Local Monotonicity in Probabilistic Networks

Johan Kwisthout*, Hans Bodlaender, and Gerard Tel

Department of Information and Computer Sciences, University of Utrecht,
P.O. Box 80.089, 3508 TB Utrecht, The Netherlands.
{johank, hansb, gerard}@cs.uu.nl

Abstract. It is often desirable that a probabilistic network is monotone, e.g., more severe symptoms increase the likeliness of a more serious disease. Unfortunately, determining whether a network is monotone is highly intractable. Often, approximation algorithms are employed that work on a local scale. For these algorithms, the monotonicity of the arcs (rather than the network as a whole) is determined. However, in many situations monotonicity depends on the ordering of the values of the nodes, which is sometimes rather arbitrary. Thus, it is desirable to order the values of these variables such that as many arcs as possible are monotone. We introduce the concept of local monotonicity, discuss the computational complexity of finding an optimal ordering of the values of the nodes in a network, and sketch a branch-and-bound exact algorithm to find such an optimal solution.

1 Introduction

In many probabilistic networks [9] that are used for classification in real problem domains, the variables of the network can be distinguished into observable input variables, non-observable intermediate variables and a single output variable. For example, in a medical domain the observable variables represent clinical evidence such as observable symptoms and test results, the output variable functions as a classification of a disease, and the intermediate variables model non-observable facts that are relevant for classification. Often, the relations between observable symptoms and the classification variable are monotone, e.g., higher values for the observable variable 'fever' makes higher values of the classification variable 'flu' more likely, independent of the value of other variables such as 'headache'. Such a network is *monotone in distribution* [10] if higher-ordered configurations of the observable variables make higher-ordered outputs more (isotone) or less (antitone) likely.

When a domain expert indicates that a certain relation ought to be monotone, the joint probability distribution should be such, that this property is reflected in the network. If monotonicity is violated, the probability distribution in the network can be revised in cooperation with the expert. Unfortunately, determining whether a network is monotone in distribution is, in general, highly

* The work of this author was partially supported by the Netherlands Organisation for Scientific Research NWO

intractable ([10]). One approach to overcome this unfavorable complexity, is by approximating the decision (i.e, sometimes have 'undecidable' as outcome) like the algorithm discussed in [10]. This algorithm uses qualitative influences (see e.g. [12] for an introduction in qualitative networks or QPNs) that summarize the direction of the influence of variables by signs. However, the use of these signs of course requires an ordering on the values of the variables under consideration. Such an ordering might be implicit, for example *large* > *medium* > *small* or *true* > *false*. But in practice, there are often variables in a network which do not have such 'natural' orderings. As it is desirable to have as many as possible monotone influences (to minimize the offending context), it is important to *choose* an ordering for the values of these variables that maximizes the number of monotone arcs. Or, equivalently, minimizes the number of '?' signs in the corresponding QPN.

In this paper, we determine the computational complexity of this optimization problem. In Section 2, we introduce some notations and definitions. We show that optimizing the number of monotone arcs is NP-complete and hard to approximate (Section 3). We suggest a branch-and-bound strategy as an exact algorithm in Section 4. Finally, we conclude our paper in Section 5.

2 Preliminaries

Let $\mathbf{B} = (G, \Gamma)$ be a Bayesian network where $G = (V, A)$ is an acyclic directed graph, and Γ, the set of conditional probability distributions, is composed of rational probabilities. Let Pr be the joint probability distribution of $\mathbf{B}$. The conditional probability distributions in Γ are assumed to be explicit, i.e., represented with look-up tables. For any variable $X \in V(G)$, let $\Omega(X)$ denote the set of values that X can take. A node X is denoted as a predecessor of Y if $(X, Y) \in A(G)$. The set of all predecessors of Y is denoted as $\pi(Y)$. If, for a node Y, $\pi(Y)$ is the set $X = \{X_1, \ldots, X_n\}$, the *configuration template* $\mathbf{X}$ is defined as $\Omega(X_1) \times \ldots \times \Omega(X_n)$; a particular instantiation $\mathbf{x}$ of $X_1, \ldots, X_n$ will be denoted as a *configuration* of $\mathbf{X}$.

2.1 Local monotonicity

Monotonicity can be defined as stochastical dominance (monotone in distribution) or in a modal sense (monotone in mode). In this paper, we discuss monotonicity in distribution only, and we focus on local effects, i.e., influences between two variables which are directly connected. A network is *locally monotone* if all qualitative influences along the arcs in the network are either positive or negative.

Definition 1 (local monotonicity). *Let F be the cumulative distribution function for a node $X \in V(G)$, defined by $F(x) = Pr(X \leq x)$ for all $x \in \Omega(X)$. For any arc $(X, Y) \in A(G)$, let $\mathbf{Z}$ denote the configuration template $\pi(Y) \setminus X$, and let $\mathbf{z}$ denote an individual configuration of $\mathbf{Z}$. With (X, Y), a positive influence is associated if $x < x' \rightarrow F(y \mid x\mathbf{z}) \geq F(y \mid x'\mathbf{z})$ for all $y \in \Omega(Y), x, x' \in$*

$\Omega(X)$, and $\mathbf{z} \in \mathbf{Z}$. Similarly, a negative influence is associated with this arc if $x < x' \rightarrow F(y \,|\, x\mathbf{z}) \leq F(y \,|\, x'\mathbf{z})$ for all $y \in \Omega(Y), x, x' \in \Omega(X)$, and $\mathbf{z} \in \mathbf{Z}$. We will denote an arc associated with an positive or negative influence as a isotone, respectively antitone arc. $\boldsymbol{B} = (G, \Gamma)$ is locally monotone if all arcs in $A(G)$ are either isotone or antitone.

2.2 Interpretations

The above notions of monotonicity assumed an implicit *ordering* on the values of the variables involved. Such an ordering is often trivial (e.g., $x > \bar{x}$ and *always* > *sometimes* > *never*) but sometimes it is arbitrary, like an ordering of the values { *trachea*, *mediastinum*, *diaphragm*, *heart* }. Nevertheless, a certain ordering is necessary to determine whether the network is monotone, or to determine which parts of the network are violating monotonicity assumptions. Thus, for nodes where no *a priori* ordering is given, we want to order the values of these nodes in a way that maximizes the number of monotone arcs or the number of nodes with only monotone incoming arcs (depending on the specific application).

We define the notion of an *interpretation* of X to denote a certain ordering on $\Omega(X)$, the set of values of X. Note, that the number of distinct interpretations of a node with k values equals $k!$, the number of permutations of these values. Nevertheless, in practice, the number of values a variable can take is often small. For example, in the ALARM network [2], the number of values is at most four, and in the OESOPHAGEAL network [11] it is at most six. In this paper, we assume that k is small and can be regarded as a fixed constant.

Definition 2 (interpretation). *An interpretation of $X \in V(G)$, denoted I_X, is a total ordering on $\Omega(X)$. For arbitrary interpretations we will often use σ and τ. We use the superscript T to denote a reverse ordering: if $\sigma = (x_1 < x_2 < \ldots < x_n)$, then $\sigma^T = (x_n < \ldots < x_2 < x_1)$. The interpretation set $\mathbf{I_X}$ is defined as the set of all possible interpretations of X. Note that an arc is isotone for a given interpretation σ if and only if it is antitone for σ^T and vice versa, and that the interpretations in $\mathbf{I_X}$ are pairwise symmetric. In the remainder, when $\sigma, \tau \in \mathbf{I_X}$ are distinct, then we also assume that $\sigma \neq \tau^T$.*

2.3 Monotonicity functions and schemes

We define a *monotonicity function*, which determines whether a certain combination of interpretations for the two nodes of an arc makes the arc isotone or antitone. When a node has more than one predecessor (say $\pi(Y) = \{X_1, X_2\}$), the arc (X_1, Y) is monotone for a certain combination of interpretations $\sigma \in \mathbf{I}_{X_1}$ and $\tau \in \mathbf{I}_Y$, when it is isotone for all values[1] of X_2, or when it is antitone for all values of X_2. We define the monotonicity function of (X_1, Y) for a particular *given* value $x_2 \in \Omega(X_2)$ as a *partial monotonicity function*, to emphasise the *conditional* monotonicity of (X_1, Y).

[1] Note that the *ordering* of the elements in $\Omega(X_2)$ is irrelevant for the local monotonicity of (X_1, Y).

Definition 3 ((partial) monotonicity function). *Consider the arc* $x_1 = (X_1, Y) \in A(G)$*, where* Y *has auxiliary predecessors (say* $x_2 \dots x_n$*), whose configuration template we denote with* $\mathbf{Z}_N$*. Assume* $\sigma \in \mathbf{I}_{X_1}$ *and* $\tau \in \mathbf{I}_Y$*. Then* $M_{X_1Y}(\sigma, \tau)$ *is true if and only if* x_1 *is either isotone (denoted* $M^+_{X_1Y}$*) or antitone (denoted* $M^-_{X_1Y}$*) for interpretations* σ *and* τ*, for all possible configurations of* $\mathbf{Z}_N$*. The partial monotonicity function* $M_{X_1Y}(\sigma, \tau \,|\, \mathbf{z}_N)$ *is true if and only if* x_1 *is isotone or antitone for interpretations* σ *and* τ*, given a specific configuration* $\mathbf{z}_N$ *of* $\mathbf{Z}_N$*.*

Observe, that $M_{XY}(\sigma, \tau) = M_{XY}(\sigma^T, \tau) = M_{XY}(\sigma, \tau^T) = M_{XY}(\sigma^T, \tau^T)$ since $M_{XY} = M^+_{XY} \vee M^-_{XY}$, and $M^+_{XY}(\sigma, \tau) \leftrightarrow M^-_{XY}(\sigma^T, \tau)$. Partial monotonicity functions and schemes can be combined for multiple configurations of $\mathbf{Z}_N$. Informally, the combined partial monotonicity function for instantiation x_ϕ and x_ψ is true for a certain combination of interpretations, if the individual partial monotonicity functions are all isotone, or all antitone, for that combination.

Definition 4 (combining partial monotonicity functions). *Consider again the arc* x_1 *as defined before, with* $\mathbf{Z}_N$ *as the configuration template of* $\pi(Y) \setminus X_1$*. Then, for* $\delta \in \{+, -\}$*,*

$$M^\delta_{X_1Y}(\sigma, \tau \,|\, \mathbf{z}_\phi) \wedge M^\delta_{X_1Y}(\sigma, \tau \,|\, \mathbf{z}_\psi) = M^\delta_{X_1Y}(\sigma, \tau \,|\, \mathbf{z}_\phi \wedge \mathbf{z}_\psi)$$

and consequently,

$$\bigwedge_{z_N \in Z_N} M^\delta_{X_1Y}(\sigma, \tau \,|\, \mathbf{z}_N) = M^\delta_{x_1Y}(\sigma, \tau)$$

With every monotonicity function M_{XY}, a binary matrix $\mathbf{M_{XY}}$ is associated, denoted as the *monotonicity scheme* of M_{XY}. Similarly, a *partial monotonicity scheme* $\mathbf{M_{XY|z_N}}$ is associated with the corresponding partial monotonicity function. These matrices have dimensions $\frac{1}{2} \mid \mathbf{I}_X \mid \times \frac{1}{2} \mid \mathbf{I}_Y \mid$, since the interpretations in $\mathbf{I}$ are pairwise symmetric. We will often illustrate these matrices using a grid, where shaded areas denote monotone combinations of interpretations in $\mathbf{I}_X$ and $\mathbf{I}_Y$.

Using these definitions, in the following section we will discuss the problem of optimizing the number of monotone arcs, i.e., choosing an interpretation for the values of all nodes, such that the number of arcs that are isotone or antitone is maximal. Note that a network where some nodes are fixed (i.e., an interpretation is given) can be translated in an equivalent network with non-fixed interpretations, where the number of monotone arcs will be optimal if and only if that particular interpretation is chosen. For example, if the ordering of a node C with values $\{low, mid, high\}$ and degree n is to be fixed at $low < mid < high$, we can enforce this condition by adding $n + 1$ dummy nodes D with $\Omega(D) = \{T, F\}$ and arcs from C to these nodes that are only monotone if C has ordering $low < mid < high$, for example $\Pr(T \,|\, low) = 0.2$, $\Pr(T \,|\, mid) = 0.4$, $\Pr(T \,|\, high) = 0.6$. It can be easily verified that the optimal number of monotone arcs enforces the given ordering on C. In a similar way,

a partial order can be guaranteed, e.g., the variable 'Stereo Sound' with values $\{none, left, right, both\}$ where no obvious ordering for '*left*' and '*right*' exists, but $none \prec left \prec both$ and $none \prec right \prec both$.

3 Optimizing the number of monotone arcs

In this section, we formalize the problem of optimizing the number of monotone arcs, and show that it is NP-complete, i.e., infeasible in general. A similar complexity result is established for the derived problem of optimizing the number of nodes with only monotone incoming arcs. Both problems can be used as a measure for the size of the monotonicity-violating context. Furthermore, we prove that these problems — apart from infeasible to solve exactly — are hard to approximate as well. In the remainder of this section, we assume that the reader is familiar with NP-completeness proofs; more background can be found in textbooks like [6] and [7].

In the formal problem definitions, we assume that the (conditional) probabilities in the network are specified using rationals, rather than reals, to ensure an efficient coding of these probabilities. Since these probabilities are often specified by experts or approximated using learning methods, this is a realistic constraint. Furthermore, we assume that the conditional probabilities in the network are coded explicitly, i.e., using look-up tables, rather than using some computable function. Lastly, for technical reasons we formulate our problems as decision problems (returning 'yes' or 'no'), rather than functions (returning a number).

Max-Local Monotonicity
Instance: Let $\mathbf{B} = (G, \Gamma)$ be a Bayesian network where Γ is composed of rational probabilities, and let Pr be its joint probability distribution. Let $\Omega(X)$ denote the set of values that $X \in V(G)$ can take, and let k be a positive integer $\leq |\ A(G)\ |$.
Question: Is there an interpretation I_X for all $X \in V(G)$, such that the number of arcs in G that are monotone in distribution is at least k?

Max-Nodes-Local Monotonicity
Instance: Let $\mathbf{B} = (G, \Gamma)$ and $\Omega(X)$ be as above, and let k be a positive integer $\leq |\ V(G)\ |$.
Question: Is there an interpretation I_X for all $X \in V(G)$, such that the number of nodes in G that have only incoming arcs that are monotone in distribution, is at least k?

In our hardness proof, we use the Graph 3-Colorability problem, defined in [6]. In this problem, the instance is an undirected graph $G = (V, E)$, and we want to know whether there is a function $f : V \to \{1, 2, 3\}$, such that $f(u) \neq f(v)$ whenever $(u, v) \in E$, i.e., all nodes can be colored with three colors, such that no adjacent nodes have the same color.

3.1 NP-completeness proof

Let $G = (V, E)$ be an instance of the GRAPH 3-COLORABILITY problem. From this undirected graph G, we construct the directed graph $G' = (V', A)$ as follows (See Figure 1):

- if $X \in V(G)$, then $X \in V'$.
- if $(X, Y) \in E(G)$, then $E_1, E_2, E_3, E_4, E_5, E_6 \in V'$.
- if $(X, Y) \in E(G)$, then $(X, E_1), \ldots, (X, E_6), (Y, E_1), \ldots, (Y, E_6) \in A$.
- $\Omega(X) = \{x_1, x_2, x_3\}$ for all $X \in V'$

We number the interpretations of all nodes in V' as follows:

- $i_1 = x_2 < x_1 < x_3$.
- $i_2 = x_1 < x_2 < x_3$.
- $i_3 = x_1 < x_3 < x_2$.

Now, for all nodes E_i we construct a conditional probability table such that $M(I_X, I_{E_i})$ has the following monotonicity scheme:

E_i	E_1	E_2	E_3	E_4	E_5	E_6
I_X	$\{i_1, i_2\}$	$\{i_1, i_2\}$	$\{i_1, i_3\}$	$\{i_1, i_3\}$	$\{i_2, i_3\}$	$\{i_2, i_3\}$
I_{E_i}	$\{i_2, i_3\}$	$\{i_1, i_3\}$	$\{i_1, i_2\}$	$\{i_2, i_3\}$	$\{i_1, i_2\}$	$\{i_1, i_3\}$

and the probability table for the arc (Y, E_i) is such, that (Y, E_i) is a monotone relation if and only if $I_Y = I_{E_i}$. An example of such a table is given in Table 1; the other tables can be generated likewise. Observe, that a graphical representation of these schemes would be a 2×2 square, which is *transposed* from the origin. We claim that, in the thus constructed network, there is a maximum of eight arcs that have a monotone relation, if $I_X = I_Y$, and nine arcs if $I_X \neq I_Y$. We assume, without loss of generality, that $I_Y = i_1$. If we choose $I_{E_i} = e_1$ for all E_i, then all six outgoing arcs from Y to E_i have monotone relations. Now there are two cases:

- $I_X = i_1$. There are two monotone relations: (X, E_2) and (X, E_3). Both E_2 and E_3 have only monotone incoming arcs.
- $I_X = i_2$ or i_3. There are *three* monotone relations: either (X, E_2), (X, E_5) and (X, E_6); or (X, E_3), (X, E_5) and (X, E_6), which all have only monotone incoming arcs.

Note that there is no way to make *more* than three monotone arcs. We will use this construct to prove NP-hardness.

Theorem 1. MAX-LOCAL MONOTONICITY *and* MAX-NODES-LOCAL MONOTONICITY *are NP-complete.*

Proof. Membership of NP is trivial for both problems. Using a certificate that consists of interpretations for all vertices, we can easily test whether at least k arcs are monotone in distribution, or at least k nodes have the property that all

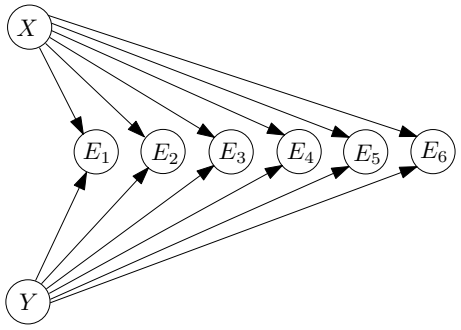

Fig. 1. Construction with 6 extra nodes

$\mathbf{Pr}(E \mid y_1, X)$			
	e_1	e_2	e_3
x_1	0.42	0.30	0.28
x_2	0.28	0.44	0.28
x_3	0.28	0.30	0.42

$\mathbf{Pr}(E \mid y_2, X)$			
	e_1	e_2	e_3
v_1	0.44	0.28	0.28
v_2	0.28	0.44	0.28
v_3	0.28	0.28	0.44

$\mathbf{Pr}(X \mid y_3, X)$			
	e_1	e_2	e_3
v_1	0.44	0.28	0.28
v_2	0.28	0.44	0.28
v_3	0.28	0.28	0.44

Table 1. Conditional probability table for node E_1 with incoming arcs from X and Y

incoming arcs are monotone in distribution. To prove *NP*-hardness, we construct a transformation from the Graph 3-Colorability problem. Let $G = (V, E)$ be an instance of this problem, and let $G' = (V', A)$ be the directed acyclic graph the is constructed from this instance, as described above. If and only if $9\times \mid E \mid$ arcs in G' are monotone, then all nodes X and Y that were adjacent in G, have different interpretations, hence G would be 3-colorable. Since $G' = (V', A)$ can be computed from $G = (V, E)$ in polynomial time, we have a polynomial-time transformation from Graph 3-Colorability to the Max-Local Monotonicity problem, which proves NP-hardness of the latter. A similar argument holds for the number of nodes with only monotone incoming arcs, therefore Max-Nodes-Local Monotonicity is NP-hard as well. □

3.2 Approximation

When confronted with intractable (e.g., NP-hard) problems, a number of options are available. One could try to find polynomial algorithms for particular instances (special cases) of the problem, try to construct an exact algorithm that works reasonably fast most of the time, or try to approximate the problem. In the latter case, one tries to find a solution that is *close* to optimal, in reasonable time. Not all problems are easy to approximate. An example of an NP-hard problem which can be approximated in polynomial time within an arbitrary margin is Scheduling Independent Tasks [3]: given a set of tasks of variable length and a set of processors on which they run, which schedule leads to a minimum

total finishing time? Other problems are very hard to approximate. For example, the well known Travelling Salesman Problem cannot be approximated in general within any fixed constant factor, unless P = NP.

Optimization problems can be classified based on the character of the *performance ratio* of their approximation algorithms. We will give a short introduction on this classification; for a more thorough introduction the reader can refer to e.g. [8], [5], or [1]. For maximization problems, the ratio $R(X, Y)$ of an approximation algorithm Y, given instance X, is defined as $R(X, Y) = \frac{OPT(X)}{APP_Y(X)}$, where $OPT(X)$ denotes the optimal solution for X, and $APP_Y(X)$ denotes the solution, given by algorithm Y. The class of all optimization problems is denoted with NPO. A subset of this class is the class APX. A problem A belongs to APX if it is approximable within a fixed ratio, i.e. there is an algorithm T and a ratio r, such that for all instances X, $R(X, T) \leq r$. A problem A belongs to PTAS (has a polynomial time approximation scheme) if it is approximable within any ratio r in time polynomial in the input size, and it belongs to FPTAS (has a *fully* polynomial time approximation scheme) if this approximation is polynomial in r as well.

In this section, we will show that Max-Local Monotonicity is APX-hard. This hardness result is a very strong indicator that there is no polynomial time approximation scheme for it – otherwise, *all* problems in APX would enjoy such a PTAS – and that the problem can only be approximated within a constant factor. We will reduce Max-3color-Subset [8], a known APX-hard problem, to Max-Local Monotonicity using a so-called *A-reduction*. In [4], an A-reduction is defined as a reduction from A to B, such that an approximation for B within a fixed ratio r implies an approximation for A within a fixed ratio $c(r)$, where c is a computable function $\mathbf{Q} \cap (1, \infty) \rightarrow \mathbf{Q} \cap (1, \infty)$. If there is an A-reduction from A to a known APX-hard problem B, then A is APX-hard as well. We will show that the reduction from 3color constructed in the previous section is actually an A-reduction to Max-3color-Subset.

Theorem 2. Max-Local Monotonicity *A-reduces to* Max-3color-Subset.

Proof. Let $G = (V, E)$ be an instance of Max-3color-Subset, and let $G' = (V', A)$ be the directed acyclic graph the is constructed from this instance, as described in Section 3.1. Let $OPT(3C)$ denote the maximum number of nodes in G that can be colored with three colors, and let $APP_Y(3C)$ be the number of nodes colorable with three colors with a certain approximation algorithm Y. The ratio r of this approximation is then

$$\frac{OPT(3C)}{APP_Y(3C)}$$

By construction, G' has an optimal solution $8 \mid V \mid + OPT(3C)$, and Y would approximate this to $8 \mid V \mid + APP_Y(3C)$, with ratio

$$r' = \frac{8 \mid V \mid + OPT(3C)}{8 \mid V \mid + APP_Y(3C)}$$

But then, there clearly exists a function c such that $r \leq \phi \Rightarrow r' \leq c(\phi)$. □

Corollary 1. MAX-LOCAL MONOTONICITY *is APX-hard.*

4 A branch-and-bound algorithm

In the previous section we proved that there does not exist a PTAS for MAX-LOCAL MONOTONICITY unless APX = PTAS. However, there might exist approximations for MAX-LOCAL MONOTONICITY that are within a fixed ratio r. Nevertheless, r may be very large and such approximations may not be particularly useful. Therefore, we now construct an exact algorithm for this problem, based on a so-called branch-and-bound strategy (see for example [13]). In such a strategy, the set of possible solutions is partitioned (the branch step), and upper (or lower, for minimalization problems) bounds for this partition are calculated. Whenever these bounds are lower than or equal to the current best solution (i.e., further exploration of these branches will not lead to a better solution) the branch is terminated, and other, yet unvisited branches are explored. This procedure continues until all branches terminate (we can return an optimal solution), or a given ratio between current best solution and upper bound is reached (we can return a 'good enough' solution).

4.1 Initial heuristic - a lower bound

In this section we discuss how a lower bound on the number of monotone arcs can be calculated in polynomial time (for fixed k). First we will present a procedure to compute monotonicity schemes efficiently, exploiting a particular property of monotonicity functions. We will distinguish between *factorizing* and *non-factorizing* monotonicity schemes, and we will show how a lower bound heuristic can be calculated, using arcs with factoring monotonicity schemes, in polynomial time.

Trivially, computing a monotonicity scheme for any node takes $O((k!)^2)$, since there are $k!$ interpretations for both ends of the arc, and computing local monotonicity takes linear time. Notice that the complexity of calculating schemes for an arc whose endpoint has multiple predecessors, is proportional in the size of the input (i.e., the conditional probability table). If all other variables in the graph have an arc towards this endpoint, there are $\prod_{i=1}^{n} \mid X_i \mid$ configurations that we need to consider. However, the conditional probability table has size $O(\prod_{i=1}^{n} \mid X_i \mid)$ as well, since we assumed explicit probability representation.

This running time can be reduced to $O(\frac{1}{2}(k!)^2)$ in the worst case, and $O(2(k!))$ in the best case, by exploiting the following observation. If a relation $X \rightarrow Y$ is monotone for two distinct interpretations $\sigma, \sigma' \in \mathbf{I}_X$ given an interpretation $\tau \in \mathbf{I}_Y$, then there are at least two equal columns in the joint probability table, i.e., $\Pr(y_k \,|\, x_i) = \Pr(y_k \,|\, x_j)$ for all $y_k \in \Omega(Y)$. But then, $M_{XY}(\sigma, \tau') = M_{XY}(\sigma', \tau')$ for *all* interpretations $\tau' \in \mathbf{I}_Y$. Of course, in two distinct interpretations[2] there

[2] Note that we defined σ and σ' to be distinct, only if also $\sigma' \neq \sigma^T$.

exist i and j such that $x_i \leq x_j$ in one interpretation and $x_j \leq x_i$ in the other. From this property follows, that two columns in a monotonicity scheme are either equal or disjoint. It suffices to observe that there exists a $\tau \in \mathbf{I}_Y$ such that $M_{XY}(\sigma, \tau) = M_{XY}(\sigma', \tau) = \text{TRUE}$ to conclude that this is the case for *all* $\tau \in \mathbf{I}_Y$.

To compute a lower bound heuristic, we consider only arcs that have *factorizing* monotonicity schemes. We use these factorizing schemes to calculate *allowed sets* for each variable.

Definition 5 (factorizing monotonicity scheme). $\mathbf{M_{XY}}$ *is called factorizing over* $\mathbf{I}_X$ *and* $\mathbf{I}_Y$ *if there exist subsets* $\mathbf{I_X^+} \subseteq \mathbf{I_X}$ *and* $\mathbf{I_Y^+} \subseteq \mathbf{I_Y}$ *such that* $M_{XY}(\sigma, \tau)$ *is true if and only if* $\sigma \in \mathbf{I_X^+}$ *and* $\tau \in \mathbf{I_Y^+}$.

For a variable Z, let $\pi(Z)$ denote its set of predecessors and let $\sigma(Z)$ denote its set of children. Then, if all arcs $(X \in \pi(Z), Z)$ and $(Z, Y \in \sigma(Z))$ have a factoring monotonicity scheme, an interpretation I_Z for Z that is an element of $\bigcap_{X \in \pi(Z)} \mathbf{M_{XZ}} \cap \bigcap_{Y \in \sigma(Z)} \mathbf{M_{ZY}} \neq \emptyset$ is always an interpretation that can be chosen for Z without violating local monotonicity of the network. Of course, not all monotonicity schemes are factoring. If $\pi(Z)_f$ and $\sigma(Z)_f$ denote the predecessors, respectively children of Z such that $(X \in \pi(Z)_f, Z)$, respectively $(Z, Y \in \sigma(Z)_f)$ are arcs with factoring monotonicity schemes, we will denote $\mathcal{M}_Z$ as the *allowed set* of Z, where $\mathcal{M}_Z = \bigcap_{X \in \pi(Z)_f} \mathbf{M_{XZ}} \cap \bigcap_{Y \in \sigma(Z)_f} \mathbf{M_{ZY}} \cap \mathbf{I_Z}$. Note, that the allowed set consists of interpretations that can be chosen, if all arcs *without* factoring monotonicity schemes would be removed. In other words, there exists a network $G' = (V, A')$ where A' is the (possibly empty) set of arcs with factoring monotonicity schemes, and the allowed set of all $Z \in V(G')$ is the set of interpretations that can be chosen without violating monotonicity of G'. Now, we can calculate a lower bound for the maximal number of arcs in G that can be made monotone as follows. We initialise $\mathcal{M}_Z$ to $\mathbf{I_Z}$ for all $Z \in V'$ and A^+ to the empty set, and iteratively consider arcs in A'. If an arc does not cause any allowed set to become empty, it is added to A^+, and $\mathcal{M}$ is adapted for both endpoints of that arc. On the other hand, if the arc does lead to an empty allowed set, it is dismissed. After considering all arcs in A', $\mid A^+ \mid$ is a lower bound.

4.2 Branching and bounding

Using this lower bound, we consecutively branch on the possible interpretations of the nodes, terminating branches whose upper bound is not higher than the current best solution (or lower bound). While different strategies can be followed to choose a node to branch on at any step in the algorithm, a reasonable heuristic is to pick the node that has the highest degree of all unexplored nodes. We fix the interpretation of the variable we branch on (i.e., the allowed set is a singleton, corresponding with the branch value) and calculate how many factorizing arcs remain monotone in the network. This value is added to the number of non-factorizing arcs; this is an upper bound for the total number of monotone arcs in the network.

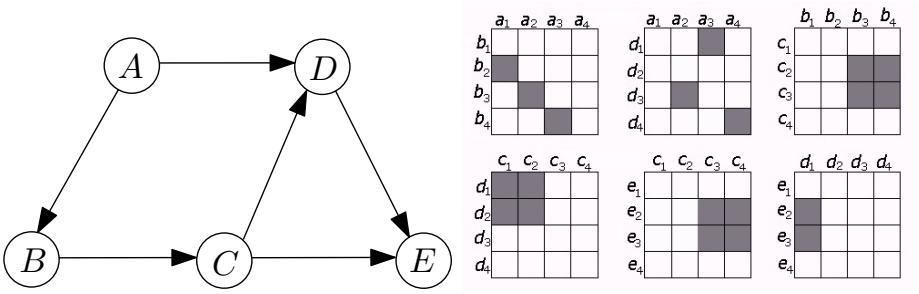

Fig. 2. An example graph

Of course, there are many degrees of freedom in this branch-and-bound strategy. We chose to compute rather loose bounds; one can compute tighter bounds by considering a number of non-factoring arcs that can be made monotone. Nevertheless, the constraints imposed by these arcs might require re-evaluation of all allowed sets in the network, so there is a tradeoff between the tightness of the bounds - and thus the number and depth of the branches - and the time needed to calculate such bounds.

4.3 An example

We will use the graph in Figure 2 as a example to sketch our branch-and-bound algorithm. We assume that, for every variable, only four interpretations are relevant; we will denote a particular interpretation with indexed lowercase variables, e.g., $\mathbf{I_C} = \{c_1, c_2, c_3, c_4\}$. On the right part of Figure 2, the monotonicity schemes for the arcs in the graph are shown. For example, (A, B) is monotone if $I_A = a_1$ and $I_B = b_2$.

We start with the heuristic lower bound calculated in Section 4.1. The factorizing arcs are (B, C), (C, D), (C, E), and (D, E), and if we consider these in this order and calculate the allowed sets for all nodes, we will find that we can make at least three arcs monotone, namely (B, C), (C, D), and (D, E). The lower bound will thus be three in this example. Now we branch on one of the nodes with maximal degree, say C, and explore the branches $I_C = c_1$, $I_C = c_2$, $I_C = c_3$, and $I_C = c_4$, terminating branches with an upper bound lower than three. Eventually, the algorithm will find the optimal solutions $\{I_A = a_3, I_B = b_4, I_C = c_3, I_D = d_1, I_E = e_2 \vee e_3\}$.

5 Conclusion

Optimising the number of monotone arcs in a network, and thus minimising the number of '?'s in the corresponding QPN, is a computationally hard problem, and hard to approximate as well. We proposed a branch-and-bound approach

to calculate optimal orderings. This approach may work rather well in practice with 'real world' networks, provided that the number of values per node is small. However, for networks where some nodes have a large range of possible values, this approach will be infeasible. Other methods must be used in such cases to calculate or approximate an optimal solution. Currently, we are working on an implementation of this branch-and-bound algorithm.

Acknowledgements

This research has been (partly) supported by the Netherlands Organisation for Scientific Research (NWO). The authors wish to thank Linda van der Gaag and Silja Renooij for their insightful comments on this subject and access to the information stored in the Oesophageal Cancer network, and Jesper Nederlof for his work on the implementation of the algorithm. Furthermore, we wish to thank three anonymous reviewers for their useful comments on an earlier version of this paper.

References

1. G. Ausiello, P. Crescenzi, and M. Protasi. Approximate solution of NP optimization problems. *Theoretical Compututer Science*, 150(1):1–55, 1995.
2. I. Beinlich, G. Suermondt, R. Chavez, and G. Cooper. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the Second European Conference on AI and Medicine*. Springer-Verlag, 1989.
3. J. Bruno, E. G. Coffman jr., and R. Sethi. Scheduling independent tasks to reduce mean finishing time. *Communications of the ACM*, 17(7):382–387, 1974.
4. P. Crescenzi. A short guide to approximation preserving reductions. In *12th Annual IEEE Conference on Computational Complexity (CCC'97)*, pages 262–273. IEEE, 1997.
5. P. Crescenzi and A. Panconesi. Completeness in approximation classes. *Information and Computation*, 93:241–262, 1991.
6. M. R. Garey and D. S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., San Francisco, 1979.
7. C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
8. C. H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991.
9. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, Palo Alto, 1988.
10. L. C. van der Gaag, H.L. Bodlaender, and A. Feelders. Monotonicity in Bayesian networks. In *Twentieth Conference on Uncertainty in Artificial Intelligence*, pages 569–576. AUAI Press, 2004.
11. L. C. van der Gaag, S. Renooij, C. L. M. Witteman, B. M. P. Aleman, and B. G. Taa. Probabilities for a probabilistic network: a case study in oesophageal cancer. *Artificial Intelligence in Medicine*, 25:123–148, 2002.
12. M. P. Wellman. Fundamental concepts of qualitative probabilistic networks. *Artificial Intelligence*, 44(3):257–303, 1990.
13. L. A. Wolsey and G. L. Nemhauser. *Integer and Combinatorial Optimization*. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley, 1988.