Building Causal Interaction Models by Recursive Unfolding

Linda C. van der Gaag LINDA.VANDERGAAG@IDSIA.CH
Dalle Molle Institute for Artificial Intelligence Research, Lugano, Switzerland

Silja Renooij S.RENOOIJ@UU.NL
Department of Information and Computing Sciences, Utrecht University, The Netherlands

Alessandro Facchini ALESSANDRO.FACCHINI@IDSIA.CH
Dalle Molle Institute for Artificial Intelligence Research, Lugano, Switzerland

Abstract

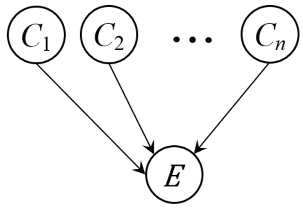
Causal interaction models, such as the well-known noisy-or and leaky noisy-or models, have become quite popular as a means to parameterize conditional probability tables for Bayesian networks. In this paper we focus on the engineering of subnetworks to represent such models and present a novel technique called *recursive unfolding* for this purpose. This technique allows inserting, removing and merging cause variables in an interaction model at will, without affecting the underlying represented information. We detail the technique, with the recursion invariants involved, and illustrate its practical use for Bayesian-network engineering by means of a small example.

Keywords: Bayesian-network engineering; Causal interaction models; Compositionality.

1. Introduction

Causal interaction models have become quite popular as a means to simplify the construction of conditional probability tables when building Bayesian networks for real-world applications. These models in essence describe the joint influence of a set of cause variables on a common effect variable by means of a parameterized conditional probability table for the latter variable, such that this table is fully defined by a small number of parameter probabilities. Various models specifying different types of causal interaction have been designed, with the noisy-or and leaky noisy-or models as the best known among them (Díez and Druzdzel, 2007; Henrion, 1989; Pearl, 1988).

When employing a causal interaction model to describe the joint influence of multiple cause variables on an effect variable, a Bayesian-network engineer typically is faced with a range of modelling decisions. While researchers have addressed the graphical representation of a noisy-or model (see for example Pearl (1988); Heckerman and Breese (1996); Renooij and van der Gaag (2019)) as well as the representation of the conditional probability table involved (see for example del Sagrado and Salmeron (2003)), little attention has focused thus far on the flexibility of causal interaction models from an engineering perspective. If all possible causes for an effect to arise have been fully identified and described in the state-of-art knowledge of a domain at hand for example, these may all be modelled individually through separate cause variables. As from a modelling perspective it may not be desirable to define a separate variable for each possible cause, the network engineer may decide to merge several causes into a single (binary) compound cause variable with a well-defined meaning. Alternatively, they may transform the noisy-or model into a leaky noisy-or model and define a leak to capture some causes that will not be modelled explicitly. Any such modelling decisions taken during the construction of a network at hand may later need to be reconsidered upon maintenance. Network engineers are offered very little support for such engineering tasks.



$\mathbf{c}$	$\Pr(\bar{e} \mid \mathbf{c})$ *basic*	*leaky*
$\bar{c}_1, \bar{c}_2, \bar{c}_3$	1	
$c_1, \bar{c}_2, \bar{c}_3$	$(1-p_1)$	
$\bar{c}_1, c_2, \bar{c}_3$	$(1-p_2)$	
$\bar{c}_1, \bar{c}_2, c_3$	$(1-p_3)$	$\cdot\ (1-p_L)$
$c_1, c_2, \bar{c}_3$	$(1-p_1)\cdot(1-p_2)$	
$c_1, \bar{c}_2, c_3$	$(1-p_1)\cdot(1-p_3)$	
$\bar{c}_1, c_2, c_3$	$(1-p_2)\cdot(1-p_3)$	
c_1, c_2, c_3	$(1-p_1)\cdot(1-p_2)\cdot(1-p_3)$	

Figure 1: A conditional subnetwork $\mathcal{M}(n)$ with n cause variables C_i and the effect variable E (*left*); the conditional probability tables imposed by the basic noisy-OR and leaky noisy-OR models respectively, for $n = 3$ (*right*).

In this paper, we detail *recursive unfolding* as a technique for building and maintaining subnetwork representations of causal interaction models. Recursive unfolding is a practical engineering technique by which explicit causes can be inserted, removed or merged in an interaction model by means of simple transforms. In presenting our technique, we will consider interaction models that have an underlying decomposable deterministic function, and exploit the property that such models allow a cascading subnetwork representation (Renooij and van der Gaag, 2019). We will further focus on the leaky noisy-or model and develop an *unfold transform* for recursively extracting causes from the leak involved. Our transform has the prior distribution over the effect variable for its invariant, which guarantees that its application leaves all underlying represented information unchanged. The transform further has a range of convenient properties which render recursive unfolding a practical engineering technique.

The paper is organised as follows. In Sections 2 and 3 we review causal interaction models and extend upon their subnetwork representations, focusing specifically on the leaky noisy-or model. Section 4 introduces our technique of recursive unfolding and specifies the invariant of its main transform; properties of the transform are discussed in Section 5. The paper concludes in Section 6.

2. Preliminaries

We consider binary random variables, denoted by (possibly indexed) capital letters X; we will write $\bar{x}$ and x to denote absence and presence respectively, of the concept modelled by a variable X. Sets of variables are indicated by bold-face capital letters $\mathbf{X}$, with bold-face small letters $\mathbf{x}$ for their joint value combinations. We further consider joint probability distributions $\Pr$ over sets of variables, represented by Bayesian networks. Within such networks, a real-world mechanism of causes and their effects[1] is captured by a *conditional subnetwork* that represents the conditional distributions $\Pr(\mathbf{E} \mid \mathbf{C})$ over the effect variables $\mathbf{E}$ involved, conditioned on the cause variables $\mathbf{C}$ (Koller and Friedman, 2009). In this paper, we will address mechanisms with a single effect variable E and one or more cause variables C_i, $i = 1, \ldots, n$, $n \geq 1$. Such a mechanism is typically represented as a conditional subnetwork with the cause variables as parents of the effect variable, as illustrated in Figure 1 (*left*); we use $\mathcal{M}(n)$ to indicate this type of representation, with the argument n referring to the number of explicitly modelled causes. A *causal interaction model* for a conditional subnetwork

1. Although we do not make any claim with respect to causal interpretation, we adopt the terminology commonly used.

$\mathcal{M}(n)$ now takes the form of a parameterized conditional probability table $\Pr(E \mid \mathbf{C})$ over the effect variable E given the joint value combinations $\mathbf{c}$ for its parents $\mathbf{C}$. The *noisy-or model* (Pearl, 1988) for example, defines the conditional probability table for the effect variable E in $\mathcal{M}(n)$ through

- the conditional probability $\Pr(e \mid \bar{c}_1, \ldots, \bar{c}_n) = 0$;
- the *noisy-effect parameters* $p_i = \Pr(e \mid \bar{c}_1, \ldots, \bar{c}_{i-1}, c_i, \bar{c}_{i+1}, \ldots, \bar{c}_n)$, for all $i = 1, \ldots, n$;
- the *definitional rule* $\Pr(e \mid \mathbf{c}) = 1 - \prod_{i \in I_{\mathbf{c}}}(1 - p_i)$ for the value combinations $\mathbf{c}$ stating the presence of *two or more* causes, where $I_{\mathbf{c}}$ is the set of indices of the causes c_i present in $\mathbf{c}$.

Figure 1 (*right*) illustrates the parameterized conditional probability table defined by the noisy-or model, for *absence* of the effect, in a conditional subnetwork with three cause variables.

The noisy-or model assumes that all possible causes for the effect to arise, are modelled explicitly. This assumption is weakened by the *leaky noisy-or model*, which allows for the existence of one or more *unmodelled causes*. These unmodelled causes are jointly referred to as the *leak cause*. With this leak cause, the leaky noisy-or model includes a *leak parameter* p_L to capture the probability of the effect e occurring in the absence of all explicitly modelled causes. In view of the semantics of this leak parameter, different interpretations of the noisy-effect parameters per cause have given rise to different definitional rules for the entries in the probability table for the effect variable (Díez and Druzdzel, 2007; Henrion, 1989). In this paper we adopt the interpretation proposed by Díez and Druzdzel (2007), which takes a noisy-effect parameter p_i to reflect the probability $\Pr(e \mid \bar{c}_1, \ldots, \bar{c}_{i-1}, c_i, \bar{c}_{i+1}, \ldots, \bar{c}_n, \bar{l})$, with $\bar{l}$ indicating absence of the leak cause. The parameter p_i thus is taken to capture the probability of the effect occurring as a result of just the cause c_i, in the absence of all other modelled *and* unmodelled causes. For the conditional probability table for the effect variable E in $\mathcal{M}(n)$, the leaky noisy-or model further specifies

- the *leak parameter* $p_L = \Pr(e \mid \bar{c}_1, \ldots, \bar{c}_n)$;
- the *definitional rule* $\Pr(e \mid \mathbf{c}) = 1 - (1 - p_L) \cdot \prod_{i \in I_{\mathbf{c}}}(1 - p_i)$ for the value combinations $\mathbf{c}$ stating the presence of *one or more* explicitly modelled causes, with $I_{\mathbf{c}}$ as above.

Figure 1 (*right*) illustrates that each entry in the probability table for the absence of the effect in a noisy-or model is multiplied by the factor $(1 - p_L)$ to obtain the entry for its leaky variant. We note that the alternative interpretation by Henrion (1989) assumes a noisy-effect parameter to include the leak probability as such, thereby giving rise to a different definitional rule. As this rule describes an equally simple relation between the noisy-effect parameters, the technique presented in the sequel is readily adapted to this interpretation, although the details involved would be less elegant.

3. Causal Interaction Models as Cascading Subnetworks

Causal interaction models are often viewed as partitioned into a *deterministic function* f and independent *noise variables* Z_i per cause variable (see for example Heckerman and Breese (1996); Koller and Friedman (2009); Pearl (1988)). The basic idea is illustrated for the leaky noisy-or model by the subnetwork from Figure 2 (*left*). Although commonly left implicit, we have made the leak variable L explicit for explanatory purposes; as the leak cause is assumed to be present always, its prior probability is set to $\Pr(l) = 1$. In this partition view, the variables Z_i capture the noise of the influences by which the causes c_i give the effect e. These Z_i have associated the probabilities

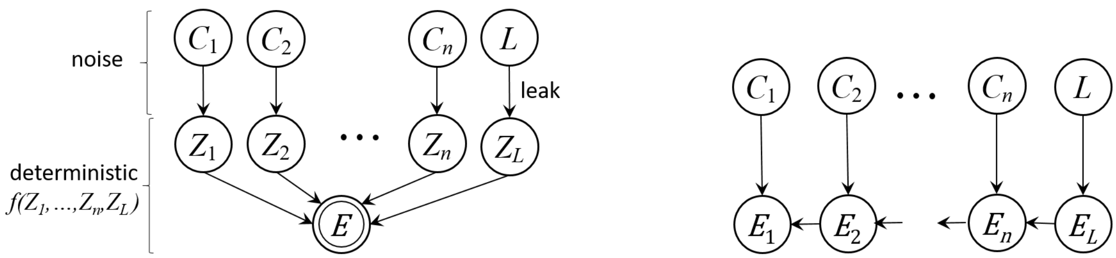


Figure 2: The partition view of a causal interaction model as a subnetwork composed of a probabilistic noise part and a deterministic functional part (*left*); the subnetwork in cascading representation $\mathcal{W}(n)$, derived from self-decomposability of the function f (*right*).

$\Pr(z_i \mid c_i) = p_i$, $\Pr(z_i \mid \bar{c}_i) = 0$, and $\Pr(z_L \mid l\,) = p_L$, $\Pr(z_L \mid \bar{l}\,) = 0$, where p_i, p_L are the parameters of the leaky noisy-or model. The deterministic function f in the partition view further equals the logical OR and is encoded in the conditional probability table $\Pr(E \mid \mathbf{Z})$ for the effect variable E by means of degenerate distributions, which in essence render E a deterministic variable.

Causal interaction models are typically represented by conditional subnetworks $\mathcal{M}(n)$ that are obtained by summing out the noise variables from the partition view (and absorbing the leak variable). If its underlying deterministic function f is self-decomposable[2] however, an interaction model can also be captured by a subnetwork in *cascading representation* (Renooij and van der Gaag, 2019). Figure 2 (*right*) shows this type of representation for the leaky noisy-or model, again with the leak variable made explicit; in the sequel, we will use $\mathcal{W}(n)$ to refer to a cascading subnetwork, with the argument n once more indicating the number of explicitly modelled causes. For the leaky noisy-or model, the conditional subnetwork $\mathcal{W}(n)$ includes n auxiliary effect variables E_i for the explicitly modelled cause variables C_i, and another such variable E_L for the leak variable. The conditional probability tables for the effect variables E_i, $i = 1, \ldots, n$, are defined as:

$$\begin{array}{rclcrcl} \Pr(e_i \mid \bar{c}_i, \bar{e}_{i+1}) & = & 0 & \qquad & \Pr(e_i \mid \bar{c}_i, e_{i+1}) & = & 1 \\ \Pr(e_i \mid c_i, \bar{e}_{i+1}) & = & p_i & \qquad & \Pr(e_i \mid c_i, e_{i+1}) & = & 1 \end{array}$$

with $E_{n+1} \equiv E_L$ and with p_i the noisy-effect parameter for the cause c_i as before. For the leak's effect variable E_L, the probability table has $\Pr(e_L \mid \bar{l}\,) = 0, \Pr(e_L \mid l\,) = p_L$, with p_L the model's leak parameter; we note that, as the leak is assumed to be present always, we have that $\Pr(e_L) = p_L$. The conditional probability tables for the effect variables in the cascading representation of the leaky noisy-or model are such that for all joint value combinations $\mathbf{c}$ for the cause variables, it holds that

$$\Pr_{\mathcal{M}}(e \mid \mathbf{c}) \;=\; \Pr_{\mathcal{W}}(e_1 \mid \mathbf{c})$$

where $\Pr_{\mathcal{M}}$ is the distribution defined over the variables $E, C_1, \ldots, C_n, L$ by the subnetwork $\mathcal{M}(n)$, and $\Pr_{\mathcal{W}}$ is the distribution over the variables $E_1, \ldots, E_n, E_L, C_1, \ldots, C_n, L$ in the cascading subnetwork $\mathcal{W}(n)$. Through this property, the two subnetwork representations are equivalent in terms of the conditional distributions over the variables E and E_1 respectively, given any value combination $\mathbf{c}$; we refer to Renooij and van der Gaag (2019) for a proof of this equivalence.

2. A function $f(\cdot)$ on a set of entities is called *self-decomposable* if, for any two disjoint subsets $\mathbf{X}, \mathbf{Y}$, the property $f(\mathbf{X} \cup \mathbf{Y}) = f(\mathbf{X}) \diamond f(\mathbf{Y})$ holds, for some commutative and associative merge operator $\diamond$ (cf. Jesus et al. (2011)).

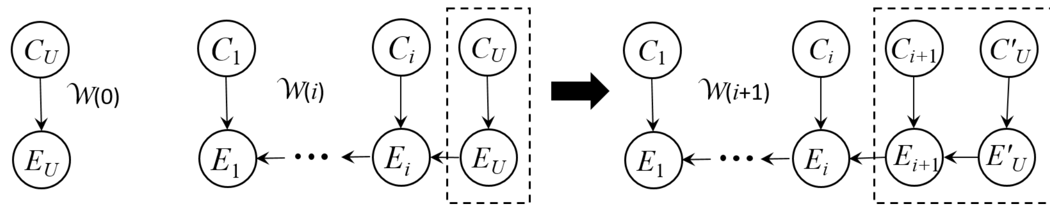


Figure 3: *Recursive unfolding*: the initial leak construct $\mathcal{W}(0)$ (*left*), and the recursive step from a representation $\mathcal{W}(i)$ to a representation $\mathcal{W}(i+1)$ of the leaky noisy-OR model (*right*).

4. Recursive Unfolding

In this section, we detail our novel technique of *recursive unfolding* for building cascading representations of causal interaction models. As will be argued in Section 5, recursive unfolding is a practical engineering technique by which explicit causes can be inserted, removed or merged in an interaction model by means of simple transforms. Focusing once again on the leaky noisy-or model, we present the technique itself in Section 4.1; in Section 4.2, we will discuss the invariant underlying the recursive scheme and thereby prove the technique's correctness.

4.1 The Recursive Scheme

Our technique of recursive unfolding of leaky noisy-or models builds on the idea of maintaining all yet unmodelled causes and their joint effect in an *auxiliary leak construct*, from which separate causes are extracted and modelled explicitly. The technique is initialised with a subnetwork $\mathcal{W}(0)$ composed of just this leak construct, as shown in Figure 3 (*left*). The probability tables associated with the variables in this subnetwork are defined by $\Pr(c_U) = 1$ for the leak variable C_U and by

$$\Pr(e_U \mid \bar{c}_U) \;=\; 0 \qquad \Pr(e_U \mid c_U) \;=\; p_U$$

for the leak's effect variable E_U, where p_U is an overall leak parameter to be specified by the network engineer; we will elaborate on this choice of parameter probabilities in Section 4.2. Starting with the auxiliary leak construct, recursive unfolding amounts to recursively extracting separate causes from the leak variable C_U and including these explicitly into the cascading representation under construction. With its underlying idea depicted in Figure 3 (*right*), the recursive step of taking a representation $\mathcal{W}(i)$ to a representation $\mathcal{W}(i+1)$ for any $i \geq 0$, now amounts to replacing the leak construct from $\mathcal{W}(i)$ by the cascading combination of the following two constructs:

- the construct composed of the *newly extracted cause variable* C_{i+1} with an arc to its associated *new effect variable* E_{i+1}, where
  - the probability $\Pr(c_{i+1})$ of the cause c_{i+1} being present is set by the network engineer;
  - the conditional probability table $\Pr(E_{i+1} \mid C_{i+1}, E'_U)$, with E'_U the effect variable of the new leak construct detailed below, is defined by

$$\begin{array}{lll} \Pr(e_{i+1} \mid \bar{c}_{i+1}, \bar{e}'_U) = 0 & \qquad & \Pr(e_{i+1} \mid \bar{c}_{i+1}, e'_U) = 1 \\ \Pr(e_{i+1} \mid c_{i+1}, \bar{e}'_U) = p_{i+1} & \qquad & \Pr(e_{i+1} \mid c_{i+1}, e'_U) = 1 \end{array}$$

    where p_{i+1} is the noisy-effect parameter associated with cause c_{i+1}, to be set by the network engineer;

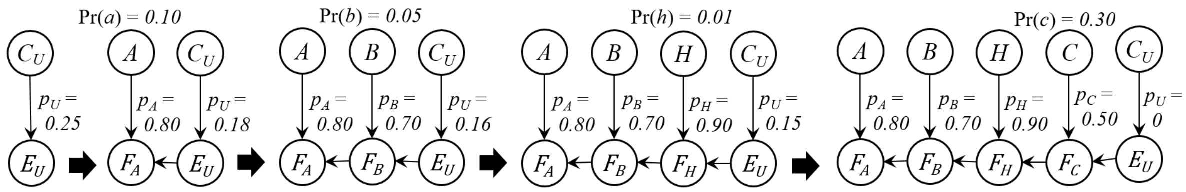


Figure 4: Building a cascading representation of an example noisy-OR model through unfolding.

- the new probability table $\Pr(E_i \mid C_i, E_{i+1})$ for the variable E_i is derived from $\Pr(E_i \mid C_i, E_U)$ by replacing each occurrence of e_U and $\bar{e}_U$ by e_{i+1} and $\bar{e}_{i+1}$, respectively.

- the leak construct composed of the *new leak variable* C'_U with an arc to its associated *new effect variable* E'_U, where
  - the prior probability $\Pr(c'_U)$ again is set to 1;
  - the conditional probability table $\Pr(E'_U \mid C'_U)$ is defined by $\Pr(e'_U \mid \bar{c}'_U) = 0$, $\Pr(e'_U \mid c'_U) = p'_U$, with the new leak parameter p'_U computed through the *leak updating rule*:

$$p'_U = \frac{p_U - p_{i+1} \cdot \Pr(c_{i+1})}{1 - p_{i+1} \cdot \Pr(c_{i+1})}$$

where p_U is the leak probability taken from $\mathcal{W}(i)$.

The recursive step detailed above will be called the *unfold transform*. We note that the transform involves two parameters to be set by the network engineer, which are the prior probability $\Pr(c_{i+1})$ and the noisy-effect parameter p_{i+1}. While p_{i+1} is a parameter of the interaction model being represented, the prior probability $\Pr(c_{i+1})$ of the newly inserted cause c_{i+1} is not. The latter is necessary information however, for computing the noisy effect of the remaining leak. If the resulting subnetwork is to be embedded in a larger Bayesian network moreover, this prior probability acts as a constraint on the conditional probability table to be included for the new cause variable. We further note that recursive unfolding has a natural *fixed point*. With each step, the leak parameter is decreased yet cannot become smaller than zero. It will reach zero when, from the remaining leak variable C_U, a new cause variable C_n is extracted with $p_n \cdot \Pr(c_n) = p_U$. The fixed point signifies that all possible causes for the effect to arise have been made explicit. The constructed subnetwork then in fact represents a non-leaky noisy-or model, and its leak construct can be removed.

We illustrate our recursive-unfolding scheme by means of a fictitious medical example.

Example 1 *We consider four possible causes of a fever and construct, through recursive unfolding, a cascading representation of a noisy-or model to capture their joint effect. The transform steps involved are shown in Figure 4. Unfolding is initialised with an auxiliary leak construct, as described above. As, from our medical context, we find the prior probability of a fever to be* 0.25*, we take this probability as the overall leak parameter* p_U *in the initial construct. The first cause to be modelled explicitly is a virus infection of type A. Our initial representation is transformed to include the new cause variable* A *describing the infection and its associated effect variable* F_A*. For the variable A, we acquire the prior probability* $\Pr(a) = 0.10$ *and the noisy-effect parameter* $p_A = 0.80$*. This information results in a remaining leak parameter of* $(0.25 - 0.80 \cdot 0.10)/(1 - 0.80 \cdot 0.10) \approx 0.18$*. We then identify a bacterial infection* B *as another possible cause of a fever and extract it from* *the prevailing leak variable. With the estimated prior probability* $\Pr(b) = 0.05$ *and noisy-effect parameter* $p_B = 0.70$ *for this type of infection, the leak parameter is further decreased, to roughly* 0.16*. Similarly extracting a heat stroke* H *as a possible cause of a fever, leaves a leak parameter of approximately* 0.15*. We now establish a virus infection of type* C *as the only remaining possible cause: its prior probability* $\Pr(c) = 0.30$ *and noisy-effect parameter* $p_C = 0.5$ *reduce the leak parameter to zero. The auxiliary leak construct can thus be removed from the representation without changing the joint distribution over the four cause variables and associated effect variables.* □

4.2 The Invariant of Recursive Unfolding

Our recursive scheme of unfolding was designed such that the unfold transform does not change the semantics of the represented information. More specifically, it was designed such that the prior probability of the overall effect to occur is not affected by the process of extracting cause variables from the leak variable and making them explicit. To show that this property is an invariant of the transform, we first detail in Section 4.2.1 the semantics of the leak construct and then show in Section 4.2.2 that the prior probability of the overall effect indeed remains the same upon unfolding.

4.2.1 SEMANTICS OF THE LEAK CONSTRUCT

After each step of recursive unfolding, the resulting subnetwork $\mathcal{W}(i)$ represents a leaky noisy-or model with the i explicitly modelled cause variables $C_1, \ldots, C_i$ and the leak variable C_U, along with their associated effect variables. The leak variable C_U captures all possible causes of the common effect to arise, other than $c_1, \ldots, c_i$. Some of these causes may simply be unmodelled and some may actually be unknown in a domain's state-of-the-art knowledge. More formally, the variable C_U is logically equivalent to $C_{i+1} \vee \ldots \vee C_n \vee \ldots \vee C_m$, where $C_{i+1}, \ldots, C_n$ capture the unmodelled, yet established causes and $C_{n+1}, \ldots, C_m$, for some suitably chosen $m > n$, describe unknown causes of the effect to arise. From this equivalence, we thus have that

$$\begin{aligned} c_U &= c_{i+1} \vee \ldots \vee c_n \vee \ldots \vee c_m \\ \bar{c}_U &= \bar{c}_{i+1} \wedge \ldots \wedge \bar{c}_n \wedge \ldots \wedge \bar{c}_m \end{aligned}$$

We now observe that extraction of the cause variable C_{i+1} from the leak variable C_U in essence changes the meaning of the latter variable to $C'_U \equiv C_{i+2} \vee \ldots \vee C_m$. In view of this changed meaning, the leak parameter p_U associated with C_U has to be adapted, to a parameter p'_U, to properly reflect the noisy effect of the logical combination $c_{i+2} \vee \ldots \vee c_m$ of the remaining implicit causes.

Before detailing the changes in representation that are induced by the unfold transform, we consider the leak construct $\mathcal{W}(0)$, composed of the leak variable C_U and its associated effect variable E_U, and address the choice of initial values for the prior probability $\Pr(c_U)$ and the leak parameter $\Pr(e_U \mid c_U)$. Since by the semantics of the leaky noisy-or model we have that $\Pr(e_U \mid \bar{c}_U) = 0$, we find for the prior probability $\Pr(e_U)$ of the effect to arise that

$$\Pr(e_U) = \Pr(e_U \mid c_U) \cdot \Pr(c_U)$$

From the three probabilities involved, $\Pr(e_U)$ is the easiest to acquire in practice, either from observational data or from experts; the other two probabilities would be quite difficult to obtain as they are associated with multiple implicit and, in part, unknown causes. The true values of $\Pr(c_U)$ and $\Pr(e_U \mid c_U)$ however, must be such that their product equals the acquired probability $\Pr(e_U) = p_L$.

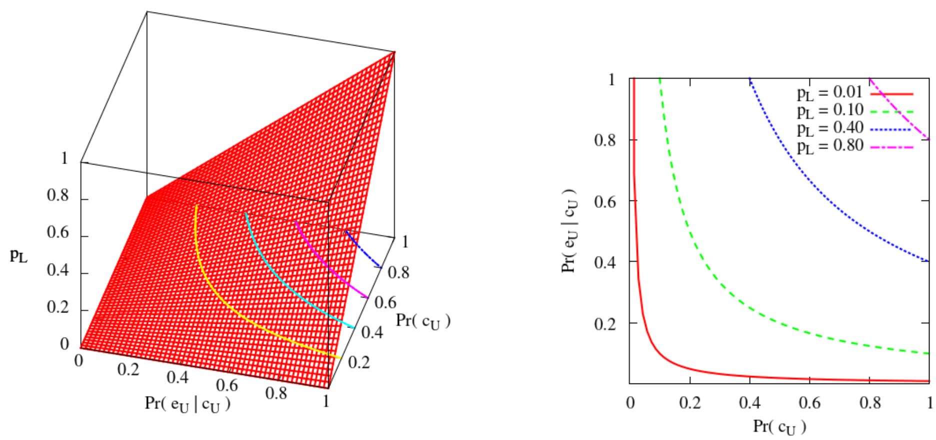


Figure 5: The prior probability $\Pr(e_U) = p_L$ as a function of the parameters $\Pr(c_U)$ and $\Pr(e_U \mid c_U)$ (*left*); $\Pr(e_U \mid c_U)$ as a function of $\Pr(c_U)$ for different values of p_L (*right*).

To illustrate the relation between the three probabilities, Figure 5 (*left*) shows the probability p_L as a function of $\Pr(c_U)$ and $\Pr(e_U \mid c_U)$; Figure 5 (*right*) shows the relation between the latter two probabilities, for different values of p_L. For inclusion in the auxiliary leak construct, convenient initial values must be chosen for $\Pr(c_U)$ and $\Pr(e_U \mid c_U)$. For this purpose, we turn once more to the effect of extracting the cause variable C_1 from the leak variable C_U in $\mathcal{W}(0)$. After the extraction, the variable C_U has been substituted by the new leak variable C'_U in $\mathcal{W}(1)$. The prior probability $\Pr(c'_U)$ to be associated with this new variable is dependent of the probability $\Pr(c_U)$ and the prior probability $\Pr(c_1)$ of the extracted cause being present. Assuming that all cause variables captured by the leak variable are mutually independent, we find that $\Pr(c'_U)$ relates to these probabilities as

$$\Pr(c'_U) = \frac{\Pr(c_U) - \Pr(c_1)}{1 - \Pr(c_1)}$$

In the design of (the variants of) the original leaky noisy-or model, the values for $\Pr(c_U)$ and $\Pr(e_U \mid c_U)$ were chosen pragmatically, setting either of them to 1 and thereby effectively reducing the freedom of choice to a single parameter (Díez and Druzdzel, 2007; Zagorecki, 2010). In the design of our unfold transform, we have taken a similar approach and chose to set $\Pr(c_U) = 1$ in the initial leak construct. The prior probability of the leak being present thereby becomes invariant and the computations involved after extracting a new cause variable are restricted to just an update of the leak parameter $\Pr(e_U \mid c_U)$. We note that if the cause variables cannot be assumed mutually independent apriori, the above formula needs to account for the dependencies involved and would thereby become less elegant.

4.2.2 INVARIANCE OF THE PRIOR PROBABILITY OF THE EFFECT

As mentioned above, we designed the unfold transform such that its application does not affect the overall semantics of the underlying represented information. More formally, we imposed the property $\Pr(e) = \Pr'(e)$ for the main invariant of the transform, where $\Pr$ is the probability distribution defined over the subnetwork $\mathcal{W}(i)$ prior to applying the transform and $\Pr'$ is the distribution over the subnetwork $\mathcal{W}(i+1)$ resulting from its application. We now show that the unfold transform as detailed in Section 4.1 indeed satisfies the above stated property.

We begin by showing that the property $\Pr(e) = \Pr'(e)$ holds upon application of the unfold transform to the subnetwork $\mathcal{W}(0)$, that is, to the initial auxiliary leak construct. We recall that in this subnetwork we have that $\Pr(e) \equiv \Pr(e_U) = p_U$. In the subnetwork $\mathcal{W}(1)$ resulting from the transform, with $\Pr'(e) \equiv \Pr'(e_1)$, we then have that

$$\begin{aligned}\Pr'(e_1) \;=\; & \Pr'(e_1 \mid c_1, e'_U) \cdot \Pr'(c_1) \cdot \Pr'(e'_U) + \Pr'(e_1 \mid \bar{c}_1, e'_U) \cdot \Pr'(\bar{c}_1) \cdot \Pr'(e'_U) \\ & + \Pr'(e_1 \mid c_1, \bar{e}'_U) \cdot \Pr'(c_1) \cdot \Pr'(\bar{e}'_U) + \Pr'(e_1 \mid \bar{c}_1, \bar{e}'_U) \cdot \Pr'(\bar{c}_1) \cdot \Pr'(\bar{e}'_U)\end{aligned}$$

Filling in the parameters as prescribed by the unfold transform, gives

$$\begin{aligned}\Pr'(e_1) &= \Pr'(c_1) \cdot \Pr'(e'_U) + \Pr'(\bar{c}_1) \cdot \Pr'(e'_U) + p_1 \cdot \Pr'(c_1) \cdot \Pr'(\bar{e}'_U) \\ &= p_1 \cdot \Pr'(c_1) + (1 - p_1 \cdot \Pr'(c_1)) \cdot p'_U\end{aligned}$$

using $p'_U = \Pr'(e'_U)$ in the latter step. From the transform's leak updating rule, we then find

$$\Pr'(e_1) \;=\; p_1 \cdot \Pr'(c_1) + (1 - p_1 \cdot \Pr'(c_1)) \cdot \frac{p_U - p_1 \cdot \Pr'(c_1)}{1 - p_1 \cdot \Pr'(c_1)} \;=\; p_U$$

from which we have that the invariant $\Pr(e) = \Pr'(e)$ holds upon applying the transform to $\mathcal{W}(0)$.

The above argument is readily generalised to application of the unfold transform to any subnetwork $\mathcal{W}(i)$, $i > 0$. To this end, we first recall that the equivalence $\Pr(e) \equiv \Pr(e_1)$ holds in all subnetworks $\mathcal{W}(i)$ with $i > 0$ (Renooij and van der Gaag, 2019). We now observe that upon unfolding a subnetwork $\mathcal{W}(i)$ to $\mathcal{W}(i+1)$, just only the leak construct of $\mathcal{W}(i)$ is modified. The probability tables of all cause variables C_j, $j = 1, \ldots, i$, and of all associated effect variables E_j remain unaltered. The marginal distributions $\Pr(E_1, C_1, \ldots, C_i)$ and $\Pr'(E_1, C_1, \ldots, C_i)$ from the subnetworks $\mathcal{W}(i)$ and $\mathcal{W}(i+1)$ respectively, can therefore differ only if $\Pr(E_U)$ in $\mathcal{W}(i)$ differs from $\Pr'(E_{i+1})$ in $\mathcal{W}(i+1)$. By the argument above it is readily seen that $\Pr(E_U) = \Pr'(E_{i+1})$ and, hence, that the invariant holds. More in general in fact, the marginal distribution over the variables $E_1, \ldots, E_i, C_1, \ldots, C_i$ coincides for any subnetwork $\mathcal{W}(k)$ with $k \geq i$.

5. Practical Properties of Recursive Unfolding

In the preceding sections we introduced and detailed our technique of recursive unfolding for leaky noisy-or models. As mentioned in our introduction, enhancing the flexibility of causal interaction models for engineering purposes was the main motivation for its development. We now return to this motivation and briefly discuss properties of our unfolding scheme that allow for such flexibility.

Order of Unfolding. With leaky noisy-or models in general, the meaning of the represented information is independent of the order in which the cause variables are specified. Our technique of recursive unfolding equally respects this property, in the sense that the result of recursive application of the unfold transform does not depend on the order in which the cause variables are extracted from the leak variable. To support this claim, we consider a subnetwork $\mathcal{W}(k)_{ij}$ capturing a leaky noisy-or model with the cause variables $C_1, \ldots, C_i, \ldots, C_j, \ldots, C_k$, $1 < i < j \leq k$, ordered according to their order of extraction, and with the remaining leak parameter p_U. We compare this representation to the subnetwork $\mathcal{W}(k)_{ji}$ in which the cause variables C_i and C_j have swapped positions in the extraction order. The property $\Pr_{ij}(e_1) = \Pr_{ji}(e_1)$, with the probability distributions defined by $\mathcal{W}(k)_{ij}$ and $\mathcal{W}(k)_{ji}$ respectively, now readily follows from the following considerations:

- The conditional probability tables for the auxiliary effect variables E_i and E_j differ in just their noisy-effect parameters. These parameters p_m, $m \in \{i, j\}$, moreover, are determined *solely* by the cause parent C_m of the variable E_m and not by its effect parent E_{m+1}.
- The remaining leak parameter p_U in the probability table of the leak's effect variable E_U is the same in both subnetworks, as a result of commutativity of the leak updating rule.

A subnetwork representing a leaky noisy-or model with the cause variables $C_1, \ldots, C_k$ thus includes the exact same conditional probability tables, albeit in a different order, regardless of the extraction order used. We note that, as a consequence, also the conditional probabilities $\Pr(e_1 \mid \mathbf{c})$ for the main effect variable E_1, given cause combinations $\mathbf{c}$, will be the same across the possible subnetworks. The semantics of the intermediate auxiliary-effect variables as formulated by Renooij and van der Gaag (2019) however, will differ among representations.

Undoing Unfolds. From an engineering perspective, it is desirable that earlier extraction of a cause variable for a leaky noisy-or model, can be *undone* without changing the underlying represented information. Our technique of recursive unfolding provides for this purpose, as it allows folding a previously extracted cause variable back into the auxiliary leak construct. To support this claim, we consider a subnetwork $\mathcal{W}(k)$ with the remaining leak variable C_U and associated leak parameter p_U, and address undoing the extraction of a cause variable C_i for some $i \leq k$. Undoing the earlier extraction now amounts to the following two steps:

- To include the previously extracted cause variable C_i back into the leak construct, the definition of the leak variable C_U is adapted from $C_U \equiv C_{k+1} \vee \ldots \vee C_n$ to $C'_U \equiv C_i \vee C_{k+1} \vee \ldots \vee C_n$ and the prior probability of c'_U is set to $\Pr(c'_U) = 1$.
- To include the noisy-effect parameter p_i of the cause C_i back into the leak construct, the leak parameter p_U in the probability table for the effect variable E_U is adapted to p'_U by means of the *inverse* of the leak updating rule, that is, to $p'_U = 1 - (1 - p_U) \cdot (1 - p_i \cdot \Pr(c_i))$.

Based on these steps, a *fold-back transform* is readily defined.

Merging Unfolded Causes. From an engineering perspective, it is also desirable that earlier extracted cause variables in a leaky noisy-or model can be *merged* into a single binary compound variable without changing the underlying represented information. Our technique of recursive unfolding readily provides for this purpose as it allows constructing the merged cause variable and its associated parameters from those of the separate variables being merged. To support this claim, we consider the two cause variables C_i and C_j that were previously extracted from the leak construct and now are to be merged. By exploiting the properties stated above, the variables C_i and C_j can each be folded back into the leak construct, after which the new compound cause variable C_{ij} is extracted. For the extraction, the prior probability $\Pr(c_{ij})$ and the noisy-effect parameter p_{ij} are required. Assuming mutual independence of the original cause variables C_i and C_j, these probabilities are readily established from the information available for the two separate cause variables and, hence, do not require any further acquisition efforts from the network engineer:

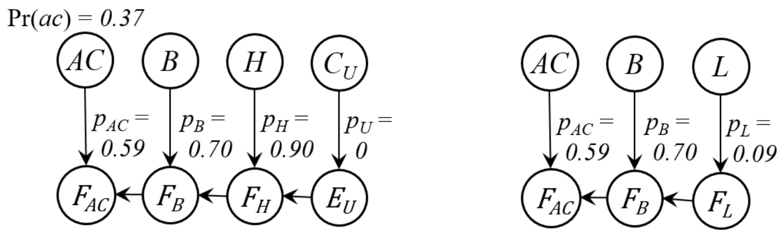


Figure 6: The results of application of the merge transform (*left*) and subsequently the fold back transform (*right*) to our example (leaky) noisy-or subnetwork.

- The compound cause variable C_{ij} has $c_{ij} = c_i \vee c_j$ for its semantics, in line with the semantics of the compound leak variable discussed in Section 4.2.1. The prior probability of the cause c_{ij} being present is found to be $\Pr(c_{ij}) = 1 - (1 - \Pr(c_i)) \cdot (1 - \Pr(c_j))$.

- Since the compound variable C_{ij} represents both cause variables C_i and C_j, the remaining leak after extracting C_{ij} should be the same regardless of whether we extracted the two cause variables separately or in merged form. We thus have that the property $p_{ij} \cdot \Pr(c_{ij}) = 1 - (1 - p_i \cdot \Pr(c_i)) \cdot (1 - p_j \cdot \Pr(c_j))$ must hold, from which the noisy-effect parameter p_{ij} for the compound cause variable C_{ij} is readily derived.

Based on the above considerations, a *merge transform* is readily defined that directly constructs the merged cause variable with the appropriate associated probabilities without the need to first undo the unfolds of the separate cause variables.

We conclude this section with an example to demonstrate the flexibility offered by our approach.

Example 2 *We reconsider our fictitious medical example from Section 4.2.2. At hindsight we decide that, from a domain's point of view, the two cause variables representing different virus infections had better be merged. We apply to this end the merge transform to the variables A and C; we substitute the merged compound variable AC for the original cause variable A (chosen arbitrarily) and replace the latter's effect variable by F_{AC}; the result is shown in Figure 6 (left). The prior probability of the compound cause ac is established from the prior probabilities of the original causes a and c to be $\Pr(ac) = 1 - 0.90 \cdot 0.70 = 0.37$. The three prior probabilities are further combined with the original noisy-effect parameters to give the parameter $p_{AC} \approx 0.59$. We note that, since effectively no causes are removed, the leak parameter p_U requires no updating. We subsequently decide that a heat stroke need not be captured as an explicit cause in the subnetwork representation after all. We therefore apply the fold-back transform to the cause variable H; the result is shown in Figure 6 (right). The leak parameter p_U is increased from zero to $0.9 \cdot 0.01 = 0.09$ as a result of the folding back. We note that the resulting subnetwork now describes a leaky noisy-or model rather than a noisy-or model, and that the leak construct is part of the final subnetwork.* □

6. Conclusions

While causal interaction models have become quite popular as a means to simplify the construction of conditional probability tables for Bayesian networks, network engineers are offered little support with the modelling decisions they typically have to take upon their application. In this paper, we have introduced the novel technique of recursive unfolding to support inserting, removing and merging cause variables in an interaction model, without affecting the underlying information; use of the technique was illustrated by means of a small fictitious example. We would like to note that, although we have detailed our technique of recursive unfolding for cascading subnetwork representations, it is also applicable to interaction models in standard representation, provided that the underlying deterministic function is self-decomposable; this observation is substantiated by the equivalence of the two representations (Renooij and van der Gaag, 2019). Furthermore, while we have used the leaky noisy-or model for our causal interaction model of study, the engineering properties of recursive unfolding are transferable to any interaction model involving binary-valued variables and an underlying self-decomposable deterministic function. For our further research we aim at extending our results to causal interaction models involving multi-valued variables, such as the noisy-MAX model, and involving other types of decomposable function.

Acknowledgment. We thank Atze Dijkstra for raising the issue of maintenance of noisy-or models.

References

J. del Sagrado and A. Salmeron. Representing canonical models as probability trees. In R. Conejo, M. Urretavizcaya, and J.-L. Pérez-de-la-Cruz, editors, *Current Topics in Artificial Intelligence, Lecture Notes in Computer Science, vol. 3040*, pages 478–487. Springer, Berlin, 2003.

F. Díez and M. Druzdzel. *Canonical Probabilistic Models for Knowledge Engineering*. Technical Report CISIAD-06-01, 2007.

D. Heckerman and J. Breese. Causal independence for probability assessment and inference using Bayesian networks. *IEEE Transactions on Systems, Man, and Cybernetics*, 26:826–831, 1996.

M. Henrion. Some practical issues in constructing belief networks. In L. Kanal, T. Levitt, and J. Lemmer, editors, *Proceedings of the Third Annual Conference on Uncertainty in Artificial Intelligence*, pages 161–173. Elsevier Science, North-Holland, 1989.

P. Jesus, C. Baquero, and P. Almeida. A survey of distributed data aggregation algorithms. *IEEE Communications Surveys & Tutorials*, 17:381–404, 2011.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques — Adaptive Computation and Machine Learning*. The MIT Press, 2009.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems. Networks of Plausible Inference*. Morgan Kaufmann, 1988.

S. Renooij and L. C. van der Gaag. The hidden elegance of causal interaction models. In N. Ben Amor, B. Quost, and M. Theobald, editors, *Proceedings of the 13th International Conference on Scalable Uncertainty Management, Lecture Notes in Artificial Intelligence, vol. 11940*, pages 38–51. Springer, Switzerland, 2019.

A. T. Zagorecki. *Local Probability Distributions in Bayesian networks: Knowledge Elicitation and Inference*. PhD thesis, University of Pittsburgh, 2010.