

An Algebraic Approach to Moralisation and Triangulation of Probabilistic Graphical Models

Antonio Lorenzin ✉ 🏠 

Computer Science Department, University College London, UK

Fabio Zanasi ✉ 🏠 

Computer Science Department, University College London, UK

Abstract

Moralisation and Triangulation are transformations allowing to switch between different ways of factoring a probability distribution into a graphical model. Moralisation allows to view a Bayesian network (a directed model) as a Markov network (an undirected model), whereas triangulation works in the opposite direction. We present a categorical framework where these transformations are modelled as functors between a category of Bayesian networks and one of Markov networks. The two kinds of network (the objects of these categories) are themselves represented as functors, from a “syntax” domain to a “semantics” codomain. Notably, moralisation and triangulation are definable inductively on such syntax, and operate as a form of functor pre-composition. This approach introduces a modular, algebraic perspective in the theory of probabilistic graphical models.

2012 ACM Subject Classification Mathematics of computing → Probabilistic representations

Keywords and phrases Functorial Semantics, Probabilistic Model, Bayesian Network

Digital Object Identifier 10.4230/LIPIcs.CALCO.2025.6

Related Version *Full Version*: <https://arxiv.org/abs/2503.11820>

Funding Our work has been supported by the ARIA Safeguarded AI TA1.1 programme.

1 Introduction

Increasingly in recent years, category-theoretic semantics has been adopted to identify the algebraic structures underpinning probabilistic computation. These studies have interleaved with the theory of probabilistic programming, Bayesian inference, and machine learning, see e.g. [3, 17, 21] for an overview. The overarching goal is to establish a principled, mathematically rigorous semantics of probabilistic reasoning, bringing formal clarity to traditional approaches. One notable example is the work [11], which employs categorical methods to classify different way of reasoning with soft evidence. A key strength of category theory in this context is its emphasis on abstraction: the categorical perspective enables a unified treatment of different probabilistic frameworks – such as discrete, measure-theoretic, and Gaussian models – allowing for generic definitions of conditioning, independence, marginals, and related concepts. A prominent programme in this direction is that of *Markov categories* [9]. Another major advantage of categorical methods is their inherent compositionality. By leveraging categorical structures, we gain insight into how the meaning of probabilistic computation can be systematically decomposed into more fundamental components.

The focus of this work is the categorical semantics of probabilistic graphical models (PGMs). PGMs provide a structured representation of probability distributions, making them useful for a variety of tasks in machine learning, statistics, and artificial intelligence. The “structure” is expressed by a combinatorial object (usually a graph), whose vertices represent events, and edges capture probabilistic dependencies between them. Two of the most prevalent examples of PGMs are Bayesian networks, which are built on directed acyclic graphs, and Markov networks (also called Markov random fields), based on undirected



© Antonio Lorenzin and Fabio Zanasi;

licensed under Creative Commons License CC-BY 4.0

11th Conference on Algebra and Coalgebra in Computer Science (CALCO 2025).

Editors: Corina Cirstea and Alexander Knapp; Article No. 6; pp. 6:1–6:21

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

graphs. These models have orthogonal expressivity, and shine in different applications. Therefore, the theory of PGMs comprises procedures transforming a Bayesian network into a Markov network, called *moralisation*, and a Markov network into a Bayesian network, called *triangulation*. Importantly, these transformations do not introduce new conditional independencies between events, but may lose some of those expressed by the original network. Moralisation and triangulation ubiquitously appear in a variety of tasks in Bayesian reasoning, such as the junction tree algorithm, clique tree message passing, variable elimination, and graph-based optimisation – see [15, 2] for an overview.

In traditional presentations of PGMs, the combinatorial representation of dependencies between variables is deeply entwined with their probabilistic meaning, variously expressed in terms of conditional probability tables (for Bayesian networks), or factors (for Markov networks). This heterogeneity reflects on the way moralisation and triangulation are defined: even though they exclusively affect the combinatorial structure of the network, strictly speaking they need to be applied to the whole network, including the probabilistic component. Moreover, formal methods for reasoning about the combinatorial structure are limited, compared to e.g. algebraically/syntactically defined objects, which hampers a systematic study of the mathematical properties of moralisation and triangulation.

In this paper, we develop a categorical semantics of Bayesian networks, Markov networks, and the translating operations of moralisation and triangulation. Our approach draws inspiration from Lawvere’s framework of functorial semantics [16], which understands an algebraic theory T via its freely generated cartesian category of terms \mathbf{FreeC}_T , and a model of T as product-preserving functors $\mathbf{FreeC}_T \rightarrow \mathbf{C}$ to a cartesian category \mathbf{C} . While Lawvere’s framework identifies cartesianity (the existence of finite products) as the core categorical structure of algebraic theories, categorical approaches to Bayesian networks, beginning with [7, 13, 12], have identified in *copy-delete (CD) categories* [6, 7, 13, 10] the basic structure for expressing the connectivity of DAGs. In particular, [12] shows a 1-to-1 correspondence between Bayesian networks on a DAG \mathcal{G} and structure-preserving functors from $\mathbf{CDSyn}_{\mathcal{G}}$, the free CD-category built from the data of \mathcal{G} , to $\mathbf{FinStoch}$, the CD-category of stochastic matrices. In other words, Bayesian networks are the models of the “theory” \mathcal{G} in $\mathbf{FinStoch}$. Also, morphisms of $\mathbf{CDSyn}_{\mathcal{G}}$ are represented as *string diagrams* [20, 19], to emphasise their interpretation as a two-dimensional syntax for graphs.

The functorial semantics perspective on Bayesian network is the starting point of our approach. We will expand it to encompass Markov networks, moralisation, and triangulation. At a conceptual level, our aims are two-fold:

- By regarding both Bayesian and Markov networks as models of a freely generated category into a category of stochastic processes, we cleanly separate their “syntax” (the combinatorial structure) from their “semantics” (the probabilistic interpretation). This allows to reason separately on each component, enabling a deeper understanding of the algebraic structures underpinning probabilistic graphical models.
- Moralisation and triangulation become examples of this paradigm. We are able to interpret them as transformations acting on the syntax of the network, defined simply by functor pre-composition. Furthermore, being functors from a freely generated category, they can be defined inductively on the syntax generators. In this way, we have reduced the traditional *combinatorial* perspective on networks to a purely *syntactic* perspective, which is more amenable to modularisation and algebraic reasoning.

We now outline our contributions, also indicating how they appear in the paper structure.

- After recalling copy-delete and hypergraph categories (Section 2), in Section 3 we characterise Bayesian and Markov networks as functors. For Markov networks, we give a functorial semantics characterisation analogous to the one of Bayesian networks described above. The key difference is that, rather than CD-categories, we need the richer structure of hypergraph categories [5, 8, 18] in order to account for the lack of directionality of Markov networks. We also provide a functorial characterisation of “irredundant” networks, this time novel both for the Bayesian and Markov case. Intuitively, irredundant networks provide the correct level of granularity if the focus is on conditional independencies, which is the case when studying moralisation and triangulation.
- Section 4 introduces a suitable notion of morphism for (irredundant) networks, culminating in the definition of categories BN and MN of Bayesian and Markov networks respectively.
- Section 5 characterises moralisation as a functor $\text{BN} \rightarrow \text{MN}$ and triangulation as a functor $\text{MN} \rightarrow \text{BN}$. Importantly, these functors have a high-level description, by precomposition with the functors that define Bayesian and Markov networks. Moreover, as mentioned they are definable inductively on the string diagrammatic syntax representing the graph structure. These clauses, displayed as (6)-(8) below, encapsulate the “essence” of moralisation and triangulation, crystallising it into a syntactic presentation.

Our work is just the beginning of a functorial framework for PGMs: we outline some research directions in Section 6. Proof sketches can be found in the appendices; for complete details, see the full version.

2 Copy-Delete and Hypergraph Categories

The fundamental distinction between Bayesian and Markov networks is that one is a *directed* and the other is an *undirected* model. In this section we recall the categorical structures necessary to account for this difference. First, we focus on the directed case. Following the usual categorical perspective on Bayesian networks [7, 13], we identify in *copy-discard* (CD) categories the requirements needed to interpret the directed acyclic structures of these models. Intuitively, in CD-categories each object has a “copy” and a “delete” map, expressing the ability of nodes of being connected to multiple edges, or to no edge at all. We assume familiarity with *string diagrams* [20, 19], the graphical language of monoidal categories, which we adopt to emphasise the interpretation of morphisms as graphical models.

► **Assumption 1** (Strictness). *Throughout, we always mean monoidal categories and functors to be **strict**: associators, unitors and coherence morphisms for the functors are all identities.*

► **Definition 2.** A **CD-category** is a symmetric monoidal category where each object X is equipped with a commutative comonoid respecting the monoidal structure. We write “copy” (comultiplication) and “delete” (counit) maps in string diagram notation, respectively as $x \leftarrow \text{X}$ and $x \rightarrow \bullet$. We omit the object label when unnecessary. The commutative comonoid equations are then displayed as:

$$\begin{array}{c} \text{---} \leftarrow \text{X} \\ \text{---} \leftarrow \text{X} \end{array} = \begin{array}{c} \text{---} \leftarrow \text{X} \\ \text{---} \leftarrow \text{X} \end{array} \quad \begin{array}{c} \text{---} \leftarrow \text{X} \\ \text{---} \leftarrow \text{X} \end{array} = \begin{array}{c} \text{---} \leftarrow \text{X} \\ \text{---} \leftarrow \text{X} \end{array} \quad \begin{array}{c} \text{---} \leftarrow \text{X} \\ \text{---} \leftarrow \text{X} \end{array} = \text{---} = \begin{array}{c} \text{---} \leftarrow \text{X} \\ \text{---} \leftarrow \text{X} \end{array} \quad (1)$$

Associativity ensures a well-defined “copy” $\leftarrow \text{X}$ with multiple outputs. A **CD-functor** is a symmetric monoidal functor between CD-categories preserving copy and delete maps. CD-categories and CD-functors form a category CDCat .

► **Example 3.** Our chief example of CD-category is FinStoch [9, Example 2.5], the category whose objects are finite sets¹ and whose morphisms $f: X \rightarrow Y$ are maps $Y \times X \rightarrow \mathbb{R}^{\geq 0}$ such that $\sum_{y \in Y} f(y, x) = 1$. We will use a “conditional notation” and write $f(y|x)$ for the image of (y, x) via f . Note that the requirement on f amounts to saying $f(-|x)$ is a probability distribution on Y for each $x \in X$. Another way to view f is as a stochastic matrix (= a matrix where each column sums to 1) with X -labelled columns and Y -labelled rows. Composition is defined via the Chapman–Kolmogorov equation, or equivalently by product of matrices: given $f: X \rightarrow Y$ and $g: Y \rightarrow Z$, $gf(z|x) := \sum_{y \in Y} g(z|y)f(y|x)$. The tensor product is the Kronecker product of matrices; more explicitly, for $f: X \rightarrow Y$ and $h: Z \rightarrow W$, $f \otimes h(y, w|x, z) := f(y|x)h(w|z)$. The structural morphisms yielding the CD structure are defined as follows:

$$\multimap(y, z|x) := \begin{cases} 1 & \text{if } x = y = z \\ 0 & \text{otherwise} \end{cases}, \quad \text{and} \quad \multimap(|x) := 1.$$

This definition motivates the names “copy” and “delete” for the comonoid operations.

► **Example 4 (Free CD-categories).** Recall that a *monoidal signature* is a pair (A, Σ) consisting of a set A of generating objects and a set Σ of generating morphisms typed in A^* (finite lists of A -elements). One may freely construct the CD-category associated with (A, Σ) , denoted $\text{FreeCD}(A, \Sigma)$: its set of objects is A^* and morphisms are obtained by combining Σ -generators with $x \multimap_X^x$ and $x \multimap$, for each $X \in A$, via sequential and parallel composition, and then quotienting by (1) and the laws of symmetric strict monoidal categories. We refer e.g. to [1, 24, 4] for details.

We now turn attention to undirected models. We argue the suitable categorical structure is an extension of CD-categories, variously called *hypergraph categories* [8] or *well-supported compact closed categories* [5].


► **Definition 5.** A **hypergraph category** is a CD-category where furthermore each object X is equipped with a commutative monoid respecting the monoidal structure, and interacting with the comonoid on X via the laws of special Frobenius algebras. We write “compare” (multiplication) and “omni” (unit) maps in string diagram notation, respectively as $\overset{x}{\curvearrowright}_X$ and $\bullet \multimap x$. The commutative monoid equations and special Frobenius equations are displayed as:

$$(2)$$

Associativity ensures a well-defined “compare” $\overset{x}{\curvearrowright}_X$ with multiple inputs. A **hypergraph functor** is a CD-functor preserving the monoid structure. Hypergraph categories and hypergraph functors form a category HypCat .

► **Notation 6.** We sometimes use the “cups” and “caps” notation: $\subset := \bullet \multimap$ and $\supset := \multimap \bullet$.

¹ To ensure strictness of FinStoch , we actually take as objects only finite sets whose elements are finite lists, and define the monoidal product $X \otimes Y$ via list concatenation. For example, if $X = \{[a], [b]\}$ and $Y = \{[a], [c]\}$, then $X \otimes Y = \{[a, a], [a, c], [b, a], [b, c]\}$. This caveat is immaterial for our developments.

The suitability of hypergraph categories to express undirected models may be better appreciated by observing that there is a bijective correspondence between homsets $[X, Y]$, $[I, X \otimes Y]$, and $[X \otimes Y, I]$ for any objects X, Y . These correspondences, obtained by “cups” and “caps”, may be seen graphically as “bending wires”, or “turning inputs into outputs” and vice versa: $x \xrightarrow{f} y$  $\xrightarrow{f} x$. We refer to [8] for a more systematic discussion. Additionally, hypergraph categories allow for the decomposition of “cups” and “caps” into the more elementary “copy”, “delete”, “compare”, and “omni” maps. The expressivity provided by these maps is actually crucial for modelling Bayesian and Markov networks, as all occurring variables are explicitly represented, ensuring that no information is hidden (cf. Figures 1 and 3; see also [12, Section 3]).

► **Example 7.** Our leading example of hypergraph category is $\mathbf{Mat}(\mathbb{R}^{\geq 0})$, the category whose objects are finite sets² and whose morphisms $f: X \rightarrow Y$ are maps $Y \times X \rightarrow \mathbb{R}^{\geq 0}$. Sequential and parallel composition are defined as in $\mathbf{FinStoch}$ (Example 3), and we may use analogous conditional notation and matrix representation for its morphisms. In fact, we may regard $\mathbf{Mat}(\mathbb{R}^{\geq 0})$ -morphisms as (generic) matrices with entries in $\mathbb{R}^{\geq 0}$. $\mathbf{FinStoch}$ is the full subcategory where we restrict to the stochastic matrices. The compare-omni morphisms are defined as follows:

$$\curvearrowright(z | x, y) := \begin{cases} 1 & \text{if } x = y = z \\ 0 & \text{otherwise} \end{cases}, \quad \bullet\text{--}(x |) := 1,$$

Note that $\bullet\text{--}\bullet = \text{id}_I$ is not necessarily satisfied.

► **Example 8.** The category $\mathbf{FinProjStoch}$, as defined in [23, Definition 6.3], is a quotient of $\mathbf{Mat}(\mathbb{R}^{\geq 0})$ given by setting an equivalence relation $f_1 \propto f_2$ whenever there exists $\lambda \in \mathbb{R}^{> 0}$ such that $\lambda f_1(y | x) = f_2(y | x)$ for all $x \in X$ and $y \in Y$. Intuitively, f_1 and f_2 only differ by a global nonzero multiplicative factor. $\mathbf{FinProjStoch}$ inherits the hypergraph category structure of $\mathbf{Mat}(\mathbb{R}^{\geq 0})$ via the functor $\mathbf{Mat}(\mathbb{R}^{\geq 0}) \rightarrow \mathbf{FinProjStoch}$ sending f to its equivalence class $[f]_\propto$. Also, $\mathbf{FinStoch}$ embeds in $\mathbf{FinProjStoch}$ via the analogously defined functor $f \mapsto [f]_\propto$.

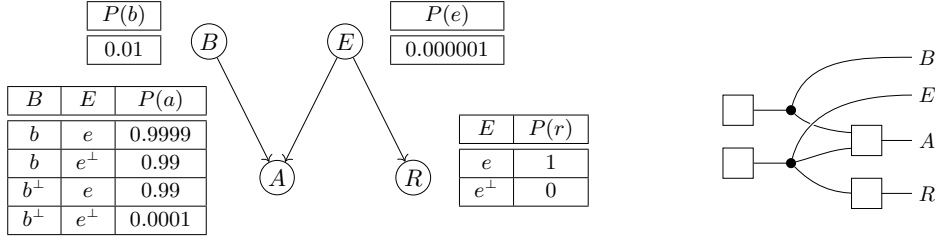
► **Remark 9 (The normalisation cospan).** The introduction of $\mathbf{FinProjStoch}$ is justified by the cospan $\mathbf{Mat}(\mathbb{R}^{\geq 0}) \xrightarrow{q} \mathbf{FinProjStoch} \xleftarrow{i} \mathbf{FinStoch}$, where q is the quotient and i is the embedding described in Example 8. This picture is central to our characterisation of Markov networks, as it describes a normalisation procedure (see Proposition 24).

► **Example 10 (Free hypergraph categories).** Analogously to the case of CD-categories (Example 4), one may freely construct a hypergraph category from a monoidal signature (A, Σ) : the only difference is that the construction of morphisms also involves “structural” generators $\overset{X}{\curvearrowright} \text{--} x$ and $\bullet\text{--} x$ for each $X \in A$, and we additionally quotient by (2).

3 Probabilistic Graphical Models as Functors

In this section we characterise both Bayesian networks and Markov networks via the paradigm of functorial semantics. For Bayesian networks, we build on a previous result [12, Prop. 3.1], whereas the characterisation for Markov networks is novel. These results also help us to characterise an “irredundant” version of networks, which is instrumental for the developments of next sections.

² The same caveat as in the previous footnote applies.



■ **Figure 1** A Bayesian network and the string diagram in **FinStoch** representing it.

Throughout, all graphs are *finite*. Also, in a DAG (directed acyclic graph) $\mathcal{G} = (V_{\mathcal{G}}, E_{\mathcal{G}})$, we denote by $\text{Pa}(v)$ the set of parents of a vertex v , i.e. the set of vertices w such that $w \rightarrow v$.

► **Definition 11.** A **Bayesian network** over \mathcal{G} is given by an assignment $\tau(v)$ of a finite set for any vertex $v \in V_{\mathcal{G}}$ together with a stochastic matrix $\phi_v: \tau(v) \times \prod_{w \in \text{Pa}(v)} \tau(w) \rightarrow [0, 1]$ interpreted as a conditional distribution of v knowing $\text{Pa}(v)$. Any such Bayesian network has an associated distribution given by $P(V_{\mathcal{G}}) = \prod_{v \in V_{\mathcal{G}}} f_v(v \mid \text{Pa}(v))$.

► **Example 12.** The leftmost picture in Figure 1 depicts a Bayesian network, borrowed from [14, Ex. 9.14]. Vertices represent an alarm ($A = \{a, a^\perp\}$), which may be activated by a burglary ($B = \{b, b^\perp\}$) or an earthquake ($E = \{e, e^\perp\}$). The radio ($R = \{r, r^\perp\}$) reliably reports any earthquake. The two elements of each set represent if the event occurred (e.g., a), or not (e.g., a^\perp). Edges indicate causal relationship between events, with probabilities given by stochastic matrices (represented as conditional probability tables).

► **Remark 13 (On ordering).** The stochastic matrix f_v as in Definition 11 can be described by a morphism $\tau(v) \rightarrow \prod_{w \in \text{Pa}(v)} \tau(w)$ in **FinStoch**. However, this assignment is only unique up to permutation. It becomes unique once we choose a specific order for $\prod_{w \in \text{Pa}(v)} \tau(w)$. Thus for simplicity we work with (totally) **ordered graphs**. Recall that a DAG \mathcal{G} is ordered if equipped with a topological ordering, i.e. a total order such that $v \rightarrow w$ implies $v < w$.

The idea behind the functorial perspective is to cleanly separate “syntax” (the graph) and “semantics” (the probability tables) of a Bayesian network. Viewing the graph as syntax is made possible by the free construction of Example 4. Given an ordered DAG $\mathcal{G} = (V_{\mathcal{G}}, E_{\mathcal{G}})$, we define $\text{CDSyn}_{\mathcal{G}}$ as the free CD-category given by the signature $(V_{\mathcal{G}}, \Sigma_{\mathcal{G}})$, where $\Sigma_{\mathcal{G}} := \{\text{Pa}(v) \text{---} \square \text{---} v \mid v \in V_{\mathcal{G}}\}$ and $\text{Pa}(v)$ indicates the parents of v . As stated in the following proposition, we can identify Bayesian networks based on \mathcal{G} with models of \mathcal{G} in **FinStoch**.

► **Proposition 14** ([12, Proposition 3.1]). *Let \mathcal{G} be an ordered DAG. Bayesian networks over \mathcal{G} are in bijective correspondence with CD-functors $\text{CDSyn}_{\mathcal{G}} \rightarrow \text{FinStoch}$.*

A useful observation for our developments, not appearing in [12], is that $\text{CDSyn}_{\mathcal{G}}$ itself may be viewed as a functorial construction. Let **ODAG** be the category whose objects are ordered DAGs, and morphisms are order-preserving graph homomorphisms. Also, recall the category **CDCat** of CD-categories from Definition 2.

► **Theorem 15.** *There is a contravariant functor $\text{CDSyn}: \text{ODAG} \rightarrow \text{CDCat}$ mapping a DAG \mathcal{G} to the CD-category $\text{CDSyn}_{\mathcal{G}}$.*

The contravariance is motivated by considering the preimage of vertices. We provide some intuition on how the functor works via Figure 2 (see Appendix A for further details).

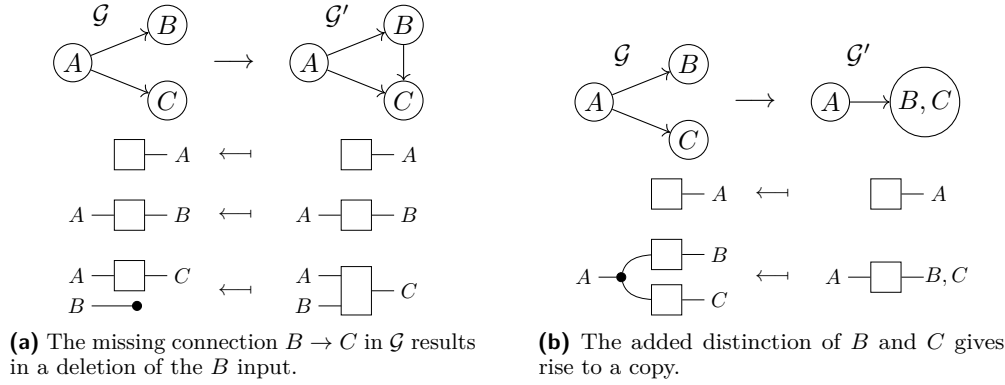
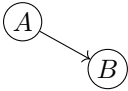


Figure 2 Examples of the contravariant action of the functor of Theorem 15 on morphisms $\mathcal{G} \rightarrow \mathcal{G}'$ (top), resulting in CD-functors (bottom), of which we describe the action on generators of $\text{CDSyn}_{\mathcal{G}}$.

When the probability distribution associated to a Bayesian network does not have full support, i.e. there is some value x such that $P(x) = 0$, the Bayesian network structure has an inherent redundancy. For example, consider the family of Bayesian networks F_p , one for each $p \in [0, 1]$, described by

$A = 0$	$A = 1$	$A = 2$
0.5	0.5	0



A	$P(b)$
0	0.5
1	0.5
2	p

where $A = \{0, 1, 2\}$, $B = \{b, b^\perp\}$. All of these networks are associated to the same probability distribution $P(A, B)$, but they are different according to Definition 11 (and Proposition 14). However, distinguishing them is not always desirable, for instance when studying the conditional independencies of $P(A, B)$. The important property is the existence, rather than the uniqueness of a Bayesian network representing the independencies of $P(A, B)$. For our developments, it is important to account for this different perspective, which we do below.

► **Definition 16** (Irredundant Bayesian Network). *An irredundant Bayesian network over \mathcal{G} is a probability distribution P admitting an assignment $\tau(v)$ of a finite set for any vertex $v \in V_{\mathcal{G}}$ such that $P(V_{\mathcal{G}}) = \prod_{v \in V_{\mathcal{G}}} f_v(v \mid \text{Pa}(v))$ for some stochastic matrices $f_v: \tau(v) \times \prod_{w \in \text{Pa}(v)} \tau(w) \rightarrow [0, 1]$.*

We introduce a characterisation analogous to Proposition 14 for the irredundant case. Note \bullet , the one-vertex graph, is the final object in ODAG. Thus given an ordered DAG \mathcal{G} , there always exists a unique morphism $\mathcal{G} \rightarrow \bullet$, contravariantly yielding a morphism (via Theorem 15) $!_{\mathcal{G}}: \text{CDSyn}_{\bullet} \rightarrow \text{CDSyn}_{\mathcal{G}}$. Another important observation is that, by definition, CDSyn_{\bullet} is generated by the monoidal signature consisting of a single generating object \bullet and a single generating morphisms $\square \dashv$, of type $I \rightarrow \bullet$. Therefore, any CD-functor $\text{CDSyn}_{\bullet} \rightarrow \text{FinStoch}$ is entirely captured by the assignments $\bullet \mapsto X$ and $\square \dashv \mapsto \omega$, where $\omega: I \rightarrow X$ is a probability distribution on X . Paired with Proposition 14, this justifies the following characterisation for “distributions factoring through a Bayesian network”.

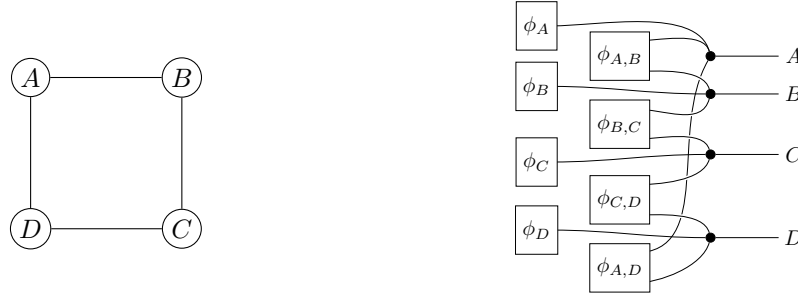
► **Proposition 17.** *Let \mathcal{G} be an ordered DAG. Irredundant Bayesian networks over \mathcal{G} are in bijective correspondence with CD-functors $\omega: \text{CDSyn}_{\bullet} \rightarrow \text{FinStoch}$ factorising as $\text{CDSyn}_{\bullet} \xrightarrow{!_{\mathcal{G}}} \text{CDSyn}_{\mathcal{G}} \xrightarrow{F} \text{FinStoch}$ for some CD-functor F , i.e. such that $\omega = F!_{\mathcal{G}}$.*

► **Example 18.** To better highlight the connection given by Proposition 17, let us consider Example 12. The CD-functor $!_G: \text{CDSyn}_\bullet \rightarrow \text{CDSyn}_G$ is then defined by sending \bullet to $B \otimes E \otimes A \otimes R$ and \square to the string diagram depicted in Figure 1. When composing $!_G$ with the CD-functor $\text{CDSyn}_G \rightarrow \text{FinStoch}$ described by the probability tables in Figure 1, we indeed obtain the probability distribution associated to the Bayesian network.

We now focus on Markov networks, with the goal of achieving characterisations analogous to Propositions 14 and 17. First, we recall how these models are defined in the literature.

► **Definition 19.** Given an undirected graph $\mathcal{H} = (V_{\mathcal{H}}, E_{\mathcal{H}})$, a **clique** C is a subset of $V_{\mathcal{H}}$ such that for each $v, w \in C$, the set $\{v, w\}$ is an edge. The set of cliques is denoted by $\text{Cl}(\mathcal{H})$. A **Markov network** over \mathcal{H} is given by an assignment $\tau(v)$ of a finite set for each vertex v , together with a function $\phi_C: \prod_{v \in C} \tau(v) \rightarrow \mathbb{R}^{\geq 0}$, called factor, for each clique $C \in \text{Cl}(\mathcal{H})$.

Such a Markov network yields a distribution defined by $P(V_{\mathcal{H}}) = \frac{1}{Z} \prod_{C \in \text{Cl}(\mathcal{H})} \phi_C(C)$, where Z is a normalisation coefficient. Note that Z can be zero, thus P is either a probability distribution or identically zero. When P satisfies the latter, we say that the Markov network is *degenerate*. The unnormalised distribution associated to a Markov network can be represented diagrammatically by comparing all occurrences of the outputs of the factors. Before formulating this construction in whole generality, we show it via an example.



■ **Figure 3** An undirected graph and the string diagram representing its unnormalised distribution. Although more graphically complex, the string diagram makes all contributing factors explicit. This approach is also commonly reflected in the theory of PGMs through the use of factor graphs, which are more descriptive.

► **Example 20.** The undirected graph in Figure 3 illustrates the differences between Markov and Bayesian networks. In this scenario, adapted from [15, Ex 3.8], four students – Alice (A), Bob (B), Charles (C), and Debbie (D) – meet in pairs to work on their class homework. A and C do not get along well, nor do B and D , so the only pairs that do *not* meet are these two. During class, the professor misspoke, leading to a potential misconception among the students. Since A and C do not communicate directly to each other, they can only influence each other through B and D . Similarly, B and D only influence each other through A and C . Note the symmetry of these relationships cannot be captured by a Bayesian network, as it inherently requires a specific choice of directionality among student-vertices. An example of a Markov network over this graph is given by the factors below, where vertices are associated with sets $A = \{a, a^\perp\}$, $B = \{b, b^\perp\}$, $C = \{c, c^\perp\}$, $D = \{d, d^\perp\}$.

ϕ_{AB}			ϕ_{BC}			ϕ_{CD}			ϕ_{AD}		
a	b	10	b	c	100	c	d	1	a	d	100
a	b^\perp	1	b	c^\perp	1	c	d^\perp	100	a	d^\perp	1
a^\perp	b	5	b^\perp	c	1	c^\perp	d	100	a^\perp	d	1
a^\perp	b^\perp	30	b^\perp	c^\perp	100	c^\perp	d^\perp	1	a^\perp	d^\perp	100

(3)

In our interpretation, x^\perp indicates that X does not have the misconception. As we do not wish to consider an inherent difference between the various students, all the factors over a single node are omitted (we can simply set them to be constantly one). The chosen numbers highlight some key aspects of the students relationships: C and D are prone to disagree, while all other pairs tend to agree. Additionally, A and B are less likely to have the misconception, and the fact that $\phi_{AB}(a, b^\perp) < \phi_{AB}(a^\perp, b)$ indicates that when they do disagree, it is more plausible that B had the misconception.

To establish an analogue of Proposition 14 for Markov networks, we use the free construction of Example 10. Given any ordered undirected graph $\mathcal{H} = (V_{\mathcal{H}}, E_{\mathcal{H}})$, consider the signature given by $(V_{\mathcal{H}}, \Sigma_{\mathcal{H}})$, with $\Sigma_{\mathcal{H}} := \{\square - C \mid C \in \text{Cl}(\mathcal{H})\}$, and define $\text{HSyn}_{\mathcal{H}} := \text{FreeHyp}(V_{\mathcal{H}}, \Sigma_{\mathcal{H}})$. Note that the order here is necessary to give a well-defined box for each clique, since otherwise we would not know how to order the outputs (cf. Remark 13).

► **Proposition 21.** *Let \mathcal{H} be an ordered undirected graph. Markov networks over \mathcal{H} are in bijective correspondence to hypergraph functors $\text{HSyn}_{\mathcal{H}} \rightarrow \text{Mat}(\mathbb{R}^{\geq 0})$.*

For instance, the Markov network of Example 20 yields $V_{\mathcal{H}} = \{A, B, C, D\}$ and $\Sigma_{\mathcal{H}}$ including all the generators ϕ appearing in the string diagram of Figure 3, and the functor $\text{HSyn}_{\mathcal{H}} \rightarrow \text{Mat}(\mathbb{R}^{\geq 0})$ is defined according to the factors defined in (3).

Just as in the case of Bayesian networks, also the construction of $\text{HSyn}_{\mathcal{H}}$ yields a functor. The categories involved are OUGr , the category of ordered undirected graphs and order-preserving graph homomorphisms, and HypCat , introduced in Definition 5.

► **Theorem 22.** *There is a contravariant functor $\text{HSyn}: \text{OUGr} \rightarrow \text{HypCat}$ mapping a ordered undirected graph \mathcal{H} to the hypergraph category $\text{HSyn}_{\mathcal{H}}$.*

We provide some intuition on how the functor works via Figure 4 (see Appendix B for further details).

The observation about redundancy (Definition 16) is even more relevant for Markov networks. The generality of cliques causes a considerable redundancy of information, which is often “pulled under the rug” in applications, but should be made explicit in a mathematically rigorous treatment.

► **Definition 23** (Irredundant Markov Network). *An irredundant Markov network over an (ordered) undirected graph \mathcal{H} is a probability distribution P admitting an assignment $\tau(v)$ of a finite set for any vertex $v \in V_{\mathcal{H}}$ such that $P(V_{\mathcal{H}}) = \frac{1}{Z} \prod_{C \in \text{Cl}(\mathcal{H})} \phi_C(x_C)$ for some factors ϕ_C , where Z is a normalisation coefficient.*

Note that, because P is a proper distribution, $Z \neq 0$, so any irredundant network is also non-degenerate. As in the case of Bayesian networks, \bullet is the final object in OUGr , yielding a morphism $!_{\mathcal{H}}: \text{HSyn}_{\bullet} \rightarrow \text{HSyn}_{\mathcal{H}}$. However, to take care of the normalisation coefficient, a characterisation analogous to the one of Proposition 17 requires a few extra pieces.

► **Proposition 24.** *Let \mathcal{H} be an ordered undirected graph. Irredundant Markov networks over \mathcal{H} are in bijective correspondence with CD-functors $\omega: \text{CDSyn}_{\bullet} \rightarrow \text{FinStoch}$ for which there exists a commutative diagram as follows for some hypergraph functor $\text{HSyn}_{\mathcal{H}} \rightarrow \text{Mat}(\mathbb{R}^{\geq 0})$:*

$$\begin{array}{ccccc}
 & & \text{HSyn}_{\mathcal{H}} & \longrightarrow & \text{Mat}(\mathbb{R}^{\geq 0}) \\
 \text{HSyn}_{\bullet} & \xrightarrow{!} & & & \searrow q \\
 & \nwarrow \star & \text{CDSyn}_{\bullet} & \xrightarrow{\omega} & \text{FinStoch} \\
 & & & & \nearrow i \\
 & & & & \text{FinProjStoch}
 \end{array} \tag{4}$$

where $\star: \text{CDSyn}_{\bullet} \rightarrow \text{HSyn}_{\bullet}$ sends the unique generator to itself, and q, i are the functors of the normalization cospan, see Remark 9.

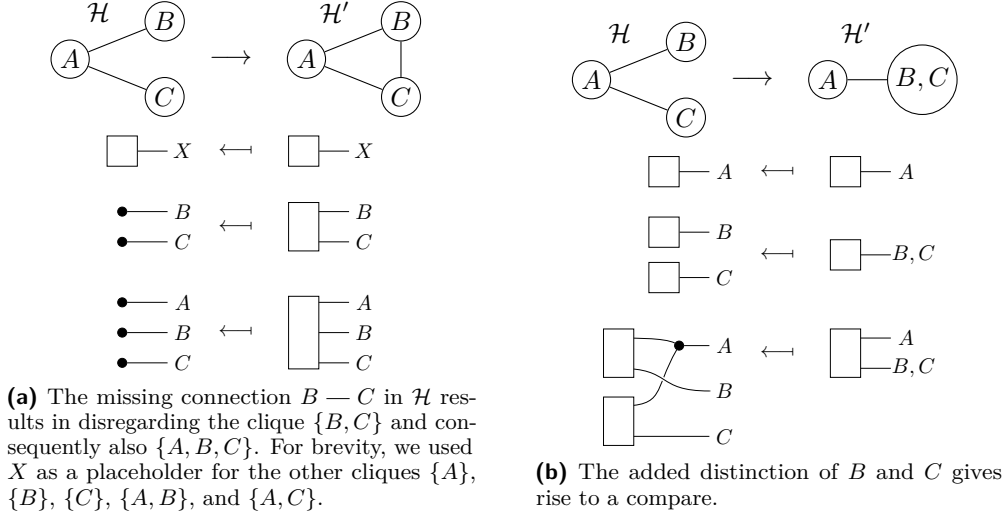


Figure 4 Examples of the contravariant action of the functor of Theorem 22 on morphisms $\mathcal{H} \rightarrow \mathcal{H}'$ (top), resulting in hypergraph functors (bottom), of which we describe the action on generators of $\text{HSyn}_{\mathcal{H}'}$.

The commutativity of (4) states precisely that the probability distribution ω associated with $\omega: \text{CDSyn}_{\bullet} \rightarrow \text{FinStoch}$ and the unnormalised distribution Q associated with $\text{CDSyn}_{\bullet} \rightarrow \text{Mat}(\mathbb{R}^{\geq 0})$ satisfies $Q \propto \omega$, i.e. there exists a normalisation coefficient Z such that $\omega = \frac{1}{Z}Q$.

► **Example 25.** To better understand Proposition 24, let us consider Example 20. The hypergraph functor $!_{\mathcal{H}}: \text{HSyn}_{\bullet} \rightarrow \text{HSyn}_{\mathcal{H}}$ sends \bullet to $A \otimes B \otimes C \otimes D$ and $\square-$ to the string diagram in Figure 3. The unnormalised distribution Q is then described by the composition $\text{HSyn}_{\bullet} \xrightarrow{!} \text{HSyn}_{\mathcal{H}} \rightarrow \text{Mat}(\mathbb{R}^{\geq 0})$, where $\text{HSyn}_{\mathcal{H}} \rightarrow \text{Mat}(\mathbb{R}^{\geq 0})$ is determined by (3).

4 Morphisms Between Networks

In order to define functorial transformations between Bayesian and Markov networks, we need a full definition of the categories involved. The characterisations of Section 3 only provide the objects of these categories. We now identify suitable notions of morphism, culminating in definitions of the categories of Bayesian networks and Markov networks (Definitions 27 and 28). As our ultimate aim is to describe moralisation and triangulation, and these modifications pertain specifically to the study of conditional independencies, the most natural approach is to focus attention to irredundant Bayesian and Markov networks.

Given the functorial perspective on networks, a natural candidate for morphisms are compatible pairs of a “morphism between syntaxes” and a “morphism between semantics”. The former will simply be an order-preserving graph homomorphism. For the latter, recall that both irredundant Bayesian and Markov networks are defined by probability distributions factoring through a certain structure. In FinStoch we may regard two such distributions as maps $\omega: I \rightarrow X$ and $\omega': I \rightarrow Y$, and a morphism between them as a stochastic matrix $f: X \rightarrow Y$ such that $\omega' = f\omega$. Now, in the functorial perspective ω, ω' are identified with CD-functors $\text{CDSyn}_{\bullet} \rightarrow \text{FinStoch}$. We can lift the notion of morphism between distributions from FinStoch to the level of such functors, as follows.

► **Definition 26.** Let $\omega, \omega': \text{CDSyn}_\bullet \rightarrow \text{FinStoch}$ be two CD-functors. A **monoidal transformation** $\eta: \omega \rightarrow \omega'$ is a family of morphisms $\eta_X: \omega(X) \rightarrow \omega'(X)$ for every object $X \in \text{CDSyn}_\bullet$ such that $\eta_{X \otimes Y} = \eta_X \otimes \eta_Y$ and $\eta_I = \text{id}_I$. We say that a monoidal transformation is **distribution-preserving** if $\eta_\bullet \omega(\square-) = \omega'(\square-)$.

The resulting notion is *not* a natural transformation, as one may initially expect: the naturality requirement is too strong, as it only holds when f is a deterministic function (each column has exactly one non-zero value). To justify our notion, observe that distribution-preserving monoidal transformations $\eta: \omega \rightarrow \omega'$ are in bijective correspondence with stochastic matrices f satisfying $\omega'(\square-) = f \omega(\square-)$. Indeed, the objects of CDSyn_\bullet are of the form $\bullet^{\otimes n}$ for some n , and thus $\eta_{\bullet^{\otimes n}} = f^{\otimes n}$ describes such a bijection. We are ready to define the two categories of networks.

► **Definition 27.** The category of (irredundant) Bayesian networks BN is defined as:

- The objects are pairs (ω, \mathcal{G}) , where ω is a CD-functor $\text{CDSyn}_\bullet \rightarrow \text{FinStoch}$ factoring through the CD-functor $!_{\mathcal{G}}: \text{CDSyn}_\bullet \rightarrow \text{CDSyn}_{\mathcal{G}}$ associated to $\mathcal{G} \rightarrow \bullet$.
- A morphism $(\omega, \mathcal{G}) \rightarrow (\omega', \mathcal{G}')$ is a pair (α, η) , where α is a morphism $\alpha: \mathcal{G}' \rightarrow \mathcal{G}$ in ODAG and η is a distribution-preserving monoidal transformation $\omega \rightarrow \omega'$. Composition is component-wise.

► **Definition 28.** The category of (irredundant) Markov networks MN is defined as:

- The objects are pairs (ω, \mathcal{H}) , where ω is a CD-functor $\text{CDSyn}_\bullet \rightarrow \text{FinStoch}$ factoring through $\text{HSyn}_{\mathcal{H}}$ according to (4).
- A morphism $(\omega, \mathcal{H}) \rightarrow (\omega', \mathcal{H}')$ is given by a morphism $\alpha: \mathcal{H}' \rightarrow \mathcal{H}$ in OUGr together with a distribution-preserving monoidal transformation $\eta: \omega \rightarrow \omega'$. Composition is component-wise.

The choice of objects is justified by Propositions 17 and 24. The chosen directionality of morphisms captures the process of *revealing information*. To support this claim, note that for any distribution ω , a morphism $(\omega, \bullet) \rightarrow (\omega, \mathcal{G})$ provides more information about ω in the form of a Bayesian network structure. Further insight is gained by considering extra variables. Explicitly, take (ω, \mathcal{G}) and some additional $v \notin V_{\mathcal{G}}$. Then, any stochastic matrix f with domain a subset of $V_{\mathcal{G}}$ and codomain v can be used to update the distribution: set $\omega' := \omega \cdot f$. The natural graph \mathcal{G}' associated with ω' is obtained from \mathcal{G} by adding the vertex v and edges $w \rightarrow v$ whenever w is an input of f . This defines a revealing variable morphism $(\omega, \mathcal{G}) \rightarrow (\omega', \mathcal{G}')$, given by the surjection $\mathcal{G}' \rightarrow \mathcal{G}$ together with f .

► **Remark 29.** The chosen direction is not suited to *hide* information. For example, take (ω, \mathcal{G}) with \mathcal{G} given by $\textcircled{B} \times \textcircled{A} \times \textcircled{C}$ and imagine we want to delete A . In general, the marginal ω' will display some dependence between B and C , so we need to consider \mathcal{G}' on B and C with an edge in either direction. But then the inclusion $\mathcal{G}' \rightarrow \mathcal{G}$ is not a graph homomorphism, so this hiding process cannot be seen as a morphism $(\omega, \mathcal{G}) \rightarrow (\omega', \mathcal{G}')$. Instead, the surjection $\mathcal{G} \rightarrow \mathcal{G}'$ sending A to either B or C , together with the conditional of A given B and C , gives a morphism $(\omega', \mathcal{G}') \rightarrow (\omega, \mathcal{G})$.

5 Moralisation and Triangulation as Functors

Moralisation and triangulation are well-known transformations that bridge Bayesian and Markov networks. Their importance lies in enabling reasonings and inference methods that are specific to the other type of network. A key example is the junction tree algorithm of Markov networks, which is widely used in machine learning to extract information about marginals and conditionals, see e.g. [15, 2].

The payoff of our perspective is the ability to describe such network transformations by only manipulating the syntactic level, i.e. the categories $\text{CDSyn}_{\mathcal{G}}$ and $\text{HSyn}_{\mathcal{H}}$. Moralisation and triangulation will amount to precomposition with inductively defined functors, see (6), (8) below. We begin by focusing on moralisation, which transforms a Bayesian network \mathcal{G} into a Markov network $\text{Mor}(\mathcal{G})$, with the property that a distribution ω factorising through \mathcal{G} also factorises through $\text{Mor}(\mathcal{G})$ [15].

► **Definition 30.** Let \mathcal{G} be a DAG. Its **moralisation** $\text{Mor}(\mathcal{G})$ is the undirected graph whose vertices are the same as \mathcal{G} and there is an edge between v and w (written $v - w$) whenever in \mathcal{G} there is an edge between them, or they are both parents of the same vertex.

We now discuss how to turn moralisation into a functor $\text{BN} \rightarrow \text{MN}$. Recall that an (irredundant) Bayesian network, that is, an object of BN , is a pair (ω, \mathcal{G}) of a DAG \mathcal{G} and a functor $\omega: \text{CDSyn}_{\bullet} \rightarrow \text{FinStoch}$ factorising as $\text{CDSyn}_{\bullet} \xrightarrow{!_{\mathcal{G}}} \text{CDSyn}_{\mathcal{G}} \xrightarrow{F} \text{FinStoch}$ for some $F: \text{CDSyn}_{\mathcal{G}} \rightarrow \text{FinStoch}$. The corresponding moralisation, an object of MN , is going to be defined as $(\omega, \text{Mor}(\mathcal{G}))$, where ω should factorise through $\text{HSyn}_{\text{Mor}(\mathcal{G})}$ according to (4), instantiated as follows.

$$\begin{array}{ccccc}
 & & \text{HSyn}_{\text{Mor}(\mathcal{G})} & \longrightarrow & \text{Mat}(\mathbb{R}^{\geq 0}) \\
 & \nearrow ! & & & \searrow q \\
 \text{HSyn}_{\bullet} & & & & \text{FinProjStoch} \\
 & \nwarrow \star & \text{CDSyn}_{\bullet} & \xrightarrow{\omega} & \text{FinStoch} \\
 & & & & \nearrow i
 \end{array} \quad (5)$$

The outstanding question is how to correctly define $\text{HSyn}_{\text{Mor}(\mathcal{G})} \rightarrow \text{Mat}(\mathbb{R}^{\geq 0})$ in (5). This goes in three steps. First, we may construct $\text{HSyn}_{\mathcal{G}} := \text{FreeHyp}(V_{\mathcal{G}}, \Sigma_{\mathcal{G}})$, the free *hypergraph* category on \mathcal{G} , which comes with an induced $\star_{\mathcal{G}}: \text{CDSyn}_{\mathcal{G}} \rightarrow \text{HSyn}_{\mathcal{G}}$ given by the identity on the generators $\Sigma_{\mathcal{G}}$. Intuitively, this amounts to taking an “indirect” perspective on the data of \mathcal{G} . Second, observe that $F: \text{CDSyn}_{\mathcal{G}} \rightarrow \text{FinStoch}$ also yields a hypergraph functor $\tilde{F}: \text{HSyn}_{\mathcal{G}} \rightarrow \text{Mat}(\mathbb{R}^{\geq 0})$, defined on generators the same way as F . Third, we define $m: \text{HSyn}_{\text{Mor}(\mathcal{G})} \rightarrow \text{HSyn}_{\mathcal{G}}$ as the hypergraph functor freely obtained by the following mapping on the generators of $\text{HSyn}_{\text{Mor}(\mathcal{G})}$:

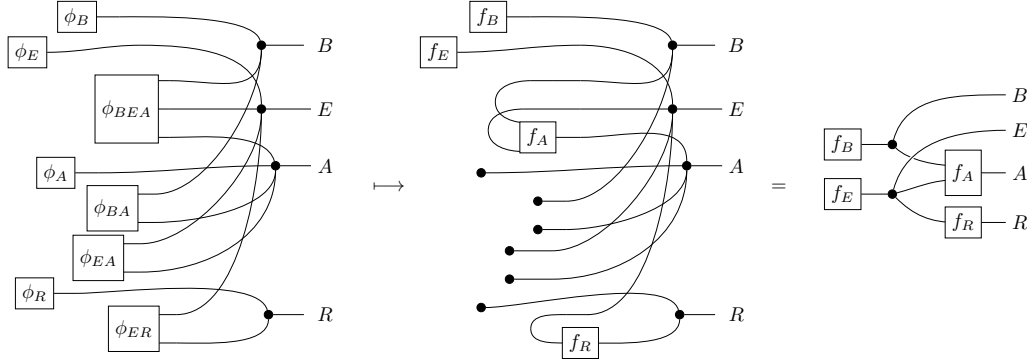
$$\square - C \quad \longmapsto \quad \begin{cases} \begin{array}{c} \text{Pa}(v) \\ \curvearrowright \\ \square - v \\ \bullet - C \end{array} & \text{if } C = \{v\} \cup \text{Pa}(v) \text{ for some } v \in \mathcal{G} \\ \bullet - C & \text{otherwise} \end{cases} \quad (6)$$

This simple description mimics at the level of string diagrammatic syntax the transformation described by Definition 30. Putting these all together, we obtain the desired hypergraph functor $\text{HSyn}_{\text{Mor}(\mathcal{G})} \xrightarrow{m} \text{HSyn}_{\mathcal{G}} \xrightarrow{\tilde{F}} \text{Mat}(\mathbb{R}^{\geq 0})$, thus completing the definition of the Markov network $(\omega, \text{Mor}(\mathcal{G}))$ given by commutativity of (5). More explicitly, this amounts to the following commutative diagram

$$\begin{array}{ccccc}
 \text{HSyn}_{\text{Mor}(\mathcal{G})} & \xrightarrow{m} & \text{HSyn}_{\mathcal{G}} & \xrightarrow{\tilde{F}} & \text{Mat}(\mathbb{R}^{\geq 0}) \\
 \uparrow !_{\text{Mor}(\mathcal{G})} & \nearrow \bar{!}_{\mathcal{G}} & \uparrow \star_{\mathcal{G}} & & \searrow q \\
 \text{HSyn}_{\bullet} & & \text{CDSyn}_{\mathcal{G}} & & \text{FinProjStoch} \\
 \nwarrow \star_{\bullet} & \uparrow !_{\mathcal{G}} & \searrow F & \uparrow & \nearrow i \\
 & \text{CDSyn}_{\bullet} & \xrightarrow{\omega} & \text{FinStoch} &
 \end{array} \quad (7)$$

Observe that the only non-obvious step in this process is the definition of $m: \mathbf{HSyn}_{\text{Mor}(\mathcal{G})} \rightarrow \mathbf{HSyn}_{\mathcal{G}}$. This is the key piece of our functorial view of moralisation, providing two insights: first, moralisation may be decomposed into an inductive definition on the string diagrammatic syntax capturing the graph structures; second, by regarding networks themselves as functors, moralisation may be simply defined by functor precomposition.

► **Example 31.** The moralisation of the DAG in Example 12 is given by adding an additional edge between B and E , since they are both parents of A . $m: \mathbf{HSyn}_{\text{Mor}(\mathcal{G})} \rightarrow \mathbf{HSyn}_{\mathcal{G}}$ maps the string diagram in $\mathbf{HSyn}_{\text{Mor}(\mathcal{G})}$ representing the moralised network, below left, to the string diagram representing the Bayesian network, below right:



where we used ϕ to denote the generators in $\mathbf{HSyn}_{\text{Mor}(\mathcal{G})}$ and f for the generators in $\mathbf{HSyn}_{\mathcal{G}}$. Postcomposing m with $\mathbf{HSyn}_{\mathcal{G}} \rightarrow \mathbf{Mat}(\mathbb{R}^{\geq 0})$ sets ϕ_{BA} , ϕ_{EA} , ϕ_B , ϕ_E and ϕ_{ER} as the stochastic matrices of the Bayesian network, while the other factors are all set to be constantly one.

In full generality, we are able to conclude the following.

► **Theorem 32.** *Moralisation gives rise to a functor $\text{Mor}(-): \mathbf{BN} \rightarrow \mathbf{MN}$ which on objects maps (ω, \mathcal{G}) to $(\omega, \text{Mor}(\mathcal{G}))$.*

Conversely, in the theory of probabilistic graphical models, obtaining a Bayesian network from a Markov network involves a process known as *triangulation*.

► **Definition 33.** *Let \mathcal{H} be an ordered undirected graph. Then its **triangulation** $\text{Tr}(\mathcal{H})$ is the ordered graph where $v \rightarrow w$ whenever $v \leq w$ and there is a path $v - w_1 - \dots - w_n = w$, with $w_i \geq w$ for each $i = 1, \dots, n$.*

In particular, every edge $v - w$ in \mathcal{H} yields a directed edge in $\text{Tr}(\mathcal{H})$. The idea is that $\text{Tr}(\mathcal{H})$ only enforces the conditional independencies that we know to hold thanks to \mathcal{H} .

As in the case of the moralisation, we want to capture triangulation as a functor, but in the opposite direction $\mathbf{MN} \rightarrow \mathbf{BN}$. The construction is similar: the key step is defining the functor $t: \mathbf{HSyn}_{\text{Tr}(\mathcal{H})} \rightarrow \mathbf{HSyn}_{\mathcal{H}}$ which will act by precomposition on the given syntax category $\mathbf{HSyn}_{\mathcal{H}}$ of the Markov network \mathcal{H} . It is defined on the generators of $\mathbf{HSyn}_{\mathcal{H}}$ by the following clause

$$\text{Pa}(v) \text{ --- } \square \text{ --- } v \quad \mapsto \quad \begin{array}{c} Q \text{ --- } \bullet \\ P \text{ --- } \bullet \\ \text{Cmp}(C_v) \text{ --- } v \end{array} \quad (8)$$

where $\text{Cmp}(C_v)$ is a *compare-composition* (see Appendix B), determined by comparing all the outputs of the set of morphisms $C_v := \{\phi: I \rightarrow C \mid v \in C \subseteq \text{Pa}(v) \cup \{v\}\} \subseteq \Sigma_{\mathcal{H}}$, $P := \text{Pa}(v) \cap C_v$ and $Q := \text{Pa}(v) \setminus P$. Intuitively, the right-hand side compares all the factors

involving v and its parents, while Q accounts for the remaining inputs. By showing that $\mathbf{HSyn}_\bullet \rightarrow \mathbf{HSyn}_\mathcal{H}$ factors through $\mathbf{HSyn}_{\mathrm{Tr}(\mathcal{H})}$, it seems we have everything to conclude, since we have the following commutative diagram.

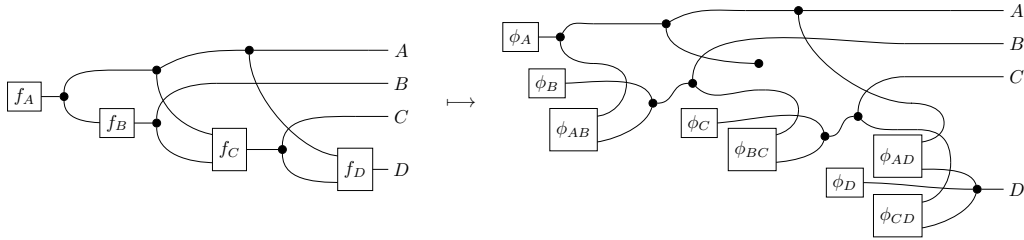
$$\begin{array}{ccccc}
 \mathbf{CDSyn}_\bullet & \xrightarrow{!_{\mathrm{Tr}(\mathcal{H})}} & \mathbf{CDSyn}_{\mathrm{Tr}(\mathcal{H})} & \xrightarrow{*_{\mathrm{Tr}(\mathcal{H})}} & \mathbf{HSyn}_{\mathrm{Tr}(\mathcal{H})} & \xrightarrow{t} & \mathbf{HSyn}_\mathcal{H} \\
 \downarrow \omega & & \swarrow \text{dashed} & & \downarrow & & \swarrow \\
 \mathbf{FinStoch} & \xrightarrow{i} & \mathbf{FinProjStoch} & \xleftarrow{q} & \mathbf{Mat}(\mathbb{R}^{\geq 0}) & &
 \end{array}$$

However, we need to ensure that $\mathbf{HSyn}_{\mathrm{Tr}(\mathcal{H})} \rightarrow \mathbf{Mat}(\mathbb{R}^{\geq 0})$ yields the functor $\mathbf{CDSyn}_{\mathrm{Tr}(\mathcal{H})} \rightarrow \mathbf{FinStoch}$, dashed above, that preserves the associated distribution $\mathbf{CDSyn}_\bullet \rightarrow \mathbf{FinStoch}$. This is true for $\mathrm{Tr}(\mathcal{H})$, due to the peculiar structure of triangulation, which allows to conclude Theorem 34 below. Interestingly, it is not necessarily true for generic DAGs, see Remark 35 below.

► **Theorem 34.** *Triangulation gives rise to a functor $\mathrm{Tr}(-): \mathbf{MN} \rightarrow \mathbf{BN}$ which on objects maps (ω, \mathcal{H}) to $(\omega, \mathrm{Tr}(\mathcal{H}))$.*

► **Remark 35.** Consider the DAG \mathcal{G} given by $\textcircled{A} \rightarrow \textcircled{C} \leftarrow \textcircled{B}$. Any Bayesian network over \mathcal{G} makes A and B independent of each other once we discard C . We claim that this independence is not necessarily true for a hypergraph functor $\mathbf{HSyn}_\mathcal{G} \rightarrow \mathbf{Mat}(\mathbb{R}^{\geq 0})$. To this end, consider $A = B = C = \{0, 1\}$, and let $\phi: I \rightarrow A \otimes B \otimes C$ be the morphism defined by $\phi(a, b, c) = 1$ if exactly two of them are equal and 0 otherwise. We then define $\Phi: \mathbf{HSyn}_\mathcal{G} \rightarrow \mathbf{Mat}(\mathbb{R}^{\geq 0})$ by sending the generators $I \rightarrow A$ and $I \rightarrow B$ to \bullet and $A \otimes B \rightarrow C$ to $\begin{array}{c} A \\ \overline{\phi} \\ B \end{array} \begin{array}{c} \overline{\phi} \\ C \end{array}$. In this way, ϕ corresponds to the distribution $\mathbf{HSyn}_\bullet \xrightarrow{!} \mathbf{HSyn}_\mathcal{G} \xrightarrow{\Phi} \mathbf{Mat}(\mathbb{R}^{\geq 0})$. By direct computations, $\begin{array}{c} \phi \\ \hline \bullet \end{array} \neq \begin{array}{c} \phi \\ \hline \square \end{array}$, so indeed A and B share some dependence even when C is discarded.

► **Example 36.** The triangulation of the undirected graph \mathcal{H} of Example 20 is the DAG obtained by making all edges direct and adding an edge $A \rightarrow C$ (because of the path $A - D - C$). The string diagram representing $\mathrm{Tr}(\mathcal{H})$ in $\mathbf{HSyn}_{\mathrm{Tr}(\mathcal{H})}$ is given below left, with its image under the functor $t: \mathbf{HSyn}_{\mathrm{Tr}(\mathcal{H})} \rightarrow \mathbf{HSyn}_\mathcal{H}$ below right:



The equations in (1) and (2) ensure that the right hand side correspond to the string diagram representing $\mathbf{HSyn}_\mathcal{H}$ (see Figure 3). Now, the factors associated with the original network, defined in (3), yield a hypergraph functor $\Phi: \mathbf{HSyn}_\mathcal{H} \rightarrow \mathbf{Mat}(\mathbb{R}^{\geq 0})$. Postcomposing t with Φ sets the following $\mathbf{Mat}(\mathbb{R}^{\geq 0})$ -semantics for $\mathbf{HSyn}_{\mathrm{Tr}(\mathcal{H})}$ -generators: $f_D(D | AC) := \phi_D(D)\phi_{AD}(AD)\phi_{CD}(CD)$, $f_C(C | AB) := \phi_{BC}(BC)\phi_C(C)$, $f_B(B | A) := \phi_B(B)\phi_{AB}(AB)$ and $f_A(A) := \phi_A(A)$. (For simplicity, here we use the same notation for the generators in the syntax categories and their image in $\mathbf{Mat}(\mathbb{R}^{\geq 0})$). Observe that, by definition, $f_C(C | AB)$ does not really depend on A , but this additional input is forced by the triangulation. We shall see how this plays an important role in our next goal, which is to derive a functor $\mathbf{CDSyn}_{\mathrm{Tr}(\mathcal{H})} \rightarrow \mathbf{FinStoch}$ from $\mathbf{HSyn}_{\mathrm{Tr}(\mathcal{H})} \xrightarrow{t} \mathbf{HSyn}_\mathcal{H} \xrightarrow{\Phi} \mathbf{Mat}(\mathbb{R}^{\geq 0})$ using a normalisation procedure (see the full version for the general statement).

Starting from the vertex D , we define the normalisation coefficient λ_D which depends on A and C , given by $\lambda_D(AC) = f_D(d|AC) + f_D(d^\perp|AC)$. Therefore, $f_D(D|AC) = \lambda_D(AC)g_D(D|AC)$ for some stochastic matrix g_D in FinStoch . The idea is that g_D will be the image of the generator $A \otimes C \rightarrow D$ in $\text{CDSyn}_{\text{Tr}(\mathcal{H})}$.

As our aim is to obtain the same distribution associated to the network, we cannot dismiss the normalisation coefficient $\lambda_D(AC)$, and we therefore consider it together with $f_C(C|AB)$, since f_C is the only morphism where C appears among the remaining $\{f_A, f_B, f_C\}$. We now define a normalisation coefficient $\lambda_C(AB) = \lambda_D(A, c)\phi_{BC}(B, c) + \lambda_D(A, c^\perp)\phi_{BC}(B, c^\perp)$ and obtain $\lambda_D(AC)f_C(C|AB) = \lambda_C(AB)g_C(C|AB)$, where g_C is a stochastic matrix. We note that g_C does depend on A although f_C did not: in particular, the additional input of f_C allows for the construction of g_C without changing the type $A \otimes B \rightarrow C$.

Now, $\lambda_C(AB)$ depends on B , so we consider it with $f_B(B|A)$. As above, we can define the normalisation coefficient $\lambda_B(A)$ and obtain $\lambda_C(AB)f_B(B|A) = \lambda_B(A)g_B(B|A)$ for some stochastic matrix g_B . Similarly for A , we set a normalisation coefficient λ_A , which does not depend on any variable, such that $\lambda_B(A)f_A(A) = \lambda_A g_A$ for some stochastic matrix g_A . By construction, the distribution $\prod_{v \in V_{\text{Tr}(\mathcal{H})}} f_v(v | \text{Pa}(v))$ associated to $\text{HSyn}_{\text{Tr}(\mathcal{H})} \rightarrow \text{Mat}(\mathbb{R}^{\geq 0})$ is equal to $\lambda_A \prod_{v \in V_{\text{Tr}(\mathcal{H})}} g_v(v | \text{Pa}(v)) \propto \prod_{v \in \text{Tr}(\mathcal{H})} g_v(v | \text{Pa}(v))$, so the CD-functor $\text{CDSyn}_{\text{Tr}(\mathcal{H})} \rightarrow \text{FinStoch}$ obtained by the family $\{g_v\}$ does preserve the distribution associated to the network. The values resulting from this procedure are written below.

A	C	$g_D(d AC)$	A	B	$g_C(c AB)$	A	$g_B(b A)$	$g_A(a)$
a	c	0.5	a	b	0.6666	a	0.2307	0.1806
a	c^\perp	0.9999	a	b^\perp	0.0002	a^\perp	0.8475	
a^\perp	c	0.0001	a^\perp	b	0.9998			
a^\perp	c^\perp	0.5	a^\perp	b^\perp	0.3334			

We conclude by making some observation on the interaction of $\text{Tr}(-)$ and $\text{Mor}(-)$.

► **Proposition 37.** *There are two natural transformations: $\text{Tr}(\text{Mor}(-)) \rightarrow \text{id}_{\text{BN}}$ and $\text{Mor}(\text{Tr}(-)) \rightarrow \text{id}_{\text{MN}}$, where id_- indicates the identity functor.*

The actions of these transformations are very simple: in $(\alpha, \eta): \text{Tr}(\text{Mor}(\omega, \mathcal{G})) \rightarrow (\omega, \mathcal{G})$, the contravariant $\alpha: \mathcal{G} \rightarrow \text{Tr}(\text{Mor}(\mathcal{G}))$ is a graph embedding, and $\eta: \omega \rightarrow \omega$ is the identity; the transformation $\text{Mor}(\text{Tr}(-)) \rightarrow \text{id}_{\text{MN}}$ is similarly defined. A more interesting observation is that $\text{Tr}(\text{Mor}(\mathcal{G}))$ may be regarded as a *chordal* representation of \mathcal{G} (see [15, Definitions 2.24 and 2.25]). Finally, the reader may wonder whether an adjunction may be achieved, but this is too much to ask as it would require to have either natural $\text{id}_{\text{BN}} \rightarrow \text{Tr}(\text{Mor}(-))$ or $\text{id}_{\text{MN}} \rightarrow \text{Mor}(\text{Tr}(-))$. However, this is not possible already at the level of graphs, because in general $\text{Tr}(\text{Mor}(\mathcal{G}))$ (resp. $\text{Mor}(\text{Tr}(\mathcal{H}))$) has more edges than \mathcal{G} (resp. \mathcal{H}), and represents less independencies. We leave the details to Appendix C, Proposition 54.

6 Conclusions

We provided a categorical framework for Bayesian and Markov networks, focussing on translations between them in the form of moralisation and triangulation. In the spirit of Lawvere’s categorical algebra, we characterised networks as functors from a “syntax” (the graph) to a “semantics” (its probabilistic interpretation). The translations were formulated abstractly by functor pre-composition, and defined inductively on the diagrammatic syntax.

A first direction for future work is accounting for additional PGMs. Our separation of syntax and semantics offers a clear way forward characterising several existing models, including Gaussian networks (where a discrete probabilistic semantics is replaced with a

Gaussian one), networks based on “partial DAGs” (where the syntax allows both for directed and undirected edges), hidden Markov models (where not all vertices of the graph are “visible” to the string diagram interfaces), and factor graphs (where the syntax describes bipartite graphs). It also abstracts and simplifies reasoning on translations between models, whose specification in the existing literature is often in natural language and prone to ambiguity.

A second direction is to use our abstract account of moralisation and triangulation to study the mathematical structure of the algorithms where these transformations are employed, such as junction tree algorithms, clique tree message passing, variable elimination, and graph-based optimisation. Ultimately, the aim is to offer a compositional perspective on these algorithms, leveraging our syntactic description of moralisation and triangulation.

References

- 1 John C. Baez, Brandon Coya, and Franciscus Rebro. Props in network theory. *Theory Appl. Categ.*, 33:727–783, 2018. URL: www.tac.mta.ca/tac/volumes/33/25/33-25abs.html.
- 2 David Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.
- 3 Gilles Barthe, Joost-Pieter Katoen, and Alexandra Silva, editors. *Foundations of Probabilistic Programming*. Cambridge University Press, 2020.
- 4 Filippo Bonchi, Paweł Sobociński, and Fabio Zanasi. Deconstructing lawvere with distributive laws. *Journal of Logical and Algebraic Methods in Programming*, 95:128–146, 2018. doi:10.1016/j.jlamp.2017.12.002.
- 5 A. Carboni and R.F.C. Walters. Cartesian bicategories I. *Journal of Pure and Applied Algebra*, 49(1):11–32, 1987. doi:10.1016/0022-4049(87)90121-6.
- 6 Andrea Corradini and Fabio Gadducci. An algebraic presentation of term graphs, via gs-monoidal categories. *Appl. Categorical Struct.*, 7(4):299–331, 1999. doi:10.1023/A:1008647417502.
- 7 Brendan Fong. Causal theories: A categorical perspective on bayesian networks, 2013. arXiv:1301.6201.
- 8 Brendan Fong and David I. Spivak. Hypergraph categories. *Journal of Pure and Applied Algebra*, 223(11):4746–4777, 2019. doi:10.1016/j.jpaa.2019.02.014.
- 9 Tobias Fritz. A synthetic approach to Markov kernels, conditional independence and theorems on sufficient statistics. *Adv. Math.*, 370:107239, 2020. arXiv:1908.07021. arXiv:1908.07021.
- 10 Tobias Fritz and Wendong Liang. Free gs-monoidal categories and free markov categories. *Applied Categorical Structures*, 31(2), April 2023. doi:10.1007/s10485-023-09717-0.
- 11 Bart Jacobs. The mathematics of changing one’s mind, via Jeffrey’s or via Pearl’s update rule. *Journal of Artificial Intelligence Research*, 65:783–806, 2019. doi:10.1613/jair.1.11349.
- 12 Bart Jacobs, Aleks Kissinger, and Fabio Zanasi. Causal inference by string diagram surgery. In Mikołaj Bojańczyk and Alex Simpson, editors, *Foundations of Software Science and Computation Structures*, pages 313–329, Cham, 2019. Springer International Publishing. doi:10.1007/978-3-030-17127-8_18.
- 13 Bart Jacobs and Fabio Zanasi. A predicate/state transformer semantics for bayesian learning. In *MFPS*, volume 325 of *Electronic Notes in Theoretical Computer Science*, pages 185–200. Elsevier, 2016. doi:10.1016/J.ENTCS.2016.09.038.
- 14 Bart Jacobs and Fabio Zanasi. The logical essentials of Bayesian reasoning. In *Foundations of probabilistic programming*, pages 295–332. Cambridge: Cambridge University Press, 2021. doi:10.1017/9781108770750.010.
- 15 Daphne Koller and Nir Friedman. *Probabilistic graphical models*. Adapt. Comput. Mach. Learn. Cambridge, MA: MIT Press, 2009.
- 16 F. W. Lawvere. Functorial semantics of algebraic theories. *Proc. Natl. Acad. Sci. USA*, 50:869–872, 1963. doi:10.1073/pnas.50.5.869.
- 17 Robin Lorenz and Sean Tull. Causal models in string diagrams. arXiv:2304.07638. doi:10.48550/arXiv.2304.07638.

- 18 Dan Marsden and Fabrizio Genovese. Custom Hypergraph Categories via Generalized Relations. In Filippo Bonchi and Barbara König, editors, *7th Conference on Algebra and Coalgebra in Computer Science (CALCO 2017)*, volume 72 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 17:1–17:16, Dagstuhl, Germany, 2017. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CALCO.2017.17.
- 19 Robin Piedeleu and Fabio Zanasi. An introduction to string diagrams for computer scientists. *CoRR*, abs/2305.08768, 2023. doi:10.48550/arXiv.2305.08768.
- 20 Peter Selinger. A survey of graphical languages for monoidal categories. In *New structures for physics*, pages 289–355. Berlin: Springer, 2011. arXiv:0908.3347. doi:10.1007/978-3-642-12821-9_4.
- 21 Dan Shiebler, Bruno Gavranovic, and Paul Wilson. Category theory in machine learning, 2021. arXiv:2106.07032.
- 22 Toby St Clere Smithe. Copy-composition for probabilistic graphical models. arXiv:2406.08286.
- 23 Dario Stein and Sam Staton. Probabilistic programming with exact conditions. *J. ACM*, 71(1), 2024. doi:10.1145/3632170.
- 24 Fabio Zanasi. Interacting Hopf algebras: the theory of linear systems. *CoRR*, abs/1805.03032, 2018. arXiv:1805.03032.

A On Bayesian networks

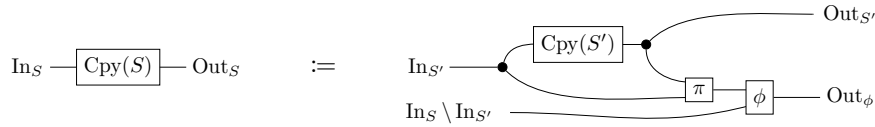
Our main aim in this section is to provide additional background and prove that CDSyn is a functor (Theorem 15).

► **Definition 38.** The category ODAG of ordered DAGs is defined as follows:

- Objects are ordered DAGs, i.e. DAGs equipped with a total order such that $v \rightarrow w$ implies $v < w$;
- Morphisms are order-preserving graph homomorphisms, i.e. $\alpha: \mathcal{G}_1 \rightarrow \mathcal{G}_2$ is a function on the sets of vertices $V_{\mathcal{G}_1} \rightarrow V_{\mathcal{G}_2}$ that preserves the order and the edges.³

► **Notation 39.** For any morphism ϕ in a CD-category, let us denote by In_ϕ and Out_ϕ the sets of inputs and outputs. Similarly, for any set of morphisms S , we consider $\text{Out}_S := \bigcup_{\phi \in S} \text{Out}_\phi$, whereas $\text{In}_S := \bigcup_{\phi \in S} \text{In}_\phi \cap \text{Out}_S^c$.

► **Definition 40.** Let \mathcal{G} be an ordered DAG. Consider a set of generators $S \subseteq \Sigma_{\mathcal{G}}$ and let ϕ be the generator in S whose output is the biggest element of Out_S . By induction, we define $\text{Cpy}(\emptyset) := \text{id}_I$ and



where $S' := S \setminus \{\phi\}$, and π is the marginalization (i.e. it deletes all the occurring vertices that are not inputs of ϕ). In the string diagram we omitted the permutations of the inputs to avoid using additional notation. For a given S , we refer to $\text{Cpy}(S)$ as the **copy-composition** of S .

► **Remark 41.** Although similar, the copy-composition above differs from the one introduced by Smithe in [22]. Indeed, for our purposes, copy-composition must allow inputs and outputs to simply overlap without entirely matching, which extends beyond Smithe's original notion.

³ If two vertices x and y have the same image under α , we assume that the possible edges (x, y) and (y, x) are respected.

On the other hand, this relaxed notion requires additional care as everything depends on a specific order. This is why we have stated it only for sets of generators $S \subseteq \Sigma_{\mathcal{G}}$, making it specific to the syntax categories $\text{CDSyn}_{\mathcal{G}}$ (and $\text{HSyn}_{\mathcal{G}} := \text{FreeHyp}(V_{\mathcal{G}}, \Sigma_{\mathcal{G}})$).

► **Remark 42 (Semantics with the Copy-Composition).** Consider the DAG $\textcircled{A} \rightarrow \textcircled{B}$, and a CD-functor $F: \text{CDSyn}_{\mathcal{G}} \rightarrow \text{FinStoch}$. Set $\omega := F(I \rightarrow A)$ and $f := F(A \rightarrow B)$. Then $F(\text{Cpy}(\Sigma_{\mathcal{G}})) = \omega \rightarrow \textcircled{f}$, which is the probability distribution given by $A \otimes B \ni (a, b) \mapsto f(b|a)\omega(a)$, where the occurrences of a as output of ω and as input of f are identified. In other words, $\omega \rightarrow \textcircled{f} = f(B|A)\omega(A)$.

In general, the interpretation of copy-composition, particularly when given a CD-functor $\text{CDSyn}_{\mathcal{G}} \rightarrow \text{FinStoch}$, is to identify all occurrences of the same variable.

► **Notation 43.** Let $\alpha: \mathcal{G} \rightarrow \mathcal{G}'$ be an order-preserving graph homomorphism, and let $v \in V_{\mathcal{G}'}$. We write S_v^{α} , or S_v when there is no confusion, for the set of generators $\phi \in \Sigma_{\mathcal{G}}$ such that $\text{Out}_{\phi} \subseteq \alpha^{-1}(v)$.

► **Definition 44.** Let $\alpha: \mathcal{G} \rightarrow \mathcal{G}'$ be an order-preserving graph homomorphism. We then define a functor $\text{CDSyn}_{\alpha}: \text{CDSyn}_{\mathcal{G}'} \rightarrow \text{CDSyn}_{\mathcal{G}}$ as follows.

- On objects, v is sent to the tensor product given by $\alpha^{-1}(v)$ (and ordered according to the order on \mathcal{G}).
- On morphisms, CDSyn_{α} is given by

$$\text{Pa}(v) \text{---} \square \text{---} v \quad \mapsto \quad \begin{array}{c} \text{In}_{S_v} \text{---} \boxed{\text{Cpy}(S_v)} \text{---} \alpha^{-1}(v) \\ \text{In}_{S_v}^c \text{---} \bullet \end{array}$$

where $\text{In}_{S_v}^c := \alpha^{-1}(\text{Pa}(v)) \setminus \text{In}_{S_v}$. As in the previous definition, we omit some permutation of objects to avoid additional notation.

Proof sketch of Theorem 15. We want to prove that the association

$$\begin{array}{ccc} \text{CDSyn} : & \text{ODAG} & \rightarrow \\ \mathcal{G}, \quad \alpha: \mathcal{G} \rightarrow \mathcal{G}' & \mapsto & \text{CDSyn}_{\mathcal{G}}, \quad \text{CDSyn}_{\alpha}: \text{CDSyn}_{\mathcal{G}'} \rightarrow \text{CDSyn}_{\mathcal{G}} \end{array}$$

where CDSyn_{α} is defined as in Definition 44, is a contravariant functor. We note that indeed $\text{CDSyn}_{\text{id}} = \text{id}$, so we are left to prove composition. Let $\alpha: \mathcal{G}_1 \rightarrow \mathcal{G}_2$ and $\beta: \mathcal{G}_2 \rightarrow \mathcal{G}_3$. On objects, $\text{CDSyn}_{\alpha}\text{CDSyn}_{\beta}(v) = \text{CDSyn}_{\beta\alpha}(v)$ holds by properties of the preimage and the fact that β and α are order-preserving.

Let us now consider a generator $\text{Pa}(v) \text{---} \square \text{---} v$ in $\text{CDSyn}_{\mathcal{G}_3}$. Via string diagrammatic calculus (see the full version for details), we are reduced to show that $\bigcup_{\phi \in S_v^{\beta}} S_{\text{Out}_{\phi}}^{\alpha} = S_v^{\beta\alpha}$, which holds by direct check. ◀

► **Remark 45 (Factorising Distributions).** An important morphism is the contraction $\mathcal{G} \rightarrow \bullet$, which we often use in the main text. Via the functor CDSyn , this contraction gives the CD-functor $!_{\mathcal{G}}: \text{CDSyn}_{\bullet} \rightarrow \text{CDSyn}_{\mathcal{G}}$, which sends the single generator $I \rightarrow \bullet$ to $\text{Cpy}(\Sigma_{\mathcal{G}})$. This, together with Remark 42, highlights how Proposition 17 is ensured.

B On Markov networks

Here, we provide additional insights into Markov networks. We begin by proving Proposition 21, and then turn to the functoriality of HSyn .

Proof of Proposition 21. Whenever we have a Markov network, we can define the hypergraph functor on generators by setting $F(v) := \tau(v)$ and $F(\square - c) := \phi_C$. Conversely, given a hypergraph functor $F: \mathbf{HSyn}_{\mathcal{H}} \rightarrow \mathbf{Mat}(\mathbb{R}^{\geq 0})$, one simply uses the definitions above in the converse direction: $\tau(v) := F(v)$ and $\phi_C := F(\square - c)$. \blacktriangleleft

► **Definition 46.** The category \mathbf{OUGr} of ordered undirected graphs is defined as follows:

- Objects are ordered undirected graphs;
- Morphisms are order-preserving graph homomorphisms, i.e. $\alpha: \mathcal{H}_1 \rightarrow \mathcal{H}_2$ is a function on the sets of vertices $V_{\mathcal{H}_1} \rightarrow V_{\mathcal{H}_2}$ that preserves the order and the edges.⁴

Similarly to Appendix A, we need to define what it means to consider several morphisms together. Importantly, as here we do not need to take care of directionality, this definition can be extended to a general situation, which will be important in Appendix C.

► **Definition 47.** A morphism $\phi: I \rightarrow C$ in a hypergraph category is a **factor** if it comes equipped with a sequence of objects (X_1, \dots, X_n) such that $X_1 \otimes \dots \otimes X_n = C$.

We write $\text{Out}_{\phi} := \{X_1, \dots, X_n\}$, and, more generally, $\text{Out}_S := \bigcup_{\phi \in S} \text{Out}_{\phi}$ for a finite set of factors S .

A finite set of factors S is assumed to come with an ordering of Out_S to avoid the swapping issue discussed in Remark 13, and this is the case for both $\mathbf{HSyn}_{\mathcal{G}}$ and $\mathbf{HSyn}_{\mathcal{H}}$ by using the order of vertices.

► **Definition 48.** For every finite set of factors S in a hypergraph category, we define by induction $\text{Cmp}(\emptyset) = \text{id}_I$ and

$$\boxed{\text{Cmp}(S)} \text{---} \text{Out}_S \quad := \quad \begin{array}{c} \boxed{\text{Cmp}(S')} \\ \phi \end{array} \begin{array}{l} \text{---} \bullet \text{---} \text{Out}_{S'} \cap \text{Out}_{\phi} \\ \text{---} \text{Out}_{S'} \cap \text{Out}_{\phi}^c \\ \text{---} \text{Out}_{\phi} \cap \text{Out}_{S'}^c \end{array}$$

where ϕ is arbitrarily chosen and $S' := S \setminus \{\phi\}$. Permutations of the outputs on the right hand side are omitted for brevity. For a given S , we refer to $\text{Cmp}(S)$ as the **compare-composition** of S .

► **Remark 49 (Semantics with the Compare-Composition).** Remark 42 can be translated to this setting, meaning that the compare-composition identifies all occurrences of the same variable on the different factors.

► **Definition 50.** Let $\alpha: \mathcal{H} \rightarrow \mathcal{H}'$ be an order-preserving graph homomorphism. The hypergraph functor $\mathbf{HSyn}_{\alpha}: \mathbf{HSyn}_{\mathcal{H}'} \rightarrow \mathbf{HSyn}_{\mathcal{H}}$ is defined as follows:

- It maps an object v to the tensor product given by $\alpha^{-1}(v)$.
- On morphisms,

$$\square \text{---} C \quad \mapsto \quad \begin{array}{c} \boxed{\text{Cmp}(S_C)} \text{---} \text{Out}_{S_C} \\ \bullet \text{---} \alpha^{-1}(C) \setminus \text{Out}_{S_C} \end{array}$$

where $S_C := \{\phi: I \rightarrow D \mid \alpha(D) = C\}$.

⁴ If two vertices x and y have the same image under f , we assume that the possible edge $\{x, y\}$ is respected.

Proof sketch of Theorem 22. We aim to prove that the association

$$\begin{array}{ccc} \text{HSyn} : & \text{OUGr} & \rightarrow \\ \mathcal{H}, & \alpha : \mathcal{H} \rightarrow \mathcal{H}' & \mapsto \text{HSyn}_{\mathcal{H}}, \quad \text{HSyn}_{\alpha} : \text{HSyn}_{\mathcal{H}'} \rightarrow \text{HSyn}_{\mathcal{H}} \end{array}$$

where HSyn_{α} is defined in Definition 50, is a contravariant functor. A direct check shows that $\text{HSyn}_{\text{id}} = \text{id}$, so we focus on composition. Let $\alpha : \mathcal{H}_1 \rightarrow \mathcal{H}_2$ and $\beta : \mathcal{H}_2 \rightarrow \mathcal{H}_3$. As in the proof of Theorem 15, $\text{HSyn}_{\alpha} \text{HSyn}_{\beta}(v) = \text{HSyn}_{\beta\alpha}(v)$ because the considered graph homomorphisms are order-preserving.

Regarding composition, let $\phi : I \rightarrow C$ in $\text{HSyn}_{\mathcal{H}_3}$. By string diagrammatic calculus (see the full version for details), it suffices to show that $\bigcup_{\phi \in S_C^{\beta}} S_{\text{Out}_{\phi}}^{\alpha} = S_C^{\beta\alpha}$, where $S_Y^{\ell} := \{\psi : I \rightarrow X \mid \ell(X) = Y\}$. By unpacking the definition,

$$\bigcup_{\phi \in S_C^{\beta}} S_{\text{Out}_{\phi}}^{\alpha} = \{\psi : I \rightarrow D \mid \exists E \in C\ell(\mathcal{H}_2) \text{ such that } \alpha(D) = E \text{ and } \beta(E) = C\}.$$

Since graph homomorphisms send cliques to cliques, the condition on the right can be shortened to $\beta\alpha(D) = C$, so the wanted equality $\bigcup_{\phi \in S_C^{\beta}} S_{\text{Out}_{\phi}}^{\alpha} = S_C^{\beta\alpha}$ indeed holds. \blacktriangleleft

► **Remark 51 (Factorising Distributions).** Similarly to Remark 45, we consider the contraction map $\mathcal{H} \rightarrow \bullet$, which via HSyn yields the hypergraph functor $!_{\mathcal{H}} : \text{HSyn}_{\bullet} \rightarrow \text{HSyn}_{\mathcal{H}}$ sending the single generator $I \rightarrow \bullet$ to $\text{Cmp}(\Sigma_{\mathcal{H}})$. This highlights how to prove Proposition 24.

C Functoriality of the Moralisation and Triangulation Functors

In the following, we discuss the functoriality of moralisation and triangulation. Further details are available in the full version.

► **Definition 52.** Let \mathcal{C} be a hypergraph category. Given a morphism $f : X \rightarrow Y$, its **graph** is defined as $\boxed{\text{gr}(f)}_Y^X := \overline{\boxed{f}}_Y^X$. For a finite set of morphisms S , we also write $\text{gr}(S)$ for the set of graphs.

► **Lemma 53.** Let \mathcal{G} be an ordered DAG and let $S \subseteq \Sigma_{\mathcal{G}}$ be a set of generators. Then, in $\text{HSyn}_{\mathcal{G}}$, we have $\text{gr}(\text{Cpy}(S)) = \text{Cmp}(\text{gr}(S))$.

The proof is by induction, and it follows from the special Frobenius equations (2). This lemma enables precise discussion without the need to explicitly present string diagrams, and serves as a key ingredient in the proofs of Theorems 32 and 34, which are provided in detail in the full version.

Proof of Proposition 37. The two natural transformations are simply achieved by noting that the identity on an ordered DAG \mathcal{G} and an ordered undirected graph \mathcal{H} are also morphisms $\mathcal{G} \rightarrow \text{Tr}(\text{Mor}(\mathcal{G}))$ and $\mathcal{H} \rightarrow \text{Mor}(\text{Tr}(\mathcal{H}))$. That these correspond to natural transformation follows because the functors $\text{Mor}(-)$ and $\text{Tr}(-)$ send the morphism (α, η) to the same pair interpreted in the new type, as stated at the end of the proofs of Theorems 32 and 34. \blacktriangleleft

► **Proposition 54.** There is no adjunction given by $\text{Mor}(-)$ and $\text{Tr}(-)$.

Proof. By contradiction, let us assume that a natural transformation $\mu_{(\omega, \mathcal{G})} : (\omega, \mathcal{G}) \rightarrow (\omega, \text{Tr}(\text{Mor}(\mathcal{G})))$ exists. For any (ω, \mathcal{G}) and any vertex $v \in \mathcal{G}$, let us consider (ω_v, \bullet) where ω_v is the marginalization of ω at v . The marginalization gives rise to a distribution-

preserving monoidal transformation $\pi: \omega \rightarrow \omega_v$, so in particular we have a morphism $(i, \pi): (\omega, \mathcal{G}) \rightarrow (\omega_v, \bullet)$ where i is the inclusion of graphs $\bullet \rightarrow \mathcal{G}$ sending \bullet to v . By assumption,

$$\begin{array}{ccc} (\omega_v, \bullet) & \xrightarrow{\mu_{(\omega_v, \bullet)}} & (\omega_v, \bullet) \\ \downarrow (i, \pi) & & \downarrow (i, \pi) \\ (\omega, \mathcal{G}) & \xrightarrow{\mu_{(\omega, \mathcal{G})}} & (\omega, \text{Tr}(\text{Mor}(\mathcal{G}))) \end{array}$$

commutes. At the level of graphs, $\mu_{(\omega_v, \bullet)}$ must be given by the identity, as it is the only morphism $\bullet \rightarrow \bullet$. Therefore, whatever $\alpha: \text{Tr}(\text{Mor}(\mathcal{G})) \rightarrow \mathcal{G}$ represents $\mu_{(\omega, \mathcal{G})}$ at the level of graphs, it must respect the commutation of the diagram above, which means that $\alpha(v) = v$. The arbitrariness of v implies that α must be the identity, but the identity is not necessarily a graph homomorphism $\text{Tr}(\text{Mor}(\mathcal{G})) \rightarrow \mathcal{G}$ because $\text{Tr}(\text{Mor}(\mathcal{G}))$ has in general more edges than \mathcal{G} . For the sake of an example, let $\mathcal{G} := \textcircled{A} \rightarrow \textcircled{C} \leftarrow \textcircled{B}$, and note that $\text{Tr}(\text{Mor}(\mathcal{G}))$ is a complete DAG (i.e. we have the additional edge $A \rightarrow B$).

The same idea applies when assuming the existence of a natural transformation $(\omega, \mathcal{H}) \rightarrow (\omega, \text{Mor}(\text{Tr}(\mathcal{H})))$, and therefore $\text{Mor}(-)$ and $\text{Tr}(-)$ do not admit a unit for the possible adjunction in either direction. \blacktriangleleft