Gated Bayesian Networks

Marcus Bendtsen and Jose M. Peña
ADIT, IDA, Linköping University, Sweden
marcus.bendtsen@liu.se, jose.m.pena@liu.se

Abstract. This paper introduces a new probabilistic graphical model called *gated Bayesian network* (GBN). This model evolved from the need to represent real world processes that include several distinct phases. In essence a GBN is a model that combines several Bayesian networks (BN) in such a manner that they may be active or inactive during queries to the model. We use objects called *gates* to combine BNs, and to activate and deactivate them when predefined logical statements are satisfied. These statements are based on combinations of posterior probabilities of the variables in the BNs. Although GBN is a new formalism there are features of GBNs that are similar to other formalisms and research, including influence diagrams, context-specific independence and structural adaptation.

Keywords. Probabilistic graphical models, Bayesian networks, influence diagrams, decision support, dynamic systems

Introduction

Bayesian networks (BNs) have been successfully used to reason under uncertainty within many domains. They allow for a qualitative model using a directed acyclic graph (DAG) that can visualise the independencies amongst variables in a joint probability distribution. This qualitative model can then be quantified by specifying certain marginal and conditional probability distributions so that queries can be made to the network regarding posterior probabilities. See [1] and [2] for more details.

Despite their popularity and advantages, there are situations where a BN model is not enough. For instance, when trying to model the process of buying and selling stock shares, we wanted to model the investors continuous flow between looking for potential entry and exit points. The investor can be seen as being in two distinct phases: either looking for an opportunity to buy into the market, or an opportunity to sell and exit the market. These two phases can be very different and the variables included in the BNs modelling them are not necessarily the same or even dependent on each other. The need to switch between two different BNs was the foundation for the probabilistic graphical

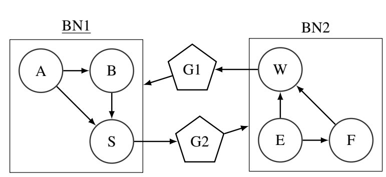

Figure 1. Gated Bayesian network

model presented herein. In Figure 1 we present a simple gated Bayesian network (GBN) that uses two different BNs (BN1 and BN2).

During the development of GBNs other potential uses were found. For instance, the model allows for multiple independent BNs, possibly having different time granularities, to influence the decision to switch phase together.

We will first present a formal definition of the GBN, such that the reader is able to follow along four examples that demonstrate how GBNs can be used to model general real world situations. Once we have exemplified the use of GBNs we will present an algorithm that can be used in conjunction with GBNs to handle observed evidence. We will also discuss how GBNs fit in with other formalisms and research development.

1. Formal definition

A GBN is a probabilistic graphical model that combines multiple BNs using objects called *gates*. These gates allow for *activation* and *deactivation* of the different BNs in the model.

Definition 1 (GBN) *A GBN can be seen as a set of gates ($G(GBN)$), a set of BNs ($BN(GBN)$) and a set of edges ($E(GBN)$) that connect the gates with the BNs. Let $ABN(GBN)$ be the set of active BNs and $IBN(GBN)$ the set of inactive BNs. $ABN(GBN)$ and $E(GBN)$ cannot be empty. A BN cannot belong to both $ABN(GBN)$ and $IBN(GBN)$ at the same time. Each BN can be seen as a set of nodes and a set of edges.*

$$
\begin{aligned}
&GBN = \{G(GBN), BN(GBN), E(GBN)\}\\
&ABN(GBN), E(GBN) \neq \emptyset\\
&if\ BN_i \in ABN(GBN) \Rightarrow BN_i \notin IBN(GBN)\ ,\ BN_i \in BN(GBN)\\
&V(BN_i) = \{all\ nodes\ in\ BN_i\}\ ,\ BN_i \in BN(GBN)\\
&E(BN_i) = \{all\ edges\ in\ BN_i\}\ ,\ BN_i \in BN(GBN)
\end{aligned}
$$

Definition 2 (Connections) *The edges $E(GBN)$ are always directed and connect either a node in $V(BN_i)$ with a gate in $G(GBN)$ or an entire BN in $BN(GBN)$ with a gate in $G(GBN)$. An edge between a node and a gate is always directed away from the node towards the gate. An edge that connects an entire BN and a gate can be directed either way.*

Definition 3 (Parent/child) *When a node is connected to a gate we consider the BN to which the node belongs to be a parent of the gate. When an entire BN is connected to a gate the parent/child relationship is defined by the direction of the edge.*

Definition 4 (Trigger node) *A node that is connected with a gate using an edge is called a trigger node. All nodes that are connected to a gate make up the gate's trigger nodes.*

Definition 5 (Trigger logic) *Each gate has to define its own trigger logic. The trigger logic has to be a logical statement about the gate's trigger nodes' posterior probability of taking a specific value given some evidence. If a trigger node belongs to an inactive BN, then any logical evaluation of the trigger node's posterior probability will default to false.*

$$TL(G_i) = trigger\ logic\ of\ G_i\ ,\ G_i \in G(GBN)$$

Definition 6 (Triggering, activation and deactivation) *If evidence is supplied to a GBN that leads to the trigger logic for some gate being satisfied, then the gate is said to trigger. When a gate triggers it activates all its child BNs and deactivates all its parent BNs. If several gates trigger due to the same set of evidence then the union of all child BNs are activated and the union of all parent BNs minus the union of all child BNs are deactivated.*

$$
\begin{aligned}
UCBN &= Union\ of\ all\ child\ BNs\ to\ triggered\ gates \\
UPBN &= Union\ of\ all\ parent\ BNs\ to\ triggered\ gates \\
BNs\ to\ activate &= UCBN \\
BNs\ to\ deactivate &= UPBN \setminus UCBN
\end{aligned}
$$

Definition 7 (Entire BN as parent) *According to Definition 3 a BN can be a parent to a gate without supplying any trigger nodes to the gate. This tells the model that the BN would like to be deactivated in case the gate triggers, but does not want to participate in the trigger logic.*

Definition 8 (Memory and evidence handling) *Handling available evidence consists in iterating through the following step until no change occurs: First, update the variable instantiations of BNs that are active, then evaluate the trigger logic of all gates, which may activate some BNs according to Definition 6. Once the previous loop has completed, BNs are deactivated according to Definition 6. The active/inactive state of BNs, as well as variable instantiations, are remembered between evidence sets. This definition is formalised later in an algorithm (Algorithm 1).*

Remark 1 (Supplying evidence) *Due to the nature of GBNs, users may not be aware of which BNs are active and which are not. Since evidence can be discarded from some sets it is important that all available evidence is given to the model each time new evidence is made available, even for variables for which the evidence might not have changed since the last set.*

2. Modelling capabilities

We will here present four examples that aim to highlight the modelling capabilities of GBNs.

2.1. Sequence of phases

Imagine that a potential investor is tempted to buy stock shares. We can look at the potential investor as being in one of two distinct phases: either looking for an opportunity to enter the market, or looking for an opportunity to exit the market.

The investor has found a set of rules for when to enter the market (A, B). This set of rules can be defined based on historic data, economical reports, etc. They have already been modelled in a BN (BN1). BN1 also contains a variable (S) that gives the probability of a successful investment, conditioned on the variables A and B.

There are also a set of rules (E, F) that are useful when considering selling the stock. E and F influences the variable W, which models the probability that the potential of an investment is peaking and that it is time to exit the market. The investor wants to model

Table 1. Trigger logic for gates G1 and G2

TL(G1) : $p(W = 1 \mid \mathbf{e}) > 0.7$
TL(G2) : $p(S = 1 \mid \mathbf{e}) > 0.9$
e is some evidence that has been supplied

Table 2. Evidence sets sent to GBN

Set 1 : A = 1, B = 0
Set 2 : A = 1, B = 1, E = 1
Set 3 : A = 1, B = 0, E = 0, F = 1
Set 4 : A = 1, B = 0, E = 1, F = 1

the variables E, F and W separately from A, B and S so that they do not influence each other, as such they have been modelled in their own BN (BN2).

To be able to model the behaviour of the investor buying and selling, a GBN is created using BN1 and BN2 as well as two gates (Figure 1). In the initial state of the GBN, we can see that BN1 has an underscore in its label, indicating that it is an active BN. BN2 does not have the underscore, and thereby is inactive.

From the node S there is a directed edge to the gate G2 (indicating that S is a trigger node of G2 and that BN1 is a parent of G2) and from the node W there is a directed edge to G1 (indicating that W is a trigger node of G1 and that BN2 is a parent of G1). From G2 there is a directed edge to BN2 and from G1 there is a directed edge to BN1 (showing parent/child relationships).

2.1.1. Numerical example of sequence of phases

Assume that we have the model described in Section 2.1 and Figure 1. Table 1 gives the trigger logic for the gates in the network. For instance, G1 will trigger if the posterior probability, given some evidence, of W being 1 is greater than 0.7.

The potential investor sits down by the trading terminal and starts using the new model. Each time data is made available concerning any of the variables A, B, E or F they are presented to the model. According to Remark 1 it is necessary that all available evidence is sent, and not only evidence for variables that the user knows have changed. Here we will leave S and W out, because we assume that they can not be observed directly. Evidence is then sent four times according to the sets in Table 2.

- Set 1 : Variables A and B belong to BN1 which is active, the variables are instantiated according to the evidence. Assume however that TL(G2) = $false$, so G2 does not trigger. TL(G1) will also be false since W belongs to an inactive BN (Definition 5). At this point in time we have not observed variables E and F, so we do not supply any evidence for these variables.
- Set 2 : Variables A and B are updated as before, this time we will assume that $p(S = 1 \mid A = 1, B = 1) > 0.9$. G2 then triggers and activates BN2 (Definition 6). BN2 is now active and so the variable E is instantiated according to the evidence (Definition 8). All parent BNs to triggered gates that are not children of triggered gates are deactivated (Definition 6). This means that BN1 is deactivated. This triggering is a cue for the investor that it is an opportune time to enter the market.
- Set 3 : Variables E and F instantiations are updated since they now belong to an active BN. Assume that TL(G1) = $false$, and due to Definition 5 we know that TL(G2) = $false$. Evidence for A and B are discarded.
- Set 4 : Variables E and F are updated. Assume that this time TL(G1) = $true$. G1 then triggers and activates BN1. Since BN1 has been activated, A and B are (due to Definition 8) updated with new evidence A = 1 and B = 0. We will assume that TL(G2) = $false$. BN2 is now deactivated. This triggering is a cue for the investor that it is an opportune time to exit the market.

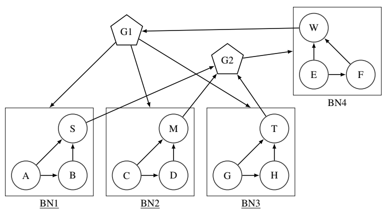

Figure 2. Parallel measurements

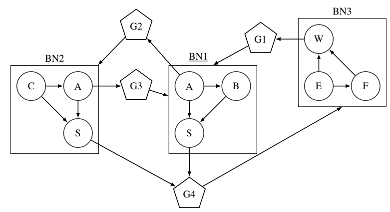

Figure 3. State Change

2.2. *Parallel measurements*

The variables that the investor used in the GBN in Figure 1 by no means model the entire picture of a stock market. There are numerous variables that might effect the pricing of a certain stock, from price, volume, fundamental data, announcements and rumours. Different investors will weigh the importance of this information differently.

Let us assume that our investor would like to consider different measurements when deciding to buy stock shares. Rather than using one BN to only model the probability of success, the investor also wants to take into consideration the confidence of the company and market sentiment. The graphical model is shown in Figure 2. BN1 only represents price and volume data (A, B) and the probability of a successful trade (S), BN2 represents fundamental data about the stock (C, D) and the confidence in the company (M), BN3 represents data taken from text mining of news articles (G, H) and the sentiment of the entire market (T). BN1-3 all make up the phase where the investor is looking to buy a stock in a market. BN4 is only based on price and volume and make up the phase where the investor is looking to sell the previously bought stock.

The three BNs (BN1, BN2 and BN3) model three distinct different measurements that follow different time granularities. In BN1, price and volume data is made available each time the underlying stock price changes, which might be several times per second. The fundamental data about the company (BN2), which could be taken from annual reports, might only change once a year. Finally, input from the text mining task (BN3) is made available when news articles, tweets, etc. are posted.

When the investor sits down in front of the trading terminal any observed evidence for the variables A, B, C, D, E, F, G and H are fed into the model. Even if only A changes the investor supplies all evidence available (Remark 1). By defining TL(G2) using all trigger nodes we allow BN1, BN2 and BN3 to all participate in the decision to trigger the gate. TL(G2) could for instance be: $p(S = 1 \mid \mathbf{e}) > 0.7 \wedge p(M = 1 \mid \mathbf{e}) > 0.5 \wedge p(T = 1 \mid \mathbf{e}) > 0.3$.

2.3. *State change*

In the previous two examples we have modelled situations where the triggering of a gate has been connected to an action (e.g. buy or sell). However this tight connection between a gate triggering and an action does not always have to be the case. Sometimes we will switch between BNs because we want another set of variables to influence our decision. Imagine a country where there is a set of policies that are prioritised when the country is at peace, and a set of policies that are prioritised for when the country is at war. The decision is the same (e.g. should we build more schools?) but the variables that influence the decision changes.

Figure 3 is an extension of the sequence of phases example (Section 2.1). We have introduced a new BN (BN2) as part of the first phase (i.e. prior to buying the stock). We have also reused variables A and S in BN1 and BN2. These variables both represent the same underlying phenomenon, however since they belong to different BNs they may have different conditional probability distributions. If G4 triggers then this means that the investor should buy stock, however if G2 or G3 triggers this just means that we are shifting our attention to some other variables. The TL(G4) could be defined as $p(S_1 = 1 \mid \mathbf{e}) > 0.7 \vee p(S_2 = 1 \mid \mathbf{e}) > 0.8$ where S_1 is the S in BN1 and S_2 is the S in BN2. If BN2 is inactive then according to Definition 5 the statement $p(S_2 = 1 \mid \mathbf{e})$ will always be false, yet the gate can still trigger since $p(S_1 = 1 \mid \mathbf{e}) > 0.7$ could evaluate as true, making the entire logic statement true.

2.4. Combined example

For an investor it is possible to *"go long"* or *"go short"*. In essence going long means that the investor makes money if the stock increases in value, and going short means the investor makes money if the stock decreases in value.

In Figure 4 we have hidden some of the details of the BNs. The entire BN1 is connected to G4 and G5 as a parent, the entire BN2 is connected to G3 as a parent and the entire BN3 is connected to G2 as a parent. These connections tell us that the BNs would like to be deactivated in case the gate triggers (Definition 6), but they do not want to participate in the trigger logic (Definition 7).

BN1 predicts a positive or negative sentiment of the entire market. This sentiment basically tells us if market actors are expecting an increase in market value or a decrease.

BN2 represents a BN that uses data to decide whether or not to go long the stock. Assume that the model is presented some evidence that sets TL(G2) = *true*. This will activate BN2 as expected, however BN1 will not be deactivated since BN1 is both a parent and a child to G2 (Definition 6). BN3 on the other hand is only a parent to G2, and it will be deactivated (although it already was).

BN2 will start updating its variable instantiations with the supplied evidence, whilst BN1 will continue to use evidence to predict market sentiment. In case the market senti-

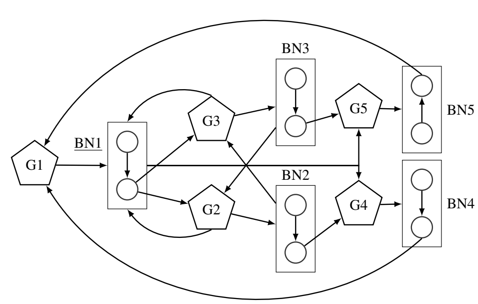

Figure 4. Combined example - Gated Bayesian network - Initial state

ment turns we want to know this before making any decisions. Let us assume that market sentiment turns and TL(G3) = *true*. BN3 will be activated, as will BN1. Since BN2 is a parent of G3 it will be deactivated. So now we are looking for opportunities to go short, whilst still processing evidence regarding market sentiment in BN1.

Assume that at some point TL(G5) = *true* (i.e. the investor goes short). This activates BN5 and deactivates BN3. Since BN1 is also a parent of G5 it is also deactivated.

3. Handling evidence

As we have seen in the examples there are several things that happen when evidence is supplied to a GBN, including variable instantiation and gate triggering. Algorithm 1 formalises evidence handling in GBNs.

On line 5 the main loop of the algorithm starts. In each iteration variable instantiations are updated for any active BNs. The algorithm then finds those gates that have not yet triggered and send them to the TRIGGER function on line 21, along with the supplied evidence. The TRIGGER function will return with any gates that triggered, and these will be added to the set ATG, which contains all the triggered gates. For each of the gates that triggered during this iteration of the loop (that started on line 5) their parent and child BNs are stored. Before the loop starts again all child BNs that belong to triggered gates are activated. This is done in order to not enforce any ordering of the gates, so we can check the trigger logic for the gates in any order, and the same gates will trigger regardless. As long as there are gates that trigger the loop will continue. Once the loop is done, all BNs that are parents of gates that triggered, but are not children of any triggered gates, are deactivated. The deactivation is done outside the loop for the same reasoning of unordered gates previously mentioned. Finally all triggered gates are returned.

Notice that on line 9 we are creating a set of gates that belong to the GBN but have not yet triggered. It is this set of gates that are sent to the TRIGGER function on line 10. So once a gate has triggered it can not be triggered again. Therefore the algorithm will always terminate, if not before then at least once all gates have triggered. This prevents any form of oscillation or infinite loop of triggering gates to happen.

The function TRIGGER on line 21 will simply loop over the gates that have not yet triggered, evaluate their trigger logic, and if it is satisfied add the gate to the set of triggered gates. This set of triggered gates is then returned to the calling function.

4. Other formalisms and research

We have not found any research that focuses on developing models that switch between multiple BNs based on logical statements of posterior probabilities. However some of the features of GBNs that have been highlighted throughout the examples have similarities to other research and formalisms.

4.1. Bayesian networks and influence diagrams

It might be tempting to remove the gates in GBNs and simply replace them with nodes, and by doing so creating one large BN. However there are some problems when attempt-

Algorithm 1 Evidence handling

```
function EVIDENCE(GBN, Z)                              ▷ Z is a set of evidence
    UCBN ← { }                                         ▷ children BNs of triggered gates
    UPBN ← { }                                         ▷ parent BNs of triggered gates
    ATG ← { }                                          ▷ all gates that triggered due to Z
    repeat
        for all BN_i in ABN(GBN) do
            Instantiate V(BN_i) according to Z
        end for
        NotTriggered ← G(GBN) \ ATG
        Triggered ← TRIGGER(NotTriggered, Z)
        ATG ← ATG ∪ Triggered
        for all G_t in Triggered do
            UCBN ← UCBN ∪ children of G_t
            UPBN ← UPBN ∪ parents of G_t
        end for
        activate all BNs in (UCBN)
    until Triggered is empty
    deactivate all BNs in (UPBN \ UCBN)
    return ATG
end function
function TRIGGER(NotTriggered, Z)
    Triggered ← { }
    for all G in NotTriggered do
        trigger ← EVALUATE(TL(G))
        if trigger then
            Triggered ← Triggered ∪ {G}
        end if
    end for
    return Triggered
end function
function EVALUATE(TriggerLogic)
    Return evaluation of TriggerLogic. This evaluation includes posterior probability queries to appropriate BNs.
end function
```

ing to do this. We will not discuss this in detail here, but instead we will offer a few observations that should make clear some of these problems.

Adding a node between the networks inevitably allows information to flow between the variables in the previously independent BNs. Also GBNs are allowed to be cyclic, so swapping the gates for nodes would in some cases create an illegal BN.

When creating GBNs we have an extra building block that is not available when creating BNs. When building BNs we estimate marginal and conditional probabilities as part of the building process, however in GBNs we are also interested in, and need to work with, posterior probabilities. This is one of the main differences between BNs and GBNs, which makes it hard or impossible to replace gates with nodes.

Influence diagrams represent dependencies among random variables and decisions [3]. Already in the initial definition there was a strong connection made between decisions and their direct influence on some cost, i.e. the underlying problem is to maximise some expected utility. Influence diagrams can, in a clear way, supply a set of decisions to form a policy that suits a specific problem domain. Influence diagrams can be explained as decision trees, and in the tree a path is found that maximises the expected utility.

GBNs can also be seen as a decision support model, however they are not as tightly connected to an exact cost function. GBNs do not produce a set of decisions that maximise the expected utility, rather GBNs only model when to activate or deactive BNs based on posterior probabilities. If thought of as a decision tree, GBNs allow the user to go into a branch, but then go back and reverse the decision in a way that is not strongly defined in influence diagrams (see section 2.4 for an example of this where the decision was made to look for long opportunities and then reversed to look for short opportunities).

The lack of policy generation and expected utility maximisation has its roots in the initial need for GBNs to be able to switch between BNs, rather than trying to choose an optimal decision. Decisions can be made based on observing which gates trigger in a GBN, but a gate triggering is not formally defined as a decision being made, rather an observation of posterior probabilities.

4.2. Similarity to other research

Taking a large BN and dividing it into to smaller subnets has been researched for some time. Notable contributions include *multiply sectioned Bayesian network* (MSBN) [4, 5] and *agent encapsulated Bayesian network* (AEBN) [6]. Although these models all section the BNs into smaller parts, they still come together to create one large BN.

We see some similarities between features of GBNs and what is known as context-specific independence (CSI) [7]. Specifically the highlights of the state change example where variables can sometimes be independent of each other and sometimes not, depending on what evidence has been supplied.

GBNs are sensitive to the order of the evidence supplied. Similar evidence sensitivity can be found in research regarding structural adaptation [8] and query based diagnostics [9]. These two approaches add or remove nodes and edges from the graphical model while GBNs do not handle the entry of new variables or dependencies into the model.

5. Future work

GBNs have yet to prove themselves useful in any real world application, as such it is one of the main paths of future work. The financial problem domain presented in the examples can be modelled in many different ways and is therefore a good candidate to test the features of GBNs. In our development we have come across several extensions of GBNs that may be of interest in the future. This includes explicitly adding decision nodes to gates that are bound to decisions, as well as the use of utility nodes (as in influence diagrams) that also could be part of the trigger logic (e.g. constraining a gate from triggering unless an expected utility is found to be sufficient).

There is an open question regarding how trigger logic is constructed. Presently, trigger logic is assumed to be decided using a combination of experts and data. However it could be possible to find ways of learning trigger logic entirely from data.

6. Conclusion

Based upon the need to represent a process that goes back and forth between two distinct phases we have theoretically defined a new probabilistic graphical model. We have given a formal definition of GBNs as well as an algorithm to handle new evidence when it is made available. In the examples enclosed we have highlighted features of GBNs that we hope will be useful in solving the initial problem. During the development we also found other situations where the model may be found useful, such as combining BNs with different time granularities.

We have used the domain of investment to exemplify GBNs, however we see other areas where GBNs may demonstrate useful. In a medical setting a doctor could choose between different tests and medicines given a patient with extreme high fever, however choose between a different set of tests and medicines if the fever is normal, this could be modelled similarly to the state change example. Interactive help guides or failure analysis systems could use different BNs depending on the response of the user, following a sequence of phases.

Acknowledgment

The second author is funded by the Center for Industrial Information Technology (CENIIT) and a so-called career contract at Linköping University, and by the Swedish Research Council (ref. 2010-4808).

References

[1] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman Publ Incorporated, 1988.

[2] Finn V. Jensen and Thomas D. Nielsen. *Bayesian Networks and Decision Graphs*. Springer, 2007.

[3] Ronald A. Howard and James E. Matheson. Influence Diagrams. *Readings on the Principles of Decision Analysis*, pages 719–762, 1984.

[4] Yang Xiang. Multiply Sectioned Bayesian Networks and Junction Forests for Large Knowledge-Based Systems. *Computational Intelligence*, 9(2):171–220, 1993.

[5] Yang Xiang and Victor Lesser. Justifying Multiply Sectioned Bayesian Networks. In *Proceedings of the Fourth International Conference on Multi-Agent*, pages 349–356, Boston, 2000.

[6] Scott Langevin, Marco Valtorta, and Mark Bloemeke. Agent-Encapsulated Bayesian Networks and the Rumor Problem. In *Proceedings of the Nineth International Conference on Autonomous Agents and Multiagent Systems*, pages 1553–1554, 2010.

[7] Craig Boutilier, Nir Friedman, Moises Goldszmidt, and Daphne Koller. Context-specific independence in Bayesian networks. In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence*, pages 115–123, 1996.

[8] Søren H. Nielsen and Thomas D. Nielsen. Adapting Bayes Network Structures to Non-Stationary Domains. *International Journal of Approximate Reasoning*, 49:379–397, 2008.

[9] John M. Agosta, Thomas R. Gardos, and Marek J. Druzdzel. Query-Based Diagnostics. In *Proceedings of the Fourth European Workshop on Probabilistic Graphical Models*, pages 1–8, 2008.