Updating Probabilistic Knowledge on Condition/Event Nets using Bayesian Networks

Benjamin Cabrera
University of Duisburg-Essen, Germany
benjamin.cabrera@uni-due.de

Tobias Heindel
University of Hawaii, USA
heindel@hawaii.edu

Reiko Heckel
University of Leicester, UK
rh122@leicester.ac.uk

Barbara König
University of Duisburg-Essen, Germany
barbara_koenig@uni-due.de

Abstract

The paper extends Bayesian networks (BNs) by a mechanism for dynamic changes to the probability distributions represented by BNs. One application scenario is the process of knowledge acquisition of an observer interacting with a system. In particular, the paper considers condition/event nets where the observer's knowledge about the current marking is a probability distribution over markings. The observer can interact with the net to deduce information about the marking by requesting certain transitions to fire and observing their success or failure.

Aiming for an efficient implementation of dynamic changes to probability distributions of BNs, we consider a modular form of networks that form the arrows of a free PROP with a commutative comonoid structure, also known as term graphs. The algebraic structure of such PROPs supplies us with a compositional semantics that functorially maps BNs to their underlying probability distribution and, in particular, it provides a convenient means to describe structural updates of networks.

1 Introduction

Representing uncertain knowledge by probability distributions is the core idea of Bayesian learning. We model the potential of an agent – the *observer* – interacting with a concurrent system with hidden or uncertain state to gain knowledge through "experimenting" with
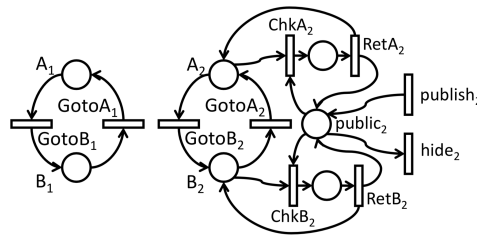
■ **Figure 1** Example: Social network account with location privacy.

the system, focussing on the problem of keeping track of knowledge updates correctly and efficiently. Knowledge about states is represented by a probability distribution. Our system models are condition/even nets where states or possible worlds are markings and transitions describe which updates are allowed.

In order to clarify our intentions we consider an application scenario from social media: preventing inadvertent disclosure, the concern of location privacy [8]. Consider the example of a social network account, modelled as a condition/event net, allowing a user to update and share their location (see Figure 1). We consider two users. User 1 does not allow location updates to be posted to the social network, they are only recorded on their device. In the net this is represented by places A_1 and B_1 modelling the user at corresponding locations, and transitions $GotoA_1$ and $GotoB_1$ for moving between them. We assume that only User 1 can fire or observe these transitions. User 2 has a similar structure for locations and movements, but allows the network to track their location. The user can decide to make their location public or hide it by firing transition $publish_2$ or $hide_2$. Any observer can attempt to fire $ChkA_2$; $RetA_2$ or $ChkB_2$; $RetB_2$ to query the current location of User 2. If $public_2$ is marked, this will allow the observer to infer the correct location. Otherwise the observer is left uncertain as to why the query fails, i.e. due to the wrong location being tested or the lack of permission, unless they test both locations. While our net captures the totality of possible behaviours, we identify different observers, the two users, the social network, and an unrelated observer. For each of these we define which transitions they can access. We then focus on one observer and only allow transitions they are authorised for. In our example, if we want to analyse the unrelated observer, we fix the users' locations and privacy choices before it is the observer's turn to query the system.

The observer may have prior knowledge about the dependencies between the locations of Users 1 and 2, for example due to photos with location information published by User 2, in which both users may be identifiable. The prior knowledge is represented in the initial probability distribution, updated according to the observations.

We also draw inspiration from probabilistic databases [28, 1] where the values of attributes or the presence of records are only known probabilistically. However, an update to the database might make it necessary to revise the probabilities. Think for instance of a database where the gender of a person (male or female) is unknown and we assume with probability 1/2 that they are male. Now a record is entered, stating that the person has married a male. Does it now become more probable that the person is female?

Despite its simplicity, our system model based on condition/event nets allows us to capture databases: the content of a database can be represented as a (hyper-)graph (where each record is a (hyper-)edge). If the nodes of the graph are fixed, updates can be represented by the transitions of a net, where each potential record is represented by a place.

Given a net, the observer does not know the initial marking, but has a prior belief, given by a probability distribution over markings. The observer can try to fire transitions and observe whether the firing is successful or fails. Then the probability distribution is updated accordingly. While the update mechanism is rather straightforward, the problem lies in the huge number of potential states: we have 2^n markings if n is the number of places.

To mitigate this state space explosion, we propose to represent the observer's knowledge using Bayesian networks (BNs) [22, 24], i.e., graphical models that record conditional dependencies of random variables in a compact form. However, we encounter a new problem as updating the observer's knowledge becomes non-trivial. To do this correctly and efficiently, we develop a compositional approach to BNs based on symmetric monoidal theories and PROPs [20]. In particular, we consider modular Bayesian networks as arrows of a freely generated PROP and (sub-)stochastic matrices as another PROP with a functor from the former to the latter. In this way, we make Bayesian networks compositional and we obtain a graphical formalism [27] that we use to modify Bayesian networks: in particular, we can replace entire subgraphs of Bayesian networks by equivalent ones, i.e., graphs that evaluate to the same matrix. The compositional approach allows us to specify complex updates in Bayesian networks by a sequence of simpler updates using a small number of primitives.

We furthermore describe an implementation and report promising runtime results.

The proofs of all results can be found in the full version of this paper [3].

2 Knowledge Update in Condition/Event Nets

We will formalise knowledge updates by means of an extension of Petri nets with probabilistic knowledge on markings. The starting point are condition/event nets [26].

▶ **Definition 1** (Condition/event net). A condition/event net (CN) $N = (S, T, {}^\bullet(), ()^\bullet, m_0)$ is a five-tuple consisting of a finite set of *places* S, a finite set of *transitions* T with *pre-conditions* ${}^\bullet() : T \to \mathcal{P}(S)$, *post-conditions* $()^\bullet : T \to \mathcal{P}(S)$, and $m_0 \subseteq S$ an *initial marking*. A *marking* is any subset of places $m \subseteq S$. We assume that for any $t \in T$, ${}^\bullet t \cap t^\bullet = \emptyset$.

A transition t can *fire* for a marking $m \subseteq S$, denoted $m \Rightarrow^t$, if ${}^\bullet t \subseteq m$ and $t^\bullet \cap m = \emptyset$. Then marking m is transformed into $m' = (m \setminus {}^\bullet t) \cup t^\bullet$, written $m \Rightarrow^t m'$. We write $m \Rightarrow^t$ to indicate that there exists some m' with $m \Rightarrow^t m'$.

We will use two different notations to indicate that a transition cannot fire, the first referring to the fact that the pre-condition is not sufficiently marked, the second stating that there are tokens in the post-condition: $m \not\Rightarrow^t_{pre}$ whenever ${}^\bullet t \not\subseteq m$ and $m \not\Rightarrow^t_{post}$ whenever $t^\bullet \cap m \neq \emptyset$. We denote the *set of all markings* by $\mathcal{M} = \mathcal{P}(S)$.

For simplicity we assume that $S = \{1, \ldots, n\}$ for $n \in \mathbb{N}$. Then, a marking m can be characterized by a boolean vector $m : S \to \{0, 1\}$, i.e., $\mathcal{M} \cong \{0, 1\}^S$. Using the vector notation we write $m(A) = \{1\}$ for $A \subseteq S$ if all places in A are marked in m.

To capture the probabilistic observer we augment CNs by a probability distribution over markings modelling uncertainty about the hidden initial or current marking.

▶ **Definition 2** (Condition/Event net with Uncertainty). A *Condition/Event Net with Uncertainty (CNU)* is a six-tuple $N = (S, T, {}^\bullet(), ()^\bullet, m_0, p)$ where $(S, T, {}^\bullet(), ()^\bullet, m_0)$ is a net as in Definition 1. Additionally, p is a function $p : \mathcal{M} \to [0, 1]$ with $\sum_{m \in \mathcal{M}} p(m) = 1$ that assigns a *probability mass* to each possible marking. This gives rise to a probability space $(\mathcal{M}, \mathcal{P}(\mathcal{M}), \mathbb{P})$ with $\mathbb{P} : \mathcal{P}(\mathcal{M}) \to [0, 1]$ defined by $\mathbb{P}(\{m_1, \ldots, m_k\}) = \sum_{i=1}^{k} p(m_i)$.

We assume that $p(m_0) > 0$, i.e. the initial marking is possible according to p.

We model the knowledge gained by observers when firing transitions and observing their outcomes. Firing $t \in T$ can either result in success (all places of $^{\bullet}t$ are marked and no place in $t^{\bullet}$ is marked) or in failure (at least one place of $^{\bullet}t$ is empty or one place in $t^{\bullet}$ is marked). Thus, there are two kinds of failure, the absence of tokens in the pre-condition or the presence of tokens in the post-condition. If a transition fails for both reasons, the observer will learn only one of them. To model the knowledge gained we define the following operations on distributions.

▶ **Definition 3** (Operations on CNUs)**.** Given a CNU $N = (S, T, {}^{\bullet}(), ()^{\bullet}, m_0, p)$ the following operations update the mass function p and as a result the probability distribution $\mathbb{P}$.

- To assert that certain places $A \subseteq S$ *all contain* a token ($b = 1$) or that none contains a token ($b = 0$) we define the operation *assert*

$$\text{ass}_{A,b}(p)(m) = \frac{p(m)}{\sum_{m' \in \mathcal{M}: m'(A) = \{b\}} p(m')}, \text{ if } m(A) = \{b\} \quad \text{and} \quad 0, \text{ otherwise.}$$

- To state that *at least one* place of a set $A \subseteq S$ *does* (resp. *does not*) *contain* a token we define operation *negative assert*

$$\text{nas}_{A,b}(p)(m) = \frac{p(m)}{\sum_{m' \in \mathcal{M}: m'(A) \neq \{b\}} p(m')}, \text{ if } m(A) \neq \{b\} \quad \text{and} \quad 0, \text{ otherwise.}$$

- *Modifying* a set of places $A \subseteq S$ such that *all places contain a token ($b = 1$) or none contains a token ($b = 0$)* requires the following operation

$$\text{set}_{A,b}(p)(m) = \sum_{m': m'|_{S \setminus A} = m|_{S \setminus A}} p(m'), \text{ if } m(A) = \{b\} \quad \text{and} \quad 0, \text{ otherwise.} \tag{1}$$

- A successful firing of a transition t leads to an assert (ass) and set of the pre-conditions $^{\bullet}t$ and the post-conditions $t^{\bullet}$. A failed firing translates to a negative assert (nas) of the pre- or post-condition and nothing is set. Thus we define for a transition $t \in T$
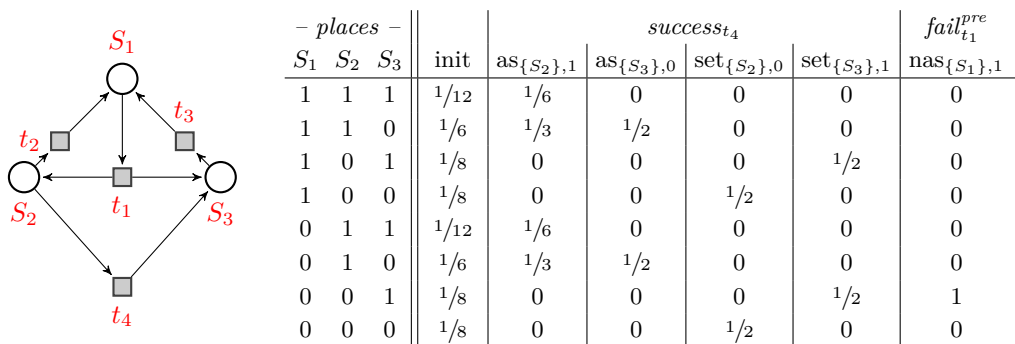
$$\text{success}_t(p) = \text{set}_{t^{\bullet},1}(\text{set}_{{}^{\bullet}t,0}(\text{ass}_{t^{\bullet},0}(\text{ass}_{{}^{\bullet}t,1}(p)))), \qquad \text{fail}_t^{pre}(p) = \text{nas}_{{}^{\bullet}t,1}(p),$$

$$\text{fail}_t^{post}(p) = \text{nas}_{t^{\bullet},0}(p).$$

Operations ass, nas are partial, defined whenever the sum in the denominator of their first clause is greater than 0. That means, the observer only fires transitions whose pre- and postconditions have a probability greater than zero, i.e., where according to their knowledge about the state it is possible that these transitions are enabled. By Definition 1 the initial marking is possible, and this property is maintained as markings and distributions are updated. If this assumption is not satisfied, the operations in Definition 3 are undefined.

The ass and nas operations result from conditioning the input distribution on (not) having tokens at A (compare Proposition 4). Also, set and ass for $A = \{s_1, \ldots, s_k\} \subseteq S$ can be performed iteratively, i.e., $\text{set}_{A,b} = \text{set}_{\{s_k\},b} \circ \cdots \circ \text{set}_{\{s_1\},b}$ and $\text{ass}_{A,b} = \text{ass}_{\{s_k\},b} \circ \cdots \circ \text{ass}_{\{s_1\},b}$. For a single place s we have $\text{ass}_{\{s\},b} = \text{nas}_{\{s\},1-b}$.

Figure 2 gives an example for a Petri net with uncertainty and explains how the observer can update their knowledge by interacting with the net. We can now show that our operations correctly update the probability assumptions according to the observations of the net.

▶ **Proposition 4.** *Let* $N = (S, T, {}^{\bullet}(), ()^{\bullet}, m_0, p)$ *be a CNU where* $\mathbb{P}$ *is the corresponding probability distribution. For given* $t \in T$ *and* $m \in \mathcal{M}$ *let* $\mathcal{M}[\Rightarrow^t] = \{m' \in \mathcal{M} \mid m' \Rightarrow^t\}$, $\mathcal{M}[\Rightarrow^t m] = \{m' \in \mathcal{M} \mid m' \Rightarrow^t m\}$, $\mathcal{M}[\not\Rightarrow_{pre}^t] = \{m' \in \mathcal{M} \mid m' \not\Rightarrow_{pre}^t\}$ *and* $\mathcal{M}[\not\Rightarrow_{post}^t] =$

| – places – | | | init | $success_{t_4}$ | | | | $fail^{pre}_{t_1}$ |
S_1	S_2	S_3		$as_{\{S_2\},1}$	$as_{\{S_3\},0}$	$set_{\{S_2\},0}$	$set_{\{S_3\},1}$	$nas_{\{S_1\},1}$
1	1	1	$1/12$	$1/6$	0	0	0	0
1	1	0	$1/6$	$1/3$	$1/2$	0	0	0
1	0	1	$1/8$	0	0	0	$1/2$	0
1	0	0	$1/8$	0	0	$1/2$	0	0
0	1	1	$1/12$	$1/6$	0	0	0	0
0	1	0	$1/6$	$1/3$	$1/2$	0	0	0
0	0	1	$1/8$	0	0	0	$1/2$	1
0	0	0	$1/8$	0	0	$1/2$	0	0

■ **Figure 2** Example of operations on a net with uncertainty. We set $m_0 = \{S_2\}$ and assume the observer first fires t_4 (and succeeds) and then tries to fire t_1 (and fails). Columns in the table represent updated distributions on the markings after each operation (ordered from left to right). For this example, in the end the observer knows that the final configuration is $\{S_3\}$ with probability 1.

$\{m' \in \mathcal{M} \mid m' \not\Rightarrow^t_{post}\}$. *Then, provided that* $\mathcal{M}[\Rightarrow^t]$, $\mathcal{M}[\not\Rightarrow^t_{pre}]$ *respectively* $\mathcal{M}[\not\Rightarrow^t_{post}]$ *are non-empty, it holds for* $m \in \mathcal{M}$ *that*

$$\text{success}_t(p)(m) = \mathbb{P}(\mathcal{M}[\Rightarrow^t m] \mid \mathcal{M}[\Rightarrow^t]), \qquad \text{fail}^{pre}_t(p)(m) = \mathbb{P}(\{m\} \mid \mathcal{M}[\not\Rightarrow^t_{pre}]),$$

$$\text{fail}^{post}_t(p)(m) = \mathbb{P}(\{m\} \mid \mathcal{M}[\not\Rightarrow^t_{post}]).$$

We shall refer to the the joint distribution (over all places) by $\mathbb{P}$. Note that it is unfeasible to explicitly store it if the number of places is large. To mitigate this problem we use a Bayesian network with a random variable for each place, recording dependencies between the presence of tokens in different places. If such dependencies are local, the BN is often moderate in size and thus provides a compact symbolic representation. However, updating the joint distribution of BNs is non-trivial. To address this problem, we propose a propagation procedure based on a term-based, modular representation of BNs.

3 Modular Bayesian Networks and Sub-Stochastic Matrices

Bayesian networks (BNs) are a graphical formalism to reason about probability distributions. They are visualized as directed, acyclic graphs with nodes random variables and edges dependencies between them. This is sufficient for static BNs whose most common operation is the inference of (marginalized or conditional) distributions of the underlying joint distribution.

For a rewriting calculus on dynamic BNs, we consider a modular representation of networks that do not only encode a single probability vector, but a matrix, with several input and output ports. The first aim is compositionality: larger nets can be composed from smaller ones via sequential and parallel composition, which correspond to matrix multiplication and Kronecker product of the encoded matrices. This means, we can implement the operations of Section 2 in a modular way.

PROPs with Commutative Comonoid Structure

We now describe the underlying compositional structure of (modular) BNs and (sub-) stochastic matrices, which facilitates a compositional computation of the underlying probability distribution of (modular) BNs. The mathematical structure are PROPs [20] (see also [13, Chapter 5.2]), i.e., strict symmetric monoidal categories $(C, \otimes, 0, \sigma)$ whose objects are (in

Figure 3 String diagrammatic composition (resp. tensor) of two arrows $f\colon m \to n$, $f'\colon n \to k$ (resp. $f_1\colon m_1 \to n_1$, $f_2\colon m_2 \to n_2$) of a PROP $(C, \otimes, 0, \sigma)$.

Table 1 Axioms for CC-structured PROPs and definition of operators of higher arity.

$$(t_1; t_3) \otimes (t_2; t_4) = (t_1 \otimes t_2); (t_3 \otimes t_4) \qquad (t_1; t_2); t_3 = t_1; (t_2; t_3)$$

$$\mathrm{id}_n; t = t = t; \mathrm{id}_m \qquad (t_1 \otimes t_2) \otimes t_3 = t_1 \otimes (t_2 \otimes t_3) \qquad \mathrm{id}_0 \otimes t = t = t \otimes \mathrm{id}_0$$

$$\sigma; \sigma = \mathrm{id}_2 \qquad (t \otimes \mathrm{id}_m); \sigma_{n,m} = \sigma_{m,n}; (\mathrm{id}_n \otimes t) \qquad \nabla; (\nabla \otimes \mathrm{id}_1) = \nabla; (\mathrm{id}_1 \otimes \nabla)$$

$$\nabla = \nabla; \sigma \qquad \nabla; (\mathrm{id}_1 \otimes \top) = \mathrm{id}_1$$

$$\mathrm{id}_1 = \mathrm{id} \qquad \mathrm{id}_{n+1} = \mathrm{id}_n \otimes \mathrm{id}_1$$

$$\sigma_{n,0} = \sigma_{0,n} = \mathrm{id}_n \qquad \sigma_{n+1,1} = (\mathrm{id} \otimes \sigma_{n,1}); (\sigma \otimes \mathrm{id}_n)$$

$$\sigma_{n,m+1} = (\sigma_{n,m} \otimes \mathrm{id}_1); (\mathrm{id}_m \otimes \sigma_{n,1})$$

$$\nabla_1 = \nabla \qquad \nabla_{n+1} = (\nabla_n \otimes \nabla); (\mathrm{id}_n \otimes \sigma_{n,1} \otimes \mathrm{id})$$

$$\top_1 = \top \qquad \top_{n+1} = \top_n \otimes \top$$

bijection with) the natural numbers, with monoidal product $\otimes$ as (essentially) addition, with unit 0. The compositional structure of PROPs can be intuitively represented using string diagrams with wires and boxes (see Figure 3). Symmetries σ serve for the reordering of wires.

A paradigmatic example is the PROP of 2^n-dimensional Euclidean spaces and linear maps, equipped with the tensor product: the tensor product of 2^n- and 2^m-dimensional spaces is 2^{n+m}-dimensional, composition of linear maps amounts to matrix multiplication, and the tensor product is also known as Kronecker product (as detailed below). We refer to the natural numbers of the domain and codomain of arrows in a PROP as their *type*; thus, a linear map from 2^n- to 2^m-dimensional Euclidean space has type $n \to m$.

We shall have the additional structure on symmetric monoidal categories that was dubbed *graph substitution* in work on term graphs [7], which amounts to a commutative comonoid structure on PROPs.

▶ **Definition 5** (PROPs with commutative comonoid structure). A *CC-structured PROP* is a tuple $(C, \otimes, 0, \sigma, \nabla, \top)$ where $(C, \otimes, 0, \sigma)$ is a PROP and the last two components are arrows $\nabla\colon 1 \to 2$ and $\top\colon 1 \to 0$, which are subject to equations (2) (cf. Figure 4).

$$\nabla; (\nabla \otimes \mathrm{id}_1) = \nabla; (\mathrm{id}_1 \otimes \nabla), \qquad \nabla = \nabla; \sigma, \qquad \nabla; (\mathrm{id}_1 \otimes \top) = \mathrm{id}_1. \qquad (2)$$

To give another, more direct definition, the arrows of a freely generated CC-structured PROP can be represented as terms over some set of generators $g \in G$ and constants $\mathrm{id}\colon 1 \to 1$, $\sigma\colon 2 \to 2$, $\nabla\colon 1 \to 2$, $\top\colon 1 \to 0$, combined with the operators sequential composition (;) and tensor ($\otimes$) and quotiented by the axioms in Table 1 (see [30]). This table also lists
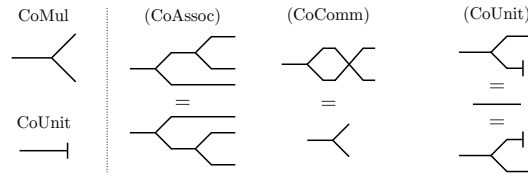
Figure 4 Comultiplication and counit arrows and the equations of commutative comonoids.

the definition of operators of higher arity. We often refer to the comultiplication Δ and its counit $\top$ as *duplicator* and *terminator*, resp. (cf. Figure 4). Intuitively, adding the commutative comonoid structure amounts to the possibility to have several or no connections to each one of the output port of gates and input ports. In other words, outputs can be shared.

(Sub-)Stochastic Matrices

We now consider (sub-)stochastic matrices as an instance of a CC-structured PROP. A *matrix* of type $n \to m$ is a matrix P of dimension $2^m \times 2^n$ with entries taken from the closed interval $[0, 1] \subseteq \mathbb{R}$. We restrict attention to sub-stochastic matrices, i.e., column sums will be at most 1; if we require equality, we obtain stochastic matrices.

We index matrices over $\{0, 1\}^m \times \{0, 1\}^n$, i.e., for $\mathbf{x} \in \{0, 1\}^m$, $\mathbf{y} \in \{0, 1\}^n$ the corresponding entry is denoted by $P(\mathbf{x} \mid \mathbf{y})$. We use this notation to evoke the idea of conditional probability (the probability that the first index is equal to $\mathbf{x}$, whenever the second index is equal to $\mathbf{y}$.) When we write P as a matrix, the rows/columns are ordered according to a descending sequence of binary numbers ($1 \ldots 1$ first, $0 \ldots 0$ last).

$$
\begin{array}{c}
\begin{array}{cccc} 11 & 10 & 01 & 00 \end{array} \\
\begin{array}{c} 11 \\ 10 \\ 01 \\ 00 \end{array}
\left(\begin{array}{cccc}
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot
\end{array}\right)
\end{array}
$$

Sequential composition is *matrix multiplication*, i.e., given $P \colon n \to m$, $Q \colon m \to \ell$ we define $P; Q = Q \cdot P \colon n \to \ell$, which is a $2^\ell \times 2^n$-matrix. The tensor is given by the *Kronecker product*, i.e., given $P \colon n_1 \to m_1$, $Q \colon n_2 \to m_2$ we define $P \otimes Q \colon n_1 + n_2 \to m_1 + m_2$ as $(P \otimes Q)(\mathbf{x}_1\mathbf{x}_2 \mid \mathbf{y}_1\mathbf{y}_2) = P(\mathbf{x}_1 \mid \mathbf{y}_1) \cdot Q(\mathbf{x}_2 \mid \mathbf{y}_2)$ where $\mathbf{x}_i \in \{0, 1\}^{n_i}$, $\mathbf{y}_i \in \{0, 1\}^{m_i}$.

The constants are defined as follows:

$$
\mathrm{id}_0 = (1) \quad \mathrm{id} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \nabla = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \quad \sigma = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \top = \begin{pmatrix} 1 & 1 \end{pmatrix}
$$

▶ **Proposition 6** ([12]). *(Sub-)stochastic matrices form a CC-structured PROP.*

Causality Graphs

We next introduce causality graphs, a variant of term graphs [7], to provide a modular representation of Bayesian networks. Nodes play the role of gates of string diagrams; the main difference to port graphs [13, Chapter 5] is the branching structure at output ports, which corresponds to (freely) added commutative comonoid structure. We fix a set of generators G (a.k.a. signature), elements of which can be thought of as blueprints of gates of a certain type; all generators $g \in G$ will be of type $n \to 1$, which means that each node can be identified with its single output port while it has a certain number of input ports.

▶ **Definition 7** (Causality Graph (CG)). A *causality graph (CG)* of type $n \to m$ is a tuple $B = (V, \ell, s, \text{out})$ where

- V is a set of *nodes*,
- $\ell \colon V \to G$ is a *labelling function* that assigns a generator $\ell(v) \in G$ to each node $v \in V$,
- $s \colon V \to W_B^*$ where $W_B = V \cup \{i_1, \ldots, i_n\}$ is the *source function* that assigns a sequence of *wires* $s(v)$ to each node $v \in V$ such that $|s(v)| = n$ if $\ell(v) \colon n \to 1$,
- $\text{out} \colon \{o_1, \ldots, o_m\} \to W_B$ is the *output function* that assigns each output port to a wire.

Moreover, the corresponding directed graph (defined by s) has to be acyclic.

By $\{i_1, \ldots, i_n\}$ we denote the set of *input ports* and by $\{o_1, \ldots, o_m\}$ the set of *output ports*. By pred and succ we denote the direct predecessors and successors of a node, i.e. $\text{pred}(v_0) = \{v \in V \mid v \in s(v_0)\}$ and $\text{succ}(v_0) = \{v \in V \mid v_0 \in s(v)\}$, respectively. By $\text{pred}^*(v_0)$ we denote the set of indirect predecessors, using transitive closure. Furthermore $\text{path}(v, w)$ denotes the set of all nodes which lie on paths from v to w.

A wire originates from a single input port or node and each node can feed into several successor nodes and/or output ports. Note that input and output are not symmetric in the context of causality graphs. This is a consequence of the absence of a monoid structure.

We equip CGs with operations of composition and tensor product, identities, and a commutative comonoid structure. We require that the node sets of Bayesian nets B_1, B_2 are disjoint.[1]

Composition. Whenever $m_1 = n_2$, we define $B_1 ; B_2 := B = (V, \ell, s, \text{out}) \colon n_1 \to m_2$ with $V = V_1 \uplus V_2$, $\ell = \ell_1 \uplus \ell_2$, $s = s_1 \uplus c \circ s_2$, $\text{out} = c \circ \text{out}_2$ where $c \colon W_{B_2} \to W_B$ is defined as follows and extended to sequences: $c(w) = w$ if $w \in V_2$ and $c(w) = \text{out}_1(o_j)$ if $w = i_j$.

Tensor. Disjoint union is parallel composition, i.e., $B_1 \otimes B_2 := B = (V, \ell, s, \text{out}) \colon n_1 + n_2 \to m_1 + m_2$ with $V = V_1 \uplus V_2$, $\ell = \ell_1 \uplus \ell_2$, $s = s_1 \uplus d \circ s_2$, where $d \colon W_{B_2} \to W_B$ and $\text{out} \colon \{o_1, \ldots, o_{m_1+m_2}\} \to W_B$ are defined as follows: $d(w) = w$ if $w \in V_2$ and $d(w) = i_{n_1+j}$ if $w = i_j$. Furthermore $\text{out}(o_j) = \text{out}_1(o_j)$ if $1 \le j \le m_1$ and $\text{out}(o_j) = \text{out}_2(o_{j-m_1})$ if $m_1 < j \le m_1 + m_2$.

Operators. Finally the constants and generators are as follows:[2]

$$\text{id}_0 = (\emptyset, [], [], []) \colon 0 \to 0 \quad \text{id} = (\emptyset, [], [], [o_1 \mapsto i_1]) \colon 1 \to 1 \quad \top = (\emptyset, [], [], []) \colon 1 \to 0$$
$$\sigma = (\emptyset, [], [], [o_1 \mapsto i_2, o_2 \mapsto i_1]) \colon 2 \to 2 \quad \nabla = (\emptyset, [], [], [o_1 \mapsto i_1, o_2 \mapsto i_1]) \colon 1 \to 2$$
$$B_g = (\{v\}, [v \mapsto g], [v \mapsto i_1 \ldots i_n], [o_1 \mapsto v]) \colon n \to 1, \text{ whenever } g \in G \text{ with type } g \colon n \to 1$$

Finally, all these operations lift to isomorphism classes of CGs.

▶ **Proposition 8** ([7]). *CGs quotiented by isomorphism form the freely generated CC-structured PROP over the set of generators G, where two causality graphs $B_i = (V_i, \ell_i, s_i, \text{out}_i) \colon n \to m$, $i \in \{1, 2\}$, are isomorphic if there is a bijective mapping $\varphi \colon V_1 \to V_2$ such that $\ell_1(v) = \ell_2(\varphi(v))$ and $\varphi(s_1(v)) = s_2(v)$ hold for all $v \in V_1$ and $\varphi(\text{out}_1(o_i)) = \text{out}_2(o_i)$ holds for all $i \in \{1, \ldots, m\}$.*[3]

In the following, we often decompose a CG into a subgraph and its "context".

▶ **Lemma 9** (Decompositionality of CGs). *Let $B = (V, \ell, s, \text{out}) \colon n \to m$ be a causality graph. Let $V' \subseteq V$ be a subset of nodes closed with respect to paths, i.e. for all $v, w \in V' \colon \text{path}(v, w) \subseteq V'$. Then there exist $k \in \mathbb{N}$ and (B_i, e_i) with $B_i = (V_i, l_i, s_i, \text{out}_i)$ for $i = 1, \ldots, 3$ such that $V_2 = V'$, $B = B_1 ; (\text{id}_k \otimes B_2) ; B_3$ and $\text{out}_2(o_i) \in V'$ for all i.*

---

[1] The case of non-disjoint sets can be handled by a suitable choice of coproducts.
[2] A function $f \colon A \to B$, where $A = \{a_1, \ldots, a_k\}$ is finite, is denoted by $f = [a_1 \mapsto f(a_1), \ldots, a_k \mapsto f(a_k)]$. We denote a function with empty domain by $[]$.
[3] We apply φ to a sequence of wires, by applying φ pointwise and assuming that $\varphi(i_j) = i_j$ for $1 \le j \le n$.
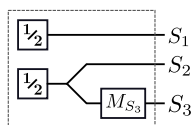
Figure 5 The initial distribution of the CNU from Figure 2 as an MBN.

Thus, given a set of nodes in a BN that contains all nodes on paths between them, we have the induced subnet of the node set and a suitable "context" such that the whole net can be seen as the result of substition of the subnet into the "context".

Modular Bayesian Networks

We will now equip the nodes of causality graphs with matrices, assigning an interpretation to each generator. This fully determines the corresponding matrix of the BN. Note that Bayesian networks as PROPs have earlier been studied in [12, 16, 17].

▶ **Definition 10** (Modular Bayesian network (MBN))**.** A *modular Bayesian network (MBN)* is a tuple (B, e) where $B = (V, \ell, s, \mathrm{out})$ is a causality graph and e an *evaluation function* that assigns to every generator $g \in G$ with $g \colon n \to 1$ a $2^n \times 2$-matrix $e(g)$. An MBN (B, e) is called an *ordinary Bayesian network (OBN)* whenever B has no inputs (i.e. $B : 0 \to m$), out is a bijection, and every node is associated with a stochastic matrix.

In an OBN every node V corresponds to a random variable and it represents a probability distribution on $\{0, 1\}^m$. OBNs are exactly the Bayesian networks considered in [14].

▶ **Example 11.** *Figure 5 gives an example of a BN where* $\mathbf{1}/\mathbf{2} = \begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix}$ *and* $M_{S_3} = \begin{pmatrix} 1/3 & 1/2 \\ 2/3 & 1/2 \end{pmatrix}$. *It encodes exactly the probability distribution from Figure 2. Its term representation is* $(g_1 \otimes (g_2; \nabla)); (\mathrm{id}_2 \otimes g_3)$ *where* $e(g_1) = e(g_2) = \mathbf{1}/\mathbf{2}$ *and* $e(g_3) = M_{S_3}$.

▶ **Definition 12** (MBN semantics)**.** Let (B, e) be an MBN where the network $B = (V, \ell, s, \mathit{out})$ is of type $n \to m$. The *MBN semantics* is the matrix $M_e(B)$ with

$$\Big(M_e(B) \Big)(x_1 \ldots x_m \mid y_1 \ldots y_n) = \sum_{\substack{b \colon W_B \to \{0,1\} \\ b(i_j) = y_j, b(out(o_i)) = x_i}} \prod_{v \in V} e\big(\ell(v)\big)\Big(b(v) \; \Big| \; b(s(v))\Big)$$

with $x_1, \ldots, x_m, y_1, \ldots, y_m \in \{0, 1\}$ where b is applied pointwise to sequences.

Intuitively the function b assigns boolean values to wires, in a way that is consistent with the input/output values $(x_1 \ldots x_m, y_1 \ldots y_n)$. For each such assignment, the corresponding entries in the matrices $\ell(v)$ are multiplied. Finally, we sum over all possible assignments.

▶ Remark. The semantics $M_e(B)$ is compositional. It is the canonical (i.e., free) extension of the evaluation map from single nodes to the causality graph of an MBN (B, e). Here, we rely on two different findings from the literature, namely, the CC-PROP structure of (sub-)stochastic matrices [12] and the characterization of term graphs as the free symmetric monoidal category with graph substitution [7]. The formal details can be found in [3].

4 Updating Bayesian Networks

We have introduced MBNs as a compact and compositional representation of distributions on markings of a CNU. Coming back to the scenario of knowledge update, we now describe how success and failure of operations requested by the observer affect the MBN. We will first
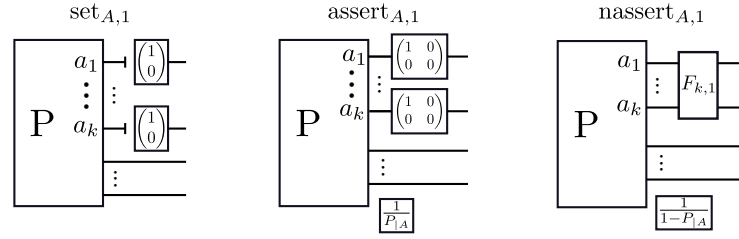
Figure 6 String diagrams of the updated distributions after $\mathrm{set}_{A,1}$, $\mathrm{ass}_{A,1}$, $\mathrm{nas}_{A,1}$ operations were applied to an initial distribution P.

describe how the operations can be formulated as matrix operations that tell us which nodes have to be added to the MBN. We shall see that updated MBNs are in general not OBNs, which makes it harder to interpret and retrieve the encoded distribution. However, we shall show that MBNs can efficiently be reduced to OBNs.

Notation. In this section we will use the following notation: first, we will use variants $\mathrm{id}_n, \nabla_n, \sigma_{n,m}, \top_n$ of the operators/matrices $\mathrm{id}, \nabla, \sigma, \top$, which have a higher arity (see the definitions in Table 1). Furthermore, we will write $\prod_{i=1}^k P_i$ for $P_1 \cdot \ldots \cdot P_k$ and $\bigotimes_{i=1}^k P_i$ for $P_1 \otimes \cdots \otimes P_k$. By $0 : 1 \to 1$ we denote the 2×2 zero matrix and set $0_k = \bigotimes_{i=1}^k 0$. We also introduce $\mathbf{1}_b$ as a notation for the matrix $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ if $b = 1$ (respectively $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ if $b = 0$).

With $\mathrm{diag}(a_1, \ldots, a_n)$ we denote a square matrix with entries $a_1, \ldots, a_n \in [0,1]$ on the diagonal and zero elsewhere. In particular, we will need the sub-stochastic matrices $F_{k,b} : k \to k$ where $F_{k,0} = \mathrm{diag}(\underbrace{1, \ldots, 1}_{2^k - 1 \text{ times}}, 0)$ and $F_{k,1} = \mathrm{diag}(0, \underbrace{1, \ldots, 1}_{2^k - 1 \text{ times}})$.

Given a bit-vector $\mathbf{x} \in \{0,1\}^n$, we will write $\mathbf{x}_{[i]}$ respectively $\mathbf{x}_{[i \ldots j]}$ to denote the i-th entry respectively the sub-sequence from position i to position j. If $A \subseteq \{1, \ldots, n\}$ we define $\mathbf{x}_{[A]} = \{\mathbf{x}_{[i]} \mid i \in A\}$.

CNU Operations on MBNs

In this section we characterize the operations of Definition 3 as stochastic matrices that can be multiplied with the distribution to perform the update. We describe them as compositions of smaller matrices that can easily be interpreted as changes to an MBN. In the following lemmas, $P : 0 \to m$ is always a stochastic matrix representing the distribution of markings of a CNU. Furthermore, $A \subseteq S$ is a set of places and w.l.o.g. we assume that $A = \{1, \ldots, k\}$ for some $k \leq m$ (as otherwise we can use permutations that preceed and follow the operations and switch wires as needed).

Starting with the $\mathrm{set}_{A,b}$ operation (1) recall that it is defined in a way so that the marginal distributions of non-affected places $S \backslash A$ stay the same while the marginals of every single place in A report $b \in \{0,1\}$ with probability one. The following lemma shows how the matrix for a set operation can be constructed (see Figure 6).

▶ **Lemma 13.** *It holds that* $\mathrm{set}_{A,b}(P) = \left(\bigotimes_{i=1}^m T_{A,b}^{\mathrm{set}}(i) \right) \cdot P$ *where* $T_{A,b}^{\mathrm{set}}(i)$ *is* $\mathbf{1}_b \cdot \top$ *if* $i \in A$, *and* id *otherwise. Moreover,* $\bigotimes_{i=1}^m T_{A,b}^{\mathrm{set}}(i)$ *is stochastic.*

Next, we deal with the ass operation. Applying it to a distribution P is simply a conditioning of P on non-emptiness of all places A. Intuitively, this means that we keep only entries of P for which the condition is satisfied and set all other entries to zero. However, in

order to keep the updated P a probability distribution, we have to renormalize, which already shows that modelling this operation introduces sub-stochastic matrices to the computation. In the next lemma normalization involves the costly computation of a marginal $P_{|A}$ (the probability that all places in A are set to b), however omitting the normalization factor will give us a sub-stochastic matrix and we will later show how such sub-stochastic matrices can be removed, in many cases avoiding the full costs of a marginal computation.

▶ **Lemma 14.** *It holds that* $\mathrm{ass}_{A,b}(P) = \frac{1}{P_{|A}} \left(\bigotimes_{i=1}^{m} T_{A,b}^{\mathrm{ass}}(i) \right) \cdot P$ *with* $P_{|A} = (\bigotimes_{i=1}^{m} Q_A(i)) \cdot P$ *where* $T_{A,b}^{\mathrm{ass}}(i)$ *is* $F_{1,1-b}$ *if* $i \in A$, *and* id *otherwise. We require that* $P_{|A} \neq 0$. *Furthermore* $Q_{A,b}^{\mathrm{ass}}(i) = \begin{pmatrix} 1 & 0 \end{pmatrix}$ *if* $i \in A$ *and* $\top$ *otherwise.*

In contrast to set and ass, the nas operation couples all involved places in A. Asserting that at least one place has no token means that once the observer learns that e.g. one particular place definitely has a token it affects all the other ones. Thus for updating the distribution we have to pass the wires of places A through another matrix that removes the possibility of all places containing a token and renormalizes.

▶ **Lemma 15.** *The following characterization holds:* $\mathrm{nas}_{A,1}(P) = \frac{1}{P_{|A}^c} (F_{k,1} \otimes \mathrm{id}_{m-k}) \cdot P$ *with* $P_{|A}^c = 1 - P_{|A}$ *($P_{|A}$ is defined as in Lemma 14). We require that* $P_{|A}^c \neq 0$.

An analogous result holds for $\mathrm{nas}_{A,0}$ by using $F_{k,0}$.

The previous lemmas determine how to update an MBN (B, e) to incorporate the changes to the encoded distribution stemming from the operations on the CNU. We denote the updated MBN by (B', e') with $B' = (V', \ell', s', \mathrm{out}')$.

For the $\mathrm{set}_{A,b}$ operation Lemma 13 shows that we have to add a new node v_s and a new generator g_s for each $s \in A$. We set $\ell(v_s) = g_s$ and $e'(g_s) = \mathbf{1}_b \cdot \top = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$, $s'(v_s) = \mathrm{out}(o_s)$ and $\mathrm{out}'(o_s) = v_s$. Similarly, this holds for the ass operation with the only difference that the associated matrix for each v_s is $\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ (cf. Figure 6).

For the $\mathrm{nas}_{A,b}$ operation Lemma 15 defines a usually larger matrix $F_{k,b} : k \to k$ that intuitively couples the random variables for all places in A. We cannot simply add a node to the MBN which evaluates to $F_{k,b}$ since nodes in the MBN always have to be of type $n \to 1$. However, one can show (see Lemma 18) that for each $F_{k,b}$-matrix, there exists an MBN (B', e') such that $M_{e'}(B')$. This can then be appended to (B, e) which has the same affect as appending a single node with the $F_{k,b}$-matrix.

Simplifying MBNs to OBNs

The characterisations of operations above ensure that updated MBNs correctly evaluate to the updated probability distributions. However, rather than OBNs we obtain MBNs where the complexity of updates is hidden in newly added nodes. Evaluating such MBNs is computationally more expensive because of the additional nodes. Below we show how to simplify the MBN, minimising the number of nodes either after each update or (in a lazy mode) after several updates.

As a first step we provide a lemma that will feature in all following simplifications. It states that every matrix can be expressed by the composition of two matrices.

▶ **Lemma 16** (Decomposition of matrices). *Given a matrix P of type $n \to m$ and a set of $k < m$ outputs – without loss of generality we pick $\{m - k + 1, \ldots, m\}$ – there exist two matrices $P^\vdash : n \to m - k$ and $P^\dashv : n + m - k \to k$ such that*

$$(\mathrm{id}_{m-k} \otimes P^\dashv) \cdot ((\nabla_{m-k} \cdot P^\vdash) \otimes \mathrm{id}_n) \cdot \nabla_n = P,$$
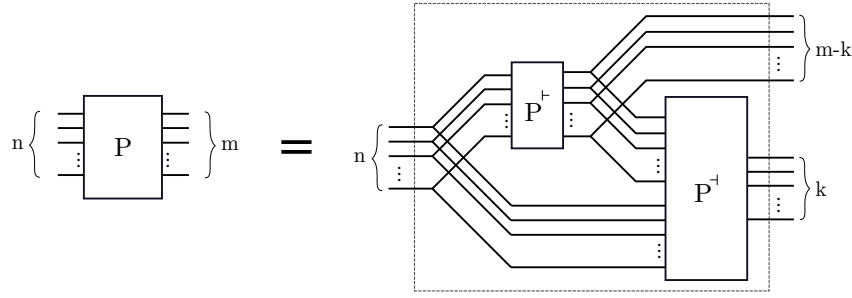
Figure 7 Schematic string diagram depiction of the decomposition of matrices.

which is visualized in Figure 7. Moreover, the matrices can be chosen so that $P^{\dashv}$ is stochastic and $P^{\vdash}$ sub-stochastic. If P is stochastic $P^{\vdash}$ can be chosen to be stochastic as well.

We can now deduce the known special case of arc reversal in OBN, stated e.g. in [4].

▶ **Corollary 17** (Arc reversal in OBNs). *Let (B, e) be an OBN with $B = (V, \ell, s, \text{out})$ and two nodes $u, y \in V$, where u is a direct predecessor of y, i.e. $u \in \text{pred}(y)$. Then there exists an OBN (B', e') with $B' = (V, \ell', s', \text{out})$, evaluating to the same probability distribution, where $\ell'(v) = \ell(v)$, $s'(v) = s(v)$ if $v \neq u$ and $v \neq y$ and $y \in \text{pred}(u)$. Thus the dependency between u and v is reversed.*

Arc reversal comes with a price: as can be seen in the proof, if u is associated with a matrix $P_u : n \to 1$ and y with a matrix $P_y : m + 1 \to 1$, then we have to create new matrices $P'_u : m + n + 1 \to 1$ and $P'_y : m + n \to 1$, causing new dependencies and increasing the size of the matrix. Hence arc reversal should be used sparingly.

After arc reversal a node might have duplicated inputs, which can be resolved by multiplying the corresponding matrix with ∇, thus reducing the dimension.

Next, we can use Lemma 16 to show that every matrix can be represented as an MBN. This MBN can always be built in a "minimal" way in that only m nodes are needed to represent a $n \to m$ matrix.

▶ **Lemma 18.** *Let $M : n \to m$ be a (sub-stochastic) matrix. Then there exists an MBN (B, e) with $B = (V, l, s, \text{out})$ such that $M = M_e(B)$, $|V| = m$ and out is a bijection. Moreover, if M is stochastic we can guarantee that $e(l(v))$ is stochastic for all $v \in V$. If M is sub-stochastic we can guarantee that v_{front} – the first node in a topological ordering of all nodes V' – is the only node where $e(l(v))$ is sub-stochastic, all other nodes have stochastic matrices.*

▶ **Corollary 19.** *Let (B, e) be an MBN without inputs and assume that $M_e(B)$ is stochastic. Then there exists an OBN (B', e') such that $M_e(B) = M_{e'}(B')$.*

Proof. The result follows trivially from the assumptions because for a stochastic MBN without input ports $M_e(B)$ is simply a column vector holding a probability distribution. It is well known that every probability distribution can be represented by some (ordinary) Bayesian net. Alternatively the result follows directly from Lemma 18. ◀

We just argued that every MBN can be simplified so that it does not contain any unnecessary nodes and at most one sub-stochastic matrix. However, while Lemma 18 shows that these simplifications are always possible it is not helpful in practice: in fact in the proof we take the full matrix represented by an MBN and then split it into (coupled) single nodes. Since we chose to use MBNs in order not to deal with large distribution vectors in the first place, this approach is not practical. Instead, in the following we will describe methods which allow us to simplify an MBN without computing the matrix first.
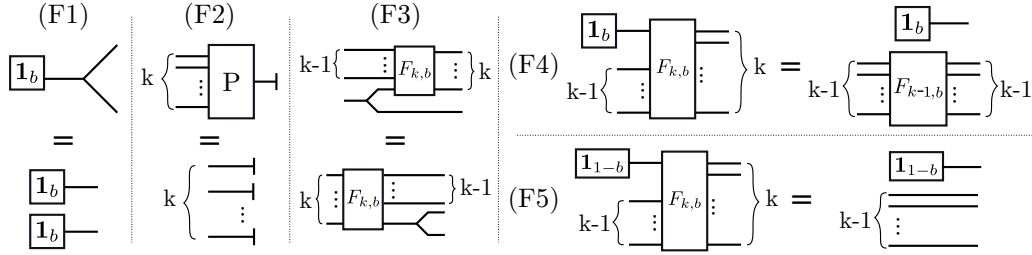
Figure 8 Equalities on sub-stochastic matrices. Note that (F2) holds only if P is stochastic and for (F4) and (F5) we have to assume $k > 1$.

First note that MBNs stemming from CNU operations can contain substructures that can locally be replaced by simpler ones. They are depicted in Figure 8.

▶ **Lemma 20.** *The equalities of Figure 8 hold for (sub-)stochastic matrices.*

As a result, it makes sense to first eliminate all of these substructures. Then there are two issues left to obtain an OBN. First, there are nodes that lost their direct connection with an output port (since output ports were terminated in a set operation or since we added an $F_{k,b}$-matrix). Those have to be merged with other nodes. Second, there are sub-stochastic matrices that have to be eliminated as well. The following lemma states that a node not connected to output ports can be merged with its direct successor nodes. This can introduce new dependencies between these successor nodes, but we remove one node from the network.

▶ **Lemma 21.** *lem Let $B = (V, \ell, s, \mathrm{out})$ be a causality graph, e an evaluation function such that (B, e) is an MBN. Assume that a node $v_0 \in V$ is not connected to an output port, i.e. for all $i \in \{1, \ldots, m\} : v_0 \neq \mathrm{out}(o_i)$, and $e(\ell(v_0))$ is stochastic. Then there exists an MBN (B', e') with $B' = (V \backslash \{v_0\}, \ell', s', \mathrm{out})$ such that $M_e(B) = M_{e'}(B')$. Moreover, $e' \circ \ell'|_{\bar{V}} = e \circ \ell|_{\bar{V}}$ and $s'|_{\bar{V}} = s|_{\bar{V}}$ where $\bar{V} = V \backslash (\{v_0\} \cup \mathrm{succ}(v_0))$.*

The conditions on ℓ' and s' mean that the update on B is local as it does not affect the whole network. Only the direct successors of v_0 are affected.

Finally, we have to get rid of sub-stochastic matrices inside the MBN, which have been introduced by the ass and nas operations (we assume that we did not normalize yet). The idea is to exchange nodes labelled with sub-stochastic matrices with the predecessor nodes and move them to the front (as in Lemma 18). Once there, normalization is straightforward by normalizing the vectors associated to these nodes.

▶ **Lemma 22.** *Let $B = (V, \ell, s, \mathrm{out})$ be a causality graph without input ports, i.e. of type $0 \to m$, e an evaluation function such that (B, e) is an MBN. Furthermore we require that there is a one-to-one correspondence between output ports and nodes, i.e., out is a bijection.*

Assume that $V' \subseteq V$ is the set of all nodes equipped with sub-stochastic matrices, i.e. $e(\ell(v))$ is sub-stochastic for all $v \in V'$. Then there exists an OBN (B', e') with $B' = (V, \ell', s', \mathrm{out})$ such that $M_e(B) = M_{e'}(B') \cdot p_B$ where $p_B = \top_m \cdot M_e(B) \leq 1$ is the probability mass of B. Moreover, $e' \circ \ell'|_{\bar{V}} = e \circ \ell|_{\bar{V}}$ and $s'|_{\bar{V}} = s|_{\bar{V}}$ where $\bar{V} = V \backslash (V' \cup \mathrm{pred}^*(V'))$.

Note that $\frac{1}{p_B}$ (whenever $p_B \neq 0$) is the normalization factor that can be obtained by terminating all input ports of B. We do not have to compute p_B explicitly, but it can be derived from the probabilities of the nodes which have been moved to the front (see proof).
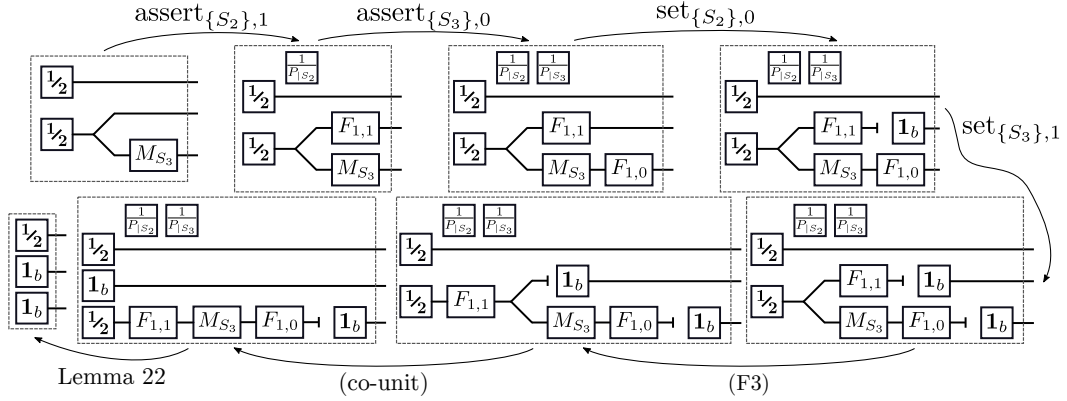
Figure 9 Exemplary update process for the success_{t_4} operation of our running CNU example. Here $\mathbf{1/2} = \begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix}$ and $M_{S_3} = \begin{pmatrix} 1/3 & 1/2 \\ 2/3 & 1/2 \end{pmatrix}$.

▶ **Corollary 23.** *Let $B = (V, \ell, s, \text{out})$ be a causality graph without input ports, i.e. of type $0 \to m$, e an evaluation function such that (B, e) is an OBN. Let $P = M_e(B)$.*

Then we can construct OBNs representing $\text{set}_{A,b}(P), \text{ass}_{A,b}(P), \text{nas}_{A,b}(P)$, *where*

- *the* set *operation modifies only* $\{\text{out}(o_i) \mid i \in A\}$ *and their direct successors and*
- *the* ass *and* nas *operations modify only* $\{\text{out}(o_i) \mid i \in A\}$ *and their predecessors.*

The operations are costly whenever a node has many predecessors or direct successors. In a certain way this is unavoidable because our operations are related to the computation of marginals, which is NP-hard [6]. However, if the Bayesian network has a comparatively "flat" structure, we expect that the efficiency is rather high in the average case, as supported by our runtime results below. Applying the nas operation will introduce dependencies for the random variables corresponding to the pre- and post-conditions of a transition, however this effect is localized if we consider particular classes of Petri nets, such as free-choice nets [9].

▶ **Example 24.** *Figure 9 shows an update process, following a lazy evaluation strategy, for a Bayesian net representing the probability distribution from Figure 2.*

5 Implementation

In order to quantitatively assess the performance of MBNs we developed a prototypical C++ implementation of the concepts in this paper, allowing to read, write, simplify, generate, and visualize MBNs as well as perform operations on CNUs that update an underlying MBN. The implementation is open-source and freely available on GitHub.[4]

As a first means of obtaining runtime results we randomly generated CNs with a range of different parameters: e.g. number of places, number of places in a precondition of a transition, places in the initial marking etc. We then successively picked transitions at random to fire and performed the necessary operations to update the MBN and simplify it to an OBN.

We chose to guarantee a success rate of transition firing of around 1/3. We argue that given the fact that we model an observer with prior knowledge it is realistic to assume a certain rate of successful transitions. A very low success rate leads to an accumulation

---

[4] https://github.com/bencabrera/bayesian_nets_program
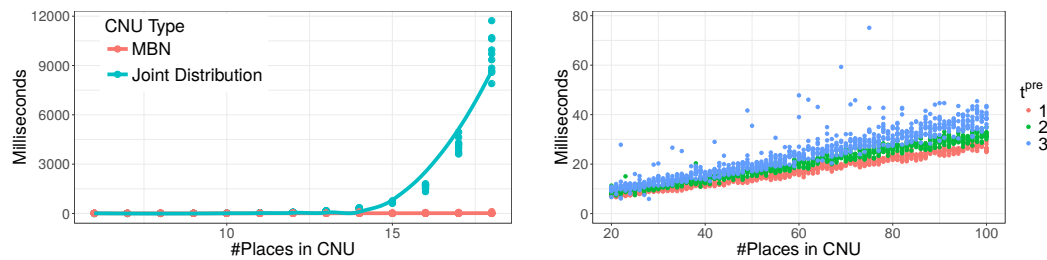
Figure 10 Averaged runtimes for performing 100 CNU operations using joint distributions or MBNs.

of successive $F_{k,b}$ matrices which can only be eliminated using the costly operations on substochastic matrices (see proof of Lemma 22). One could implement effective simplification strategies merging successive $F_{k,b}$ matrices – since composing 0,1 diagonal matrices yields again 0,1 diagonal matrices. However, this is out of scope of this publication.

The plot on the left of Figure 10 shows a comparison between run times when performing CNU operations directly on the joint distribution versus our MBN implementation. One can clearly observe the exponential increase when using the joint distribution while the MBN implementation in this setup stays relatively constant. The plot on the right of Figure 10 hints towards an increase in complexity when CNs – and thus MBNs – are more coupled. When increasing the maximum number of places in the precondition of a transition we observe an increase in run times. The number of outliers with a dramatic increase in run times seem to rise as well.

6   Conclusion

Related work.   A concept similar to our nets with uncertainty has been proposed in [18], but without any mechanism for efficiently representing and updating the probability distribution. There are also links to Hidden Markov Models [25] for inferring probabilistic knowledge on hidden states by observing a model.

Bayesian networks were introduced by Pearl in [22] to graphically represent random variables and their dependencies. Our work has some similarities to his probabilistic calculus of actions (do-calculus) [23] which supports the empirical measurement of interventions. However, while Pearl's causal networks model describe true causal relationships, in our case Bayesian networks are just compact symbolic representations of huge probability distributions. There is also a notion of dynamic Bayesian networks [21], where a random variable has a separate instance for each time slice. We instead keep only one instance of every random variable, but update the BN itself. There is substantial work on updating Bayesian networks (for instance [15]) with the orthogonal aim of learning BNs from training data.

PROPs have been introduced in [20], foundations for term-based proofs have been studied in [19] and their graphical language has been developed in [27, 5]. Bayesian networks as PROPs have already been studied in [12] under the name of causal theories, as well as in [17, 16] in order to give a predicate/state transformer semantics to Bayesian networks. However, these papers do not explicitly represent the underlying graph structure and in particular they do not consider updates of Bayesian networks.

We use the results from [7] in order to show that our causality graphs are in fact term graphs, which are freely generated gs-monoidal categories, which in turn are CC-structured PROPs. Although this result is intuitive, it is non-trivial to show: given two terms with

isomorphic underlying graphs, each can be reduced to a normal form which can be converted into each other using the axioms of a CC-structured PROP. Similar results are given in [11, 2] for PROPs with multiplication and unit, in addition to comultiplication and counit.

Future work.   We would like to investigate further operations on probability distributions, however it is unclear whether every operation can be efficiently implemented. For instance linear combinations of probability distributions seem difficult to handle.

Van der Aalst [29] showed that all reachable markings in certain free-choice nets can be inferred from their enabled transitions. An unrestricted observer may therefore be in a very strong position. Privacy research often considers statistical queries, such as how many records with certain properties exist in the database [10, 8]. To model such weaker queries we require labelled nets where instead of transitions we observe their labels. To implement this in BNs requires a disjunction of the enabledness conditions of all transitions with the same label. Furthermore we are interested in scenarios where certain transitions are unobservable.

References

1   L. Antova, C. Koch, and D. Olteanu. 10^(10^6) worlds and beyond: efficient representation and processing of incomplete information. *VLDB Journal*, 18(1021), 2009.

2   R. Bruni, F. Gadducci, and U. Montanari. Normal forms for algebras of connections. *Theoretical Computer Science*, 286(2):247–292, 2002.

3   B. Cabrera, T. Heindel, R. Heckel, and B. König. Updating probabilistic knowledge on Condition/Event nets using Bayesian networks, 2018. arXiv:1807.02566. `arXiv:1807.02566`.

4   A.Y.W. Cheuk and C. Boutilier. Structured arc reversal and simulation of dynamic probabilistic networks. In *Proc. of UAI '97 (Uncertainty in Artificial Intelligence)*, pages 72–79, 1997.

5   B. Coecke and A. Kissinger. *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning.* Cambridge University Press, 2017.

6   G.F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artif. Intell.*, 42(2-3):393–405, 1990. `doi:10.1016/0004-3702(90)90060-D`.

7   A. Corradini and F. Gadducci. An algebraic presentation of term graphs, via gs-monoidal categories. *Appl. Categor. Struct.*, 7:299–331, 1999.

8   M.L. Damiani. Location privacy models in mobile applications: conceptual view and research directions. *GeoInformatica*, 18(4):819–842, 2014. `doi:10.1007/s10707-014-0205-7`.

9   J. Desel and J. Esparza. *Free Choice Petri Nets*, volume 40 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, 1995.

10 C. Dwork. Differential privacy: A survey of results. In *Proc. of TAMC '08 (Theory and Applications of Models of Computation)*, pages 1–19. Springer, 2008. LNCS 4978.

11 M. Fiore and M. Devesas Campos. The algebra of directed acyclic graphs. In *Computation, Logic, Games, and Quantum Foundations. The Many Facets of Samson Abramsky*, pages 37–51. Springer, 2013. LNCS 7860.

12 B. Fong. Causal theories: A categorical perspective on Bayesian networks. Master's thesis, University of Oxford, 2012. arXiv:1301.6201.

13 B. Fong and D. I Spivak. Seven Sketches in Compositionality: An Invitation to Applied Category Theory. *ArXiv e-prints*, 2018. `arXiv:1803.05316`.

14 N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.

15   N. Friedman and M. Goldszmidt. Sequential update of bayesian network structure. In Dan Geiger and Prakash Shenoy, editors, *Proc. of UAI '97 (Uncertainty in Artificial Intelligence)*, pages 165–174, 1997.

16   B. Jacobs and F. Zanasi. A predicate/state transformer semantics for Bayesian learning. In *Proc. of MFPS*, volume 325 of *ENTCS*, pages 185–200, 2016.

17   B. Jacobs and F. Zanasi. A formal semantics of influence in Bayesian reasoning. In *Proc. of MFCS*, volume 83 of *LIPIcs*, pages 21:1–21:14, 2017.

18   I. Jarkass and M. Rombaut. Dealing with uncertainty on the initial state of a Petri net. In *Proc. of UAI '98 (Uncertainty in Artificial Intelligence)*, pages 289–295, 1998.

19   C. Barry Jay. Languages for monoidal categories. *Journal of Pure and Applied Algebra*, 59(1):61–85, 1989.

20   S. MacLane. Categorical algebra. *Bull. Amer. Math. Soc.*, 71(1):40–106, 1965.

21   K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, UC Berkeley, Computer Science Division, 2002.

22   J. Pearl. Bayesian networks: A model of self-activated memory for evidential reasoning. In *Proc. of the 7th Conference of the Cognitive Science Society*, pages 329–334, 1985. UCLA Technical Report CSD-850017.

23   J. Pearl. A probabilistic calculus of actions. In R. Lopez de Mantaras and D. Poole, editors, *Proc. of UAI '94 (Uncertainty in Artificial Intelligence)*, 1994.

24   J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

25   L. R. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

26   W. Reisig. *Petri Nets: An Introduction*. EATCS Monographs on Theoretical Computer Science. Springer-Verlag, Berlin, Germany, 1985.

27   P. Selinger. A survey of graphical languages for monoidal categories. In Bob Coecke, editor, *New Structures for Physics*, pages 289–355. Springer, 2011.

28   D. Suciu, D. Olteanu, C. Ré, and C. Koch. *Probabilistic Databases*. Morgan & Claypool Publishers, 2011.

29   W.M.P. van der Aalst. Markings in perpetual free-choice nets are fully characterized by their enabled transitions. In *Proc. of PN '18 (Petri Nets)*, pages 315–336. Springer, 2018. LNCS 10877.

30   F. Zanasi. *Interacting Hopf Algebras – the theory of linear systems*. PhD thesis, ENS Lyon, 2015.