Available online at www.sciencedirect.com

ScienceDirect

Cognitive Systems Research 14 (2012) 50–59

Cognitive Systems RESEARCH

www.elsevier.com/locate/cogsys

Finding MAPs using strongly equivalent high order recurrent symmetric connectionist networks

Action editors: Minho Lee and Wlodzislaw Duch

Emad A.M. Andrews [*], Anthony J. Bonner

Department of Computer Science, University of Toronto, 40 St. George Street, Room 4252, Toronto, Ontario, Canada M5S 2E4

Received 1 April 2010; accepted 12 December 2010
Available online 21 January 2011

Abstract

Belief revision is the problem of finding the most plausible explanation for an observed set of evidences. It has many applications in various scientific domains like natural language understanding, medical diagnosis and computational biology. Bayesian Networks (BN) is an important probabilistic graphical formalism widely used for belief revision tasks. In BN, belief revision can be achieved by finding *the maximum a posteriori* (MAP) assignment. Finding MAP is an NP-Hard problem. In previous work, we showed how to find the MAP assignment in BN using High Order Recurrent Neural Networks (HORN) through an intermediate representation of Cost-Based Abduction. This method eliminates the need to explicitly construct the energy function in two steps, objective and constraints. This paper builds on that previous work by providing the theoretical foundation and proving that the resultant HORN used to find MAP is strongly equivalent to the original BN it tries to solve.

Keywords: Belief revision; High order recurrent neural networks; Cost-Based Abduction; Uncertainty; Bayesian Network; Explanation

1. Introduction

1.1. Belief revision and higher cognitive processes

Belief revision is the problem of finding the most plausible explanation for an observed set of evidences. This involves many higher cognitive processes; such as, searching (the examination of alternative hypotheses), suggestion and decision making. In fact, looking for the best explanation for observations is an essential everyday activity for every human being. An obvious example is a mechanic or a physician who considers all symptoms to reach the best possible diagnosis.

In Artificial Intelligence, scientists have been trying to design computational models that resemble the previously mentioned higher cognitive processes to reach the best explanation. The most famous categories of those computational models are: automated reasoning and inference in knowledge bases, artificial neural networks and graphical probabilistic models.

1.2. Probabilistic models and Bayesian Networks

While each computational model or class of models has its own advocates, limitations and capabilities, all of them fall under the broader domain of reasoning under uncertainty, where available information is not complete or contradicting; thus, probabilistic handling seems the best candidate for those tasks. However, probabilistic reasoning was described as being "*epistemologically inadequate*" by McCarthy and Hayes in their basic paper in 1969 (McCarthy & Hayes, 1969). They showed that the number of parameters needed to compute the joint probability distribution is exponentially proportional to the size of the given dataset, which yields the whole mathematical computations intractable. As a result, researchers avoided using probabilistic reasoning until the notion of independence assumption appeared.

* Corresponding author. Tel.: +1 416 570 0845.
E-mail address: emad@cs.toronto.edu (E.A.M. Andrews).

1389-0417/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.cogsys.2010.12.013

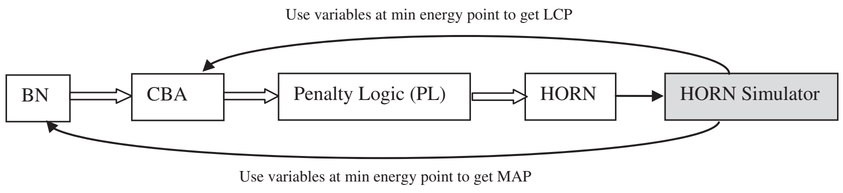
Use variables at min energy point to get LCP
BN
CBA
Penalty Logic (PL)
HORN
HORN Simulator
Use variables at min energy point to get MAP

Fig. 1. An illustration for the framework we are proposing. First we convert the BN into an equivalent CBA system; then, we convert this CBA system into a strongly equivalent HORN. Through a HORN simulator, we can optimize the HORN energy function. HORN variables values at the minimum energy point will achieve both CBA LCP and BN MAP.

In 1988, Pearl standardized the independence assumption notion and formally presented the Bayesian Network (BN) where each variable is conditionally independent of its ancestors given its parents (Pearl, 1988). BN is fully specified by two components: a Directed Acyclic Graph (DAG), whose vertices represent random variables, and a set of parameters that describe the conditional probability distribution of each variable given its parents. Together, these two components fully specify a unique joint distribution over all random variables in the graph. Let G be a DAG, and let $X_1, \ldots, X_n$ denote the set of random variables, vertices of G. The BN encodes the *Markov assumption*: Each variable is independent of all its non-descendant variables given its parents. Thus, the full joint distribution can be composed of the product form:

$$P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | \pi(X_i)) \tag{1}$$

$\pi(X_i)$ is the set of parents of X_i in G. For a specific assignment A over all nodes, Eq. (1) can be rewritten as:

$$P(A) = \prod_{i=1}^{n} P(A(X_i) | A(\pi(X_i))) \tag{2}$$

BN saves a considerable amount of memory and calculations which enables us to calculate joint distributions otherwise impossible to calculate. For example, to specify the full joint distribution for 10 binary random variables we need $2^{10} = 1024$ values to be stored and used during computation. If we use a BN with each variable depending on no more than three other variables, we end up having $10 \times 2^3 = 80$ parameters only.

Given a BN with an observed set of evidence nodes ε, we are looking for values assignment A for the rest of the network nodes, such that $P(A|\varepsilon)$ is maximized, using Bayes rule:

$$P(A|\varepsilon) = \frac{P(A)P(\varepsilon|A)}{P(\varepsilon)} \tag{3}$$

Because we have observed the values of evidence nodes ε, $P(\varepsilon)$ is constant, so it ends up maximizing $P(A)$ that represents the joint probability distribution in (1) and (2). This assignment is called *the maximum a posteriori* assignment (MAP). Once this assignment is found, we can perform all kinds of probabilistic inference needed.

Finding MAP is shown to be NP-hard (Shimony, 1994). For multiply-connected BN[1], existing algorithms suffer from exponential complexity, so new heuristics and algorithms are always needed. In this work we show how we can find MAP using strongly equivalent High Order Recurrent Network (HORN) through an intermediate representation of Cost-Based Abduction (CBA); Fig. 1 illustrates the framework we are proposing. The paper is organized as follows: The rest of the introduction lays the necessary foundation needed to understand our methodology. We will briefly review the Symmetric Connectionist Network (SCN), HORN and CBA. Then, we will trace and analyze the algorithm mentioned in Abdelbar (1998) to show how we can transform a given BN instance into an equivalent CBA system. Following that, we will illustrate the relation between SCN energy minimization and propositional logic satisfiability. We will show how we can transform CBA to HORN to obtain both CBA least cost proof (LCP) and BN MAP. HORN has been used before to solve CBA (Abdelbar, Andrews, & Wunsch, 2003, 2005). This manuscript contributes to the CBA to HORN conversions in three folds: First, we will extend the propositional logic satisfiability using HORN to logical rules with more than three variables. Second, we will show how we can solve the positive feedback problem; finally, we will prove that the resultant HORN is strongly equivalent to the system it solves. The significance and applications of each contribution will be emphasized in its corresponding section. To be able to efficiently present our mathematical proofs in Section 4, Section 3 will shed some light on the relation between logic satisfiability and SCN energy minimization in general.

1.3. Symmetric connectionist network overview

A symmetric connectionist network (SCN) is characterized by a weighted undirected graph whose nodes represent units and arcs represent weighted connections; there are two kinds of arcs:

- *Pair-wise arcs* that link two nodes. Those arcs represent a weighted connection W_{ij} between two nodes i and j.

[1] A multiply-connected Bayesian Network is defined as a network in which there is more than one directed path connecting at least one pair of nodes.

- *Monadic arcs*, each of which is attached to a single node. It represents a bias[2] (a threshold with reversed sign) (Pinkas, 1995).

The network is symmetric which means that $W_{ij} = W_{ji}$. The network is fully specified by an energy function which each unit is trying to minimize. SCNs possess the following advantages:

1. SCNs can be characterized by energy functions that make it easier to specify and follow their behavior.
2. SCNs are used to express and approximate hard problems.
3. SCNs are capable of representing a large set of symmetric networks; therefore, they are quite powerful, so restricting ourselves to SCNs will not make us lose any power.

1.4. High Order Recurrent Networks (HORN)

The Hopfield network is perhaps the best known implementation of SCNs. Hopfield network is a recurrent network (Hopfield, 1982). A Neural Network (NN) is recurrent when its underlying inter-neural connections contain at least one cycle. The underlying topology is a graph and each weighted connection is either a binary connection, T_{ij}, that connects two neurons (i,j) or a unary connection I_i which is the bias of a single neuron i. Each neuron is trying to minimize the energy function which is usually composed of two energy functions:

$$E = E^{Obj} + \beta E^{Const} \tag{4}$$

E^{Obj} describes the objective function to be either maximized or minimized while the E^{Const} ensures the feasibility of the optimized solution by enforcing a set of the constraints. β is a problem dependant free parameter to be experimentally tuned. We can think of β as a tradeoff knob between solution optimality and solution feasibility. Depending on the NN order, E can be either quadratic or higher order.

HORN is a recurrent network whose underlying topology is a hypergraph, allowing weighted hyperedges that connect more than two neurons. The degree of the edge is the number of neurons it connects. The order of the network is the highest degree in the topology. In a Kth-order HORN, a neuron with an activation level u_i and an output V_i is governed by:

$$\frac{du_i}{dt} = \sum_{d=1}^{k} \sum_{s \in S_d, i \in s} T_s^{(d)} \prod_{j \in s, j \neq i} V_j \tag{5}$$

where $V_i = g(u_i)$ and g is typically a sigmoidal activation function. k is the order of the network. $T_s^{(d)}$ is the weight of the d^{th}-degree edge connecting neurons $i_i, \ldots, i_d$. S_d denotes the set of all neuron sequences $J_i, \ldots, J_d$, such that $1 \leqslant J_{1,\ldots,d} \leqslant n$; where n is the number of neurons and $J_a \neq J_b$ if $a \neq b$ (Abdelbar, 1999). Each unit minimizes the following Kth-order energy function:

$$\begin{aligned} K = &- \sum_{1 \leqslant i_1 < i_2 < \cdots < i_k \leqslant n} T^{(k)}_{i_1 \cdots i_k} V_{i_1} \cdots V_{i_k} \\ &- \sum_{1 \leqslant i_1 < i_2 < \cdots < i_{k-1} \leqslant n} T^{(k-1)}_{i_1 \cdots i_{k-1}} V_{i_1} \cdots V_{i_{k-1}} - \cdots \\ &- \sum_{1 \leqslant i \leqslant n} T^{(1)} V_i \end{aligned} \tag{6}$$

1.5. Cost-Based Abduction (CBA)

CBA was first introduced by Charniak and Shimony (1990). Formally, a CBA system is a 4-tuple (H, R, c, G), where H is a set of hypotheses or propositions, c is a function from H to a nonnegative real $c(h)$ called the assumability cost of $h \in H$, R is a set of rules of the form: $R : (p_{i_1} \wedge p_{i_2} \cdots \wedge p_{i_n}) \rightarrow p_{i_k}$ for all $p_{i_1}, \ldots, p_{i_n} \in H$, $p_{i_k} \in H$ and $G \subseteq H$ is the goal or the evidence set (Abdelbar, 1998).

Our objective is finding the Least Cost Proof (LCP) for the goal. The cost of a proof is the sum of all costs of the hypotheses needed to be assumed to complete the proof. Any given hypothesis $p_i \in H$ can be true either by proving it or by assuming it to be true and paying its assumability cost. Hypotheses that can be assumed have assumability costs less than ∞, we call them "*assumables*". Consequent hypotheses that are proven through the assumables are called "*provables*".

Finding the optimal solution for CBA is proven to be NP-hard (Charniak & Shimony, 1994; Shimony, 1994). Previous approaches to CBA can be found in Den (1994), Ishizuka and Matsuo (2002), and Santos (1994). The only NN approach to CBA was introduced in Abdelbar et al. (2003), where the authors found the optimal solution of CBA by transforming it into HORN through an intermediate representation of Penalty Logic (PL).

Finding the LCP in CBA is equivalent to finding the MAP in BN (Charniak & Shimony, 1994; Santos, 1994). Despite their equivalency, it is believed that finding LCP is more efficient than finding MAP and it may be easier to find a heuristic for CBA than finding one for BN (Abdelbar, 1998; Charniak & Shimony, 1994). Santos found the necessary and sufficient conditions under which CBA is polynomially solvable (Santos & Santos, 1996). On the other hand, polynomial solvability for finding MAP in BN is not achievable even with applying restrictions on the graphical representation (Shimony, 1994) and even for trying to find an alternative next-best explanation (Abdelbar & Hedetniemi, 1998).

1.6. Relation to previous work

In Andrews and Bonner (2009), we showed how to find MAP for BN using HORN through an intermediate

[2] A bias is the neuron self tendency to fire.

representation of CBA; in this manuscript, we build on that method. We first transform BN into CBA system using the algorithm mentioned in Abdelbar (1998). To our knowledge, this is the only algorithm in the literature that achieves such a transformation, so it is crucial to analyze and discuss it step by step. Then, we will fill in the gap between BN and HORN by solving the resultant CBA system using HORN through the method mentioned in Abdelbar et al. (2003). Besides analyzing the only algorithm that transforms BN to CBA, this work focuses on providing the mathematical foundation which shows the strong equivalence between the BN we are trying to solve and the HORN that we ended up with as a final transformation product. Also, we will provide the mathematical framework to build energy functions equivalent to logical rules that are composed of arbitrary number of hypotheses and show how to solve the positive feedback problem that might arise as a consequence of that.

1.7. Literature review

To our knowledge, the only attempt to find MAP using HORN is in Abdelbar (1999), Abdelbar and Assaggaf (1999). However, this method requires deriving the energy function in two steps: E^{Obj} and E^{Const}; then, the two functions need to be combined into one function as in Eq. (4). That method requires extensive experimentation to set the network parameter β among other parameters (Abdelbar, 1999; Abdelbar & Assaggaf, 1999). Here, we will create the strongly equivalent energy function in one step from the CBA system equivalent to the given BN. That eliminates the need for creating and combining two energy functions and the need for tuning the free parameter β.

2. Transforming BN into CBA

In this section we will follow and analyze the linear time transformation algorithm mentioned in Abdelbar (1998) to transform a given BN into an equivalent CBA system. We will use the multiply-connected BN in Fig. 2 as an example. This example can be found in Murphy's BN tutorial at Murphy (1998).

Our objective is to explain why the grass is wet. So we want to reach an assignment A for the random variables such that $P(A|W)$ is maximized. Using Bayesian inference we can see that:

$$P(S=1|W=1) = \frac{\sum_{c,r} P(C=c, S=1, R=r, W=1)}{P(W=1)} = \frac{0.2781}{0.6471} = 0.430$$

$$P(R=1|W=1) = \frac{\sum_{c,s} P(C=c, S=s, R=1, W=1)}{P(W=1)} = \frac{0.4581}{0.6471} = 0.708$$

So, the best explanation for the wet grass is because it is raining rather than because of having the sprinkler on.

Now we will transform this multiply-connected BN to a CBA system and see whether we can reach the same explanation. The transformation to CBA goes as follows:

1. Transform the CPT of each random variable v to a linear table P_v, Tables 1–4. Each line $l \in P_v$ is a hypothesis that the premises of that line are satisfied. $V(l)$ denotes the probability corresponding to line l in the table. The cost of the hypothesis h_l that represents each line is:

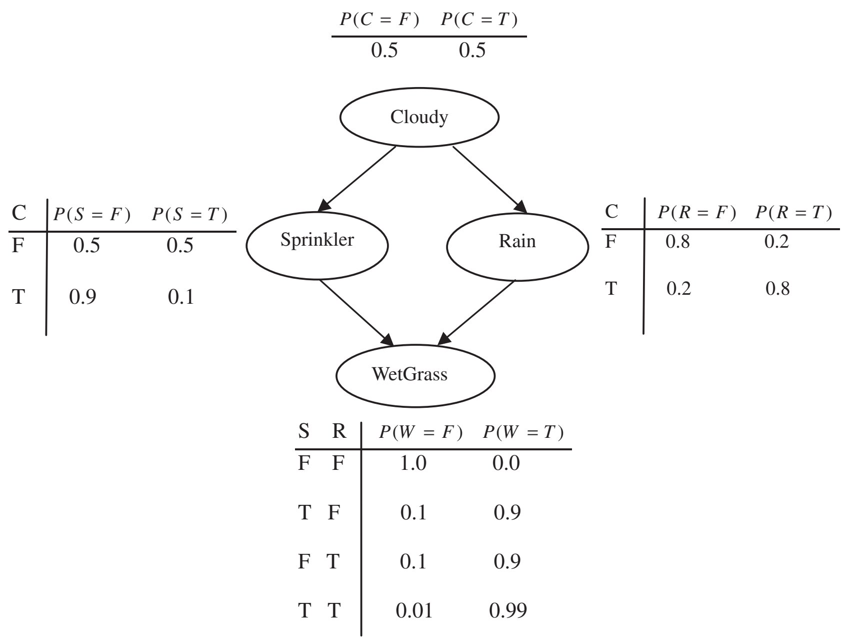

Fig. 2. This is an example of a multiply-connected BN, the original example graph can be found at Murphy (1998). The graph is redrawn for better quality.

$$c(h_l) = -\log V(l) + Q \quad \text{where :}$$
$$Q = -\log \prod_{v \in V} Q_v \quad \text{and} \quad Q_v = \min\{V(l) | l \in P_v\}$$

So:

$$Q_C = 0.5, Q_S = 0.1, Q_R = 0.2, Q_W = 0.9$$

when calculating Q, we ignore the line $P(W|S,R) = 0$ because W is our goal and we are trying to explain why the grass is wet, so $Q = 2.0458$

2. For every variable $v \in V$, create h_v representing that proposition v is assigned some truth value; $c(h_v) = \infty$. The result is a set of hypothesis $\{h_C, h_S, h_R, h_W\}$
3. For every $v \in V$ and for every $t \in D(v)$, where $D(v)$ is domain of v,
   - Construct a hypothesis h_{v_t} denoting that proposition v is assigned a value t. Add h_{v_t} to the system hypothesis and assign $c(h_{v_t}) = \infty$;

   The result is a set of new hypothesis: $\{h_{C_t}, h_{C_f}, h_{S_t}, h_{S_f}, h_{R_t}, h_{R_f}, h_{W_t}\}$, we ignore h_{W_f}.
   - Construct a rule R_{v_t} with $R^A_{v_t} = \{h_{v_t}\}$ and $R^C_{v_t} = \{h_v\}$

Table 1
Cloudy , C.

Value	$P(C)$	h_l	$c(h_l)$
0	0.5	$h_C l_1$	2.3468
1	0.5	$h_C l_2$	2.3468

Table 2
Sprinkler, S.

Value		$P(S\|C)$	h_l	$c(h_l)$
C	S			
0	0	0.5	$h_S l_1$	2.3468
0	1	0.5	$h_S l_2$	2.3468
1	0	0.9	$h_S l_3$	2.0915
1	1	0.1	$h_S l_4$	3.0458

Table 3
Rain , R.

Value		$P(R\|C)$	h_l	$c(h_l)$
C	R			
0	0	0.8	$h_R l_1$	2.1427
0	1	0.2	$h_R l_2$	2.7447
1	0	0.2	$h_R l_3$	2.7447
1	1	0.8	$h_R l_4$	2.1427

Table 4
Wet grass, W. (The goal).

Value			$P(W\|S, R)$	h_l	$c(h_l)$
S	R	W			
0	0	1	0.0	$h_W l_1$	∞
1	0	1	0.9	$h_W l_2$	2.0915
0	1	1	0.9	$h_W l_3$	2.0915
1	1	1	0.99	$h_W l_4$	2.0501

Where R^A refers to the set of R's antecedents and R^C refers to R's consequent. The result is this set of rules:

$$R_{C_t} : h_{C_t} \to h_C, \quad R_{C_f} : h_{C_f} \to h_C$$
$$R_{S_t} : h_{S_t} \to h_S, \quad R_{S_f} : h_{S_f} \to h_S$$
$$R_{R_t} : h_{R_t} \to h_R, \quad R_{R_f} : h_{R_f} \to h_R$$
$$R_{W_t} : h_{W_t} \to h_W, \quad R_{W_f} : h_{W_f} \to h_W$$

4. For every $v \in V$ and every $l \in P_v$:
   - Construct a rule R_l where: $R^A_l = \{h_l\}$
   - For every $\{u \to t'\} \subseteq l$, where $u \in \pi(v)$ and $t' \in D(u)$, set $R^A_l = R^A_l \cup \{h_{u_{t'}}\}$
   - Let $t \in D(v)$ be the value from v's domain that satisfies $\{v \to t\} \subseteq l$, set $R^C_l = \{h_{v_t}\}$. The result is the following sets of rules:

$$R_C l_1 : h_C l_1 \to h_{C_f}$$
$$R_C l_2 : h_C l_2 \to h_{C_t}$$
$$R_R l_1 : h_R l_1 \wedge h_{C_f} \to h_{R_f}$$
$$R_R l_2 : h_R l_2 \wedge h_{C_f} \to h_{R_t}$$
$$R_R l_3 : h_R l_3 \wedge h_{C_t} \to h_{R_f}$$
$$R_R l_4 : h_R l_4 \wedge h_{C_t} \to h_{R_t}$$
$$R_S l_1 : h_S l_1 \wedge h_{C_f} \to h_{S_f}$$
$$R_S l_2 : h_S l_2 \wedge h_{C_f} \to h_{S_t}$$
$$R_S l_3 : h_S l_3 \wedge h_{C_t} \to h_{S_f}$$
$$R_S l_4 : h_S l_4 \wedge h_{C_t} \to h_{S_t}$$
$$R_W l_1 : h_W l_1 \wedge h_{S_f} \wedge h_{R_f} \to h_{W_t}$$
$$R_W l_2 : h_W l_2 \wedge h_{S_t} \wedge h_{R_f} \to h_{W_t}$$
$$R_W l_3 : h_W l_3 \wedge h_{S_f} \wedge h_{R_t} \to h_{W_t}$$
$$R_W l_4 : h_W l_4 \wedge h_{S_t} \wedge h_{R_t} \to h_{W_t}$$

Finally, the goal set $G = \{h_C, h_W, h_S, h_R, h_{W_t}\}$ and $R_G : h_C \wedge h_W \wedge h_S \wedge h_R \wedge h_{W_t} \to G$.

As discussed above, finding the LCP for this derived CBA system is the same as finding MAP for the BN in Fig. 2. In other words, the values assigned to CBA variables to reach the LCP are the same values that achieve MAP for the equivalent BN. Section 4 will illustrate how HORN can be used to find LCP for the CBA which will be the same as finding MAP for BN.

3. SCNs energy minimization and propositional logic satisfiability

The previous section concluded the first step which is converting BN into an equivalent CBA system in linear time. However, what does that have to do with SCN? In other words, is there a relation between logic satisfiability and SCN energy minimization?

In Pinkas (1990, 1992), Pinkas showed that propositional logic can be represented efficiently by energy minimization connectionist networks and thus may give us a fast parallel implementation for a propositional inference engine. He provided a constructive algorithm for transforming the standard propositional logic clauses to equivalent SCN energy functions and vice-versa. He showed a two-way equivalency between both problems, such that (Pinkas, 1995):

- For every propositional sentence there is a quadratic energy function; such that, values of these energy function variables at the minimum energy point satisfy the propositional sentence.
- Any quadratic energy minimization problem can be described as a sustainable propositional sentence for the same assignments that minimize the function.

He defined a characteristic function that maps every propositional sentence σ into a characteristic algebraic form H_σ as follows:

$$H_\sigma = \begin{cases} x_i & \text{if } \sigma = x_i \text{ is an atomic proposition} \\ 1 - H_{\sigma'} & \text{if } \sigma = \neg\sigma' \\ H_{\sigma_1} \times H_{\sigma_2} & \text{if } \sigma = \sigma_1 \wedge \sigma_2 \\ H_{\sigma_1} + H_{\sigma_2} - H_{\sigma_1} \times H_{\sigma_2} & \text{if } \sigma = \sigma_1 \vee \sigma_2 \end{cases} \tag{7}$$

Using this function, the resultant algebraic form H_σ has its maximal points at the truth assignments that satisfy the clause. H_σ has values of zero or one which map to logical values of false or true, respectively. For example, the expression $a \wedge b \rightarrow c$ will be converted to its equivalent characteristic function as follows:

$$\begin{aligned} H(a \wedge b \rightarrow c) &= H(\neg(a \wedge b) \vee c) = H(\neg a \vee \neg b \vee c) \\ &= H(\neg a \vee \neg b) + c - H(\neg a \vee \neg b)c \\ &= (1-a) + (1-b) - (1-a)(1-b) + c \\ &\quad - c(1-a) - c(1-b) + c(1-a)(1-b) \\ &= 1 - ab + abc \end{aligned} \tag{8}$$

Table 5 shows that this algebraic form has its maximal points at the truth assignments that satisfy the clause.

The equivalent energy function for a given proposition sentence σ is the characteristic function of the sentence negation $H_{-\sigma}$.

Table 5
Truth table for maximal points.

a	b	c	s	$H(s)$
0	0	0	T	1
0	0	1	T	1
0	1	0	T	1
0	1	1	T	1
1	0	0	T	1
1	0	1	T	1
1	1	0	F	0
1	1	1	T	1

3.1. Penalty logic

Pinkas's idea illustrated in the previous section serves satisfiability well, but it does not provide a useful tool for reasoning under uncertainty because it is based on the first-order propositional logic (propositional calculus). In 1995, Pinkas presented an extension to the propositional calculus to be able to express the strength of a given belief. This extension resulted in the Penalty Logic (PL). Any PL is mapped to an equivalent SCN through the same characteristic function in Eq. (7). In PL, each rule is associated with a numeric value, *a penalty*, which represents the strength of the belief or the reliability of the rule. PL is defined in Pinkas (1995) as:

"*A Penalty logic well formed formula (PLWFF)* ψ *is a finite set of pairs. Each pair is composed of a real positive number, called penalty, and a standard propositional formula called assumption (or belief); i.e.,* $\psi = \{\langle p_i, \varphi_i\rangle | p_i \in \mathbb{R}^+$, φ_i *is* WFF, $i = 1, \ldots, n\}$. *The set of beliefs that are in* ψ, *denoted by* U_ψ, *is* $U_\psi = \{\varphi_i | \langle p_i, \varphi_i\rangle \in \psi\}$"

3.2. Penalty logic violation rank

Each well formed formula (WFF) φ_i is associated with a real-valued penalty p_i which represent the credibility of φ_i. The truth assignment of the atomic propositions in the PLWFF is called a model for this PLWFF. If a model violates φ_i, then this real value p_i is paid as a penalty for this violation. The violation rank of a model is the sum of all WFFs penalties violated through this model.

For example, consider the following PL over the set of propositions $\{p, q, r, s\}$:

$$\psi = \begin{matrix} (100, p \wedge \neg r \rightarrow s) \\ (75, q \wedge \neg r \vee \neg q \wedge r) \\ (500, p) \\ (200, s \leftrightarrow \neg q) \\ (50, \neg r \vee \neg p) \\ (120, q) \end{matrix} \tag{9}$$

The minimum violation rank of this PL is 100 as a result of violating the first assumption through the model $\vec{x} = \{p \leftarrow T, r \leftarrow F, s \leftarrow F, q \leftarrow T\}$. No other model can give us a lower violation rank (Abdelbar et al., 2003). We say that $Vrank_\psi(\vec{x}) = 100$. We can use the characteristic function in Eq. (7) to derive the energy function which represents the SCN equivalent to the any PL. The Energy function which fully specifies the equivalent HORN for an arbitrary PL pairs $\psi = \wedge_i^n \sigma_i$ is defined by the general form.

$$E_\psi = \sum_i^n H_{\neg\sigma} \tag{10}$$

In Pinkas (1992), the author defined the characteristic function of an energy function E_ψ to be:

"*The characteristic function of an energy function $E(\vec{X}, \vec{T})$ with $\vec{X}$ visible variables and $\vec{T}$ hidden variables is the function: $rank_E(\vec{X}) = \min_{\vec{T}}\{E(\vec{X}, \vec{T})\}$.*"

Thus, the rank of a model is the energy level obtained by instantiating the visible units with the truth-values specified by the model and letting the hidden units be free to settle at the minimum.

3.3. Strong vs. weak equivalence between PLs and SCNs

In his dissertation (Pinkas, 1992), Pinkas showed that the equivalence between PLs as an input to the characteristic function and the SCNs as an output of the characteristic function can be in two types:

- *Strong equivalence*: the search space of the violation rank function of the PL, including both local and global minimum points, is equivalent to the search space of the energy function of the resultant SCN.
- *Weak equivalence*: the global minimum points of the SCN energy function and the PL violation rank function are identical, but the local minimum points might differ.

The energy function E will be strongly equivalent to its PL if:

$$rank_E = Vrank_\psi + c \tag{11}$$

Pinkas has shown that the strong equivalence between PL and its SCN will always occur as long as each rule in the PL consists of three different propositions at most. That is why his method always uses Conjunction of Triples Form (CTF) for all his logical rules and consequently he never used HORN and restricted himself to quadratic energy functions. If HORN to be used; then, strong equivalence is not guaranteed and needs an explicit proof. In the next section we will show how we use PL as an intermediate representation to solve CBA and BN using HORN. We will prove that the high order SCN generated by our method is strongly equivalent to the CBA and its PL despite not restricting ourselves to neither CTF nor quadratic energy functions. We are allowing logical rules to be of any length and consequently we are allowing energy functions to be of any order, not necessarily limited to quadratic. It is worth mentioning that this brief introduction to PL and its relation to SCN is presented here to be able to follow our work presented in the next section and it is never meant to be comprehensive. The reader is directed to Pinkas (1990, 1992, 1995) for a fuller explanation.

4. Finding LCP using HORN

This section will show the final step, finding the BN MAP. We will convert the CBA system into a strongly equivalent HORN. First, we will summarize how to solve the CBA system using HORN. Then; we will define what the positive feedback problem is and how to solve it; finally, we will prove that the HORN is strongly equivalent to the system it tries to solve. To solve a CBA system using HORN, the process is summarized as follows, the reader is directed to Abdelbar et al. (2003, 2005) for more details:

1. Without loss of generality, we start by processing the CBA system such that all consequents are provables. Also, we make sure that each provable appears only once as a consequent in the system.
2. Given the preprocessed CBA, we reverse the implication direction of all rules to avoid null antecedent proofs.
3. We transform the CBA into PL pairs.
4. We generate the equivalent energy function for all PL formulas using the characteristic function in Eq. (7).

4.1. Deriving energy functions for logical rules with more than 3 variables

As we mentioned before, we are not converting our logical rules into CTF because that will limit us to use only quadratic energy functions. We allow logical rules with more than three variables to be mapped as higher order energy functions. As a result, we need a generic mathematical form that enables us to convert a logical rule of an arbitrary length into an equivalent SCN with high order energy functions. This derivation is of a great significance because it provides a closed form to generate higher order energy functions. Without this derivation, all applications have to be limited to quadratic energy functions only, which vastly limits the size and complexity of the CBA problems that can be solved using recurrent NN.

The derivation goes as follows:

We start by OR rules; consider $\beta = x_n \rightarrow x_1 \vee x_2 \vee\ x_3 \vee \cdots \vee x_{n-1}$:

$$\begin{aligned}\neg\beta &= x_n \wedge \neg(x_1 \vee x_2 \vee x_3 \vee \cdots \vee x_{n-1})\\ &\equiv x_n \wedge \neg x_1 \wedge \neg x_2 \wedge \neg x_3 \wedge \cdots \wedge x_{n-1}\\ \therefore H_{\neg\beta} &= x_n[(1-x_1)(1-x_2)(1-x_3)\ldots(1-x_{n-1})]\\ \therefore E_\beta &= x_n - x_n\left(\sum_{k=1}^{n-1}(-1)^{k+1}\left(\sum_{1\leqslant i_1<\ldots.<i_k\leqslant n-1} x_{i_1}x_{i_2}\ldots x_{i_k}\right)\right)\end{aligned} \tag{12}$$

For AND rules, consider $\beta = x_n \rightarrow x_1 \wedge x_2 \wedge\ x_3 \wedge \cdots \wedge x_{n-1}$

$$\begin{aligned}\neg\beta &\equiv x_n \wedge \neg(x_1 \wedge x_2 \wedge x_3 \wedge \cdots \wedge x_{n-1})\\ &\equiv x_n(1 - x_1x_2x_3\ldots x_{n-1})\\ \therefore H_{\neg\beta} &= x_n - x_nx_1x_2x_3\ldots x_{n-1}\\ \therefore E_\beta &= x_n - \prod_{i=1}^{n} x_i\end{aligned} \tag{13}$$

4.2. The positive feedback problem

Eqs. (12) and (13) are generic enough to model logical rules of an arbitrary length to an equivalent high order energy function or HORN for short. However, if a certain hypothesis in repeated more than once as an antecedent of the same consequent, we will end up with a positive feedback problem. The positive feedback in SCN means that a certain unit is providing a positive feedback to itself. This violates the original definition of recurrent SCN where each unit can be connected to all other units but itself. For example, consider those two rules:

$$\begin{aligned} &p \wedge r \rightarrow s \\ &p \wedge u \rightarrow s \end{aligned} \tag{14}$$

According to our method, we combine them into one rule to be:

$$s \rightarrow (p \wedge r) \vee (p \wedge u) \tag{15}$$

Using the characteristic function in Eq. (7) will result in the following energy function:

$$E = s - spr - spu + sp^2ru \tag{16}$$

In this energy function we have a quadratic powered variable in the fourth term. The neuron corresponding to the proposition p appears twice in the same edge. The edge will look like that "*sppru*" in the HORN. This is a positive feedback which, by definition, is not allowed in HORN.

One way to solve this problem is distributing the repeated variable, so the logical rule in Eq. (16) will be:

$$s \rightarrow p \wedge (r \vee u) \tag{17}$$

But this logical formulation does not agree with the CBA definition mentioned earlier, as we can have disjunction of conjunctions but not conjunction of disjunctions. As a result, we applied the distributive law on the characteristic function level instead, so we restricted the power of any variable to be at most 1. As a result, Eq. (16) can be re-written as:

$$E = s - spr - spu + spru \tag{18}$$

Claim 4.1. *Applying the distributive law at the characteristic function level, by restricting the power of any variable at any algebraic term to be at most* 1, *will result in the same energy function of applying the distributive law on the same variable at the logic level.*

Proof

- We will start by deriving the general energy function for a conjunction of disjunctions after applying the distributive law at the logical rule level, i.e. before applying the characteristic function:Let β be the rule:

$$s \rightarrow (p \wedge x_1) \vee (p \wedge x_2) \vee \cdots \vee (p \wedge x_n) \tag{19}$$

By applying the distributive law, β can be re-written as

$$\neg s \vee [p \wedge (x_1 \vee x_2 \vee \cdots \vee x_n)] \tag{20}$$

Then $\neg\beta$ *is*:

$$\begin{aligned} &\neg(\neg s \vee [p \wedge (x_1 \vee x_2 \vee \cdots \vee x_n)]) \\ &\quad \equiv s \wedge \neg[p \wedge (x_1 \vee x_2 \vee \cdots \vee x_n)] \\ &\quad \equiv s \wedge [\neg p \vee \neg(x_1 \vee x_2 \vee \cdots \vee x_n)] \\ &\quad \equiv s \wedge [\neg p \vee (\neg x_1 \wedge \neg x_2 \wedge \cdots \wedge \neg x_n)] \end{aligned} \tag{21}$$

Hence we have:

$$\begin{aligned} H_{\neg s} &= s\begin{bmatrix} ((1-p) + [(1-x_1)(1-x_2)\ldots(1-x_n)]) \\ -(1-p)[(1-x_1)(1-x_2)\ldots(1-x_n)] \end{bmatrix} \\ &= s\begin{bmatrix} \left((1-p) + 1 - \sum_{k=1}^{n}(-1)^{k+1}\left(\sum_{1\leqslant i_1<\ldots\ldots<i_k\leqslant n} x_{i_1}x_{i_2}\ldots x_{i_k}\right)\right) \\ -(1-p)\left[1 - \sum_{k=1}^{n}(-1)^{k+1}\left(\sum_{1\leqslant i_1<\ldots\ldots<i_k\leqslant n} x_{i_1}x_{i_2}\ldots x_{i_k}\right)\right] \end{bmatrix} \\ &= s\begin{bmatrix} 1 - p + 1 - \sum_{k=1}^{n}(-1)^{k+1}\left(\sum_{1\leqslant i_1<\ldots\ldots<i_k\leqslant n} x_{i_1}x_{i_2}\ldots x_{i_k}\right) - \\ 1 + p + \sum_{k=1}^{n}(-1)^{k+1}\left(\sum_{1\leqslant i_1<\ldots\ldots<i_k\leqslant n} x_{i_1}x_{i_2}\ldots x_{i_k}\right) - \\ p\sum_{k=1}^{n}(-1)^{k+1}\left(\sum_{1\leqslant i_1<\ldots\ldots<i_k\leqslant n} x_{i_1}x_{i_2}\ldots x_{i_k}\right) \end{bmatrix} \\ &= s\left[1 - p\sum_{k=1}^{n}(-1)^{k+1}\left(\sum_{1\leqslant i_1<\ldots\ldots<i_k\leqslant n} x_{i_1}x_{i_2}\ldots x_{i_k}\right)\right] \end{aligned} \tag{22}$$

Then, the full energy function E will be:

$$E = s - sp\sum_{k=1}^{n}(-1)^{k+1}\left(\sum_{1\leqslant i_1<\ldots\ldots<i_k\leqslant n} x_{i_1}x_{i_2}\ldots x_{i_k}\right) \tag{23}$$

- Now, let us proceed with the same logical rule in Eq. (19) in its original form without applying the distributive law; instead, we will apply it after deriving the energy function:

  Using the same β as in (19):

$$\begin{aligned} &s \rightarrow (p \wedge x_1) \vee (p \wedge x_2) \vee \cdots \vee (p \wedge x_n) \\ &\quad \equiv \neg s \vee [(p \wedge x_1) \vee (p \wedge x_2) \vee \cdots \vee (p \wedge x_n)] \end{aligned} \tag{24}$$

Then $\neg\beta$ *is*:

$$\begin{aligned} &\neg(\neg s \vee [(p \wedge x_1) \vee (p \wedge x_2) \vee \cdots \vee (p \wedge x_n)]) \\ &\quad \equiv s \wedge \neg[(p \wedge x_1) \vee (p \wedge x_2) \vee \cdots \vee (p \wedge x_n)] \\ &\quad \equiv s \wedge [\neg(p \wedge x_1) \wedge \neg(p \wedge x_2) \wedge \cdots \wedge \neg(p \wedge x_n)] \end{aligned} \tag{25}$$

Hence we have:

$$H_{\neg s} = s[(1 - px_1)(1 - px_2)\ldots(1 - px_n)] \tag{26}$$

Thus, the general notation of the energy function E_s will be:

$$E = s - s\left(\sum_{k=1}^{n}(-1)^{k+1}p^k\left(\sum_{1\leqslant i_1<\ldots\ldots<i_k\leqslant n} x_{i_1}x_{i_2}\ldots x_{i_k}\right)\right) \tag{27}$$

By applying the distributive law at the characteristic function level, we restrict the power of the variable p to be at most one; then, E will be:

$$E = s - sp\left(\sum_{k=1}^{n}(-1)^{k+1}\left(\sum_{1\leqslant i_1<\ldots<i_k\leqslant n} x_{i_1}x_{i_2}\ldots x_{i_k}\right)\right) \tag{28}$$

From the two equations in (23) and (28) we can see that we have the same energy function, which proves our claim. As a result, applying the distributive law on the characteristic function level will result in the same energy function of applying it at the logic level.

4.3. Proving the strong equivalency

In this subsection we will show that the HORN generated by our previous method is strongly equivalent to the CBA and to the BN it tries to solve. Showing the strong equivalence between the two problems is very important because that will eliminate any restrictions on any heuristic that might be used because the two search spaces will be identical including both local and global minima. It is even of a special importance to map local minima when we search for an alternative or the next-best explanation.

Claim 4.2. *The HORN generated by the previous method is strongly equivalent to the CBA it tries to find its LCP and hence to the BN it tries to find its MAP.*

Proof. For any truth assignment $(\vec{x})$, the ranking function of the energy function equivalent to the PL ψ is:

$$rank_E = \sum_{i=1}^{n} E_{\varphi_i} = \sum_{\vec{x}\models\varphi_i} E_{\varphi_i} + \sum_{\neg(\vec{x}\models\varphi_i)} E_{\varphi_i}$$

When $(\vec{x})$ satisfies φ_i, then $E_{\varphi_i} = 0$ (No penalty is paid, nothing is added to the energy level)[3] Then, the energy function rank will be:

$$rank_E = \sum_{i=1}^{n} E_{\varphi_i} = 0 + \sum_{\neg(\vec{x}\models\varphi_i)} E_{\varphi_i} = \sum_{\neg(\vec{x}\models\varphi_i)} E_{\varphi_i}$$

On the other hand, when $(\vec{x})$ violates φ_i; then, $E_{\varphi_i} = p_i$, where p_i is the penalty of the assumption φ_i. Then, the energy function rank will be:

$$rank_E = \sum_{\neg(\vec{x}\models\varphi_i)} E_{\varphi_i} = \sum_{\neg(\vec{x}\models\varphi_i)} p_i = Vrank_\psi \tag{29}$$

We can see that Eq. (29) agrees to the strongly equivalence condition in Eq. (11), which means our method will produce a strongly equivalent high order recurrent SCN to the CBA and the BN it tries to solve.

[3] In the original characteristic function, when $(\vec{x})$ satisfies φ_i; then, its characteristic function (H_φ) value is equal to one. In the case of the energy function $(H_{\neg\varphi})$, when $(\vec{x})$ satisfies φ_i, $H_{\neg\varphi} = 0$ to prevent paying the penalty. On the other hand when $(\vec{x})$ violates φ_i, $H_{\neg\varphi} = 1$, to pay the penalty.

5. Solution quality and size complexity

For the example traced in this manuscript, it is clear that the network reached the LCP and assigned values which gave the maximum joint probability for the random variables of the BN in Fig. 2. In general, we judge whether the network reached the global minima by benchmarking the solution against the results obtained by the popular public domain *lp-solve* engine which solves the CBA system after converting it to the equivalent Linear Programming (LP) form.

The example above showed that HORN solved a problem of size 26-hypothesis, 22-rule. Problem size is not the only factor which determines a CBA instance difficulty and its search space complexity. Other factors like solution depth, rules length, and the ratio between the number of rules to the total number of hypotheses are also taken into consideration when considering a CBA instance difficulty level. Our previous work showed that HORN constantly found feasible solutions for a CBA system with 300-hypothesis, 900-rule with particular high difficulty (Abdelbar et al., 2003, Abdelbar, El-Hemely, Andrews, & Wunch, 2005).

6. Results summary

Using the previously mentioned transformations, we constructed the energy function which represents the CBA system derived in Section 2. Then, we used HORN simulator to minimize this energy function. The LCP was found through the following assignments $\{C \to T, R \to T,\ S \to F,\ W \to T\}$. The total cost is 8.6725 by assuming the following hypotheses $h_C l_2$, $h_S l_3$, $h_R\ l_4$ and $h_W l_3$. This is the same solution we reached using Bayesian inference for the BN in Fig. 2. Table 6 summarizes the results of the HORN which solved this example.

We can also find the LCP by backtracking the rules starting from the goal rule. We only need to calculate towards h_{W_t} because all other hypotheses in the goal rule are provables with assumability cost of ∞. By backtracking rules R_{W_t}'s, it is clear that we cannot use $R_W l_1$ as it costs us ∞, because it explains that the grass is wet while there is neither rain nor sprinkler on. That leaves us with rules $R_W l_2$, $R_W l_3$ and $R_W l_4$ with costs: 8.9278, 8.6725 and 9.4884, respectively. That means the best explanation for the observation that the grass is wet is $R_W l_3$ which assumes that the sprinkler is off and there is rain. The LCP assignment of the constructed CBA system is the same assignment for the variables in the BN to achieve MAP.

Table 6
Results summary.

R_G	Network order	Network iterations	Cost
$h_C \wedge h_W \wedge h_R \wedge h_S \wedge h_{W_t} \to G$	9	98489	8.6725
$h_{W_t} \to G$	9	136128	8.6725

7. Concluding remarks and future work

In this paper we showed how to find MAP in BN using HORN through an intermediate representation of CBA. This method creates the full integrated energy function directly without explicitly deriving separate objective and constraint functions. We traced and analyzed the only algorithm in literature that transforms BN to CBA. We derived a general framework to transform logical rules of any length to strongly equivalent energy functions and showed how to solve the positive feedback problem that might arise when using this framework. Future work would be to invent new algorithms that transform BN to CBA while taking care of the size ratio between both systems. Finding MAPs for BN with continuous probability distribution will be an interesting follow up for this work. Also, studying which classes of BN can create polynomially solvable CBA systems can be a very interesting research problem.

References

Abdelbar, A. M. (1998). An algorithm for finding MAPs for belief networks through cost-based abduction. *Artificial Intelligence, 104*(1–2), 331–338.

Abdelbar, A. M. (1999). Designing high order recurrent networks for Bayesian belief revision. In L. R. Medsker & L. C. Jain (Eds.), *Recurrent neural networks: design and applications* (pp. 77–98). Boca Raton: CRC Press.

Abdelbar, A.M. & Assaggaf, M. (1999). Recurrent high-order networks for probabilistic explanation. In Proceedings of the IEEE/INNS international joint conference on neural networks.

Abdelbar, A. M., Andrews, E. A. M., & Wunsch, Donald C. (2003). Abductive reasoning with recurrent neural networks. *Neural Networks, 16*(5–6), 665–673.

Abdelbar, A. M., El-Hemely, M. A., Andrews, E. A. M., & Wunch, D. C. (2005). Recurrent neural networks with backtrack-points and negative reinforcement applied to cost-based abduction. *Neural Networks, 18*(5–6), 755–764.

Abdelbar, A.M., Andrews, E.A.M. & Tagliarini, G.A. (2003). Finding least cost proofs using high order recurrent networks. In Proceedings of the IEEE/INNS international joint conference on neural networks.

Abdelbar, A. M., & Hedetniemi, S. M. (1998). Approximating MAPs for belief networks is NP-hard and other theorems. *Artificial Intelligence, 102*(1), 21–38.

Andrews, E. A., & Bonner, A. J. (2009). Finding MAPs using high order recurrent networks. In C. S. Leung, M. Lee, & J. H. Chan (Eds.), *Proceedings of the 16th international conference on neural information processing: Part I* (pp. 100–109). Berlin/Heidelberg: Springer-Verlag.

Charniak, E., & Shimony, S. E. (1990). Probabilistic semantics for cost-based abduction. In *Proceedings of the AAAI national conference on artificial intelligence*. Brown University.

Charniak, E., & Shimony, S. E. (1994). Cost-based abduction and MAP explanation. *Artificial Intelligence, 66*(2), 345–374.

Den, Y. (1994). Generalized chart algorithm: an efficient procedure for cost-based abduction. In Proceedings of the 32nd annual meeting on association for computational linguistics, association for computational linguistic.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *National Academy of Science, 79*, 2554–2558.

Ishizuka, M., & Matsuo, Y. (2002). SL method for computing a near-optimal solution using linear and non-linear programming in cost-based hypothetical reasoning. *Knowledge-based Systems, 15*(7), 369–376.

McCarthy, J., & Hayes, P. J. (1969). Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer & D. Michie (Eds.). *Machine intelligence* (Vol. 4, pp. 463–502). Edinburgh University Press.

Murphy, K. (1998, 2010). A brief introduction to graphical models and bayesian networks. <http://www.cs.ubc.ca/~murphyk/Bayes/bnintro.html>.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. San Francisco: Morgan Kaufmann.

Pinkas, G. (1990). The equivalence of connectionist energy minimization and propositional calculus satisfiability. Technical report, WUCS-90-03, Department of Computer Science, Washington University.

Pinkas, G. (1992). Logical inference in symmetric connectionist networks. PhD Thesis, Washington University.

Pinkas, G. (1995). Reasoning, nonmonotonicity and learning in connectionist networks that capture prepositional knowledge. *Artificial Intelligence, 77*, 203–237.

Santos, E. J. (1994). A linear constraint satisfaction approach to cost-based abduction. *Artificial Intelligence, 65*(1), 1–27.

Santos, E. J., & Santos, E. S. (1996). Polynomial solvability of cost-based abduction. *Artificial Intelligence, 86*(1), 157–170.

Shimony, S. E. (1994). Finding MAPs for belief networks is NP-hard. *Artificial Intelligence, 68*(2), 399–410.