\in\mathcal{A}} \mathbb{R}$ the extension of F_{Bethe} to real valued functions, i.e.

$$\forall a \in \mathcal{A}, \mathrm{FE}(h)(a) = \sum_{x_a \in F_a} h_a(x_a) H_a(x_a) + \sum_{x_a \in F_a} h_a(x_a) \ln h_a(x_a) \tag{12}$$

Let us denote $b \leq a$ as $a \to b$. Let $\delta_F : \bigoplus_{a \in \mathcal{A}} F_a \to \bigoplus_{a,b \in \mathcal{A}: b \leq a} F_b$ be defined as, for $(v_a \in F_a, a \in \mathcal{A})$:

$$\forall a, b \in \mathcal{A}, \text{s.t. } b \leq a, \quad \delta_F(v)(a \to b) = F_b^a(v_a) - v_b \tag{13}$$

For a (covariant) functor G, $d_G : \bigoplus_{a,b \in \mathcal{A}: b \leq a} G(b) \to \bigoplus_{a \in \mathcal{A}} G_a$ is defined for $(v_{a \to b}, a, b \in \mathcal{A}$ such that $a \geq b)$ as

$$d_G(v)(a) = \sum_{b: b \leq a} G_a^b(v_{a \to b}) - \sum_{b: a \leq b} v_{b \to a}. \tag{14}$$

Here, G will be either F^* or $F^\dagger$ when each space F_a is equipped with a scalar product $\langle \cdot, \cdot \rangle_a$.

The ζ function of a functor G plays an import role in the Belief propagation algorithm. $\zeta_G : \bigoplus_{a \in \mathcal{A}} G_a \to \bigoplus_{a \in \mathcal{A}} G_a$ is defined as, for $v \in \bigoplus_{a \in \mathcal{A}} G_a$,

$$\zeta_G(v)(a) = \sum_{b \leq a} G_a^b(v_b) \tag{15}$$

Proposition 2. *Let F be a presheaf from a finite to finite sets. Let $H_a \in \mathbb{R}^{F_a}, a \in \mathcal{A}$, be a collection of Hamiltonians. The differential $dFE : \bigoplus_{a \in \mathcal{A}} \widetilde{F}_a \to \bigoplus_{a \in \mathcal{A}} \widetilde{F}_a^*$ that sends h to $d_h FE$ is invertible. We will denote its inverse by $g_H : \bigoplus_{a \in \mathcal{A}} \widetilde{F}_a \to \bigoplus_{a \in \mathcal{A}} \widetilde{F}_a$.*

Proof. Recall that $\mathrm{FE}_a(h_a) = \sum_{x_a} h_a(x_a) H_a(x_a) + \sum_{x_a} h_a(x_a) \ln h_a(x_a)$, where $h \in \mathbb{R}^{E_a}$, and d_x denotes the differential of the function at point x. Therefore,

$$\mathrm{d}_h \mathrm{FE}_a = \sum_{x_a} \mathrm{d}h_a(x_a)(H_a(x_a) + \ln h_a(x_a) + 1). \tag{16}$$

with $\mathrm{d}h_a(x_a) \in \bigoplus \widetilde{F}_a^*$ being the linear form acting as $(h_b(x_b); b \in \mathcal{A}, x_b \in F_b) \mapsto h_a(x_a)$. Consider the scalar products $\langle h_a, h_a' \rangle_a = \sum_{x_a \in F_a} h_a(x_a) h_a'(x_a)$ for each $a \in \mathcal{A}$; denote $y_a \in \widetilde{F}_a$ the identification of $d_h \mathrm{FE}_a \in F_a^*$ in F_a; then the reformulation of Equation 16 is the following,

$$\forall x_a \in F_a, y_a(x_a) = H_a(x_a) + \ln h_a(x_a) + 1$$

The previous equation is equivalent to

$$\forall x_a \in F_a, h_a(x_a) = e^{-H_a(x_a) + y_a(x_a) - 1}$$

Therefore, we define

$$\forall a \in \mathcal{A}, \forall x_a \in E_a, \quad g_{H_a,a}^+(l)(x_a) = e^{-H_a(x_a) + l_a(x_a) - 1}. \tag{17}$$

The function $g_H = (g_{H_a,a}; a \in \mathcal{A})$ is the inverse of dFE when the image of dFE, i.e., $\bigoplus_{a\in\mathcal{A}} \widetilde{F}_a^*$, is identified with $\bigoplus_{a\in\mathcal{A}} \widetilde{F}_a$ through the scalar product $\langle h, h_1\rangle = \sum_{a\in\mathcal{A}} \langle h_a, h_{1,a}\rangle_a$.

□

Remark 1. Any other choice of scalar product $\langle\cdot,\cdot\rangle_1$ on $\widetilde{F}_a$ induces another map $g_a^{\langle\cdot,\cdot\rangle_1} : \widetilde{F}_a \to \widetilde{F}_a$ defined as $g_a \circ f_{\langle\cdot,\cdot\rangle} \circ f^{-1}_{\langle\cdot,\cdot\rangle_1}$, where $\langle\cdot,\cdot\rangle$ is given by $\langle h, h_1\rangle = \sum_{x_a\in E_a} h(x_a)h_1(x_a)$. The way the inverse $g_H = \widetilde{F}_a^* \to \widetilde{F}_a$ of Proposition 2 is represented as a function $g_H^{\langle\cdot,\cdot\rangle}$ depends on the choice of scalar product on F_a. For the scalar products $\langle h_a, h_a^{'}\rangle = \sum_{x_a} h_a(x_a)h_a^{'}(x_a)$, we denote the identification $f^{\langle\cdot,\cdot\rangle} : \widetilde{F}_a^* \to \widetilde{F}_a$ as f_+ and the associated inverse map as g_H^+.

In what follows, we denote $\bigoplus_{a,b\,:\,b\leq a} F_b$ as $F_\to$ to simplify the notations.

Definition 13 (Message passing algorithms [22])**.** Choose for each $a \in \mathcal{A}$ a scalar product $\langle\cdot,\cdot\rangle_a$ on $\widetilde{F}_a$; equip $F_\to$ with the following scalar product: $\langle l, l_1\rangle = \sum_{b,a\,:\,b\leq a}\langle l_{a\to b}, l_{1,a\to b}\rangle_b$. The message-passing algorithm $\mathrm{MP}_{F,H}$ for inference on a presheaf F from a finite poset $\mathcal{A}$ to finite sets, with Hamiltonians $(H_a \in \mathbb{R}^{E_a}; a \in \mathcal{A})$, is defined as follows:

$$\forall l \in F_\to, \quad \mathrm{MP}_{F,H}(l) = l + \delta_F \circ g_H^{\langle\cdot,\cdot\rangle} \circ \zeta_{F^\dagger} \circ d_{F^\dagger}(l) \tag{18}$$

The algorithm starts with a random initialization of messages $l_0 \in F_\to$, and l^{t+1} is updated into the value $\mathrm{MP}_{F,H}(l^t)$. We will denote $\Delta MP_{F,H}$ the increment $\delta_F \circ g_H \circ \zeta_{F^\dagger} \circ d_{F^\dagger}$.

Proposition 3 (Sergeant-Perthuis)**.** *Let F be a presheaf from a poset $\mathcal{A}$ taking values in finite sets. Let $(H_a \in \mathbb{R}^{E_a}; a \in \mathcal{A})$ be a collection of Hamiltonians. Let $(l_{a\to b} \in \widetilde{F}_b^*; a, b \in \mathcal{A} : b \leq a)$ be a fixed point of $MP_{F,H}$, i.e., $MP_{F,H}(l) = l$. Let $b_a \in \mathbb{P}(F_a)$ be the unique probability distribution such that*

$$b_a \propto g_H \circ \zeta_{F^*} \circ d_F(l).$$

Then $(b_a, a \in \mathcal{A})$ is a critical point of F_{Bethe} under the constraint that $p \in \lim \mathbb{P}F$. Furthermore, any critical point of F_{Bethe} in $\lim \mathbb{P}F$ is a belief associated to a fixed point of $MP_{F,H}$.