Journal of Artificial Intelligence Research 49 (2014) 601-633 Submitted 10/13; published 04/14

Algorithms and Applications for the Same-Decision Probability

Suming Chen SUMING@CS.UCLA.EDU
Arthur Choi AYCHOI@CS.UCLA.EDU
Adnan Darwiche DARWICHE@CS.UCLA.EDU
Computer Science Department
University of California, Los Angeles
Los Angeles, CA 90095

Abstract

When making decisions under uncertainty, the optimal choices are often difficult to discern, especially if not enough information has been gathered. Two key questions in this regard relate to whether one should stop the information gathering process and commit to a decision (stopping criterion), and if not, what information to gather next (selection criterion). In this paper, we show that the recently introduced notion, *Same-Decision Probability (SDP)*, can be useful as both a stopping and a selection criterion, as it can provide additional insight and allow for robust decision making in a variety of scenarios. This query has been shown to be highly intractable, being $\mathrm{PP}^{\mathrm{PP}}$-complete, and is exemplary of a class of queries which correspond to the computation of certain expectations. We propose the first exact algorithm for computing the SDP, and demonstrate its effectiveness on several real and synthetic networks. Finally, we present new complexity results, such as the complexity of computing the SDP on models with a Naive Bayes structure. Additionally, we prove that computing the non-myopic value of information is complete for the same complexity class as computing the SDP.

1. Introduction

Probabilistic graphical models have often been used to model a variety of decision problems, e.g., in medical diagnosis (Pauker & Kassirer, 1980; Kahn, Roberts, Shaffer, & Haddawy, 1997; van der Gaag & Coupé, 1999), fault diagnosis (Lu & Przytula, 2006), classification (Friedman, Geiger, & Goldszmidt, 1997; Ramoni & Sebastiani, 2001; Jordan, 2002), troubleshooting (Heckerman, Breese, & Rommelse, 1995), educational diagnosis (Butz, Hua, & Maguire, 2004; Arroyo & Woolf, 2005; Millán, Descalco, Castillo, Oliveira, & Diogo, 2013), and in intrusion detection (Kruegel, Mutz, Robertson, & Valeur, 2003; Modelo-Howard, Bagchi, & Lebanon, 2008). In these and similar applications, a decision maker is typically in a position where they must decide which tests to perform or observations to make in order to make a better informed decision. Perhaps more critically, a decision maker must also decide when to stop making observations and commit to a particular decision.

The *Same-Decision Probability (SDP)* was recently proposed by Darwiche and Choi (2010), in order to help *quantify* the robustness of a decision, in the context of decision-making with Bayesian networks. In short, the SDP is the probability that we would make the same decision, if we were to perform further observations that have yet to be made. As such, the SDP can be treated as a measure for a decision's *robustness* with respect to unknown variables, quantifying our confidence that we would make the same decision, even if we made further observations.

In this paper, we show how we can apply the SDP as a tool for information gathering, in particular, as a way to determine if we should *stop* information gathering (as a stopping criterion), and if not, which pieces of information to gather next (as a selection criterion). We compare the SDP to classical stopping and selection criteria through illustrative examples. For instance, we demonstrate that the SDP can distinguish between stable and unstable decisions that are indistinguishable by classical criteria. Additionally, we also show that there are scenarios where classical criteria may call for performing further observations, but where the SDP indicates that our decision is unlikely to change.

Notably, the SDP has been shown to be highly intractable, by Choi, Xue, and Darwiche (2012), and the exact computation of the SDP has been limited to toy examples, with few variables, through brute-force enumeration. In this paper, we propose the first exact algorithm for computing the SDP. This algorithm can be applied to real-world networks that are out of the scope of both brute-force enumeration and previously proposed approximation algorithms, and can be further applied to synthetic networks with as many as 100 variables. We further provide new complexity results on the SDP, which both highlight its relative intractability (even in Naive Bayes networks), but also its relationship to a broader class of expectation computation problems, emphasizing the broader importance of developing effective algorithms for the SDP and related problems.

Our paper is thus structured as follows. We first introduce notation and discuss some common stopping and selection criteria in Section 2. We then review our previously introduced work on the SDP in Section 3. In Section 4, we discuss how the SDP can be applied as both a stopping criterion and as a selection criterion. In Section 5, we present a novel exact algorithm for computing the SDP and discuss experimental results in Section 6. In Section 7, we present some recent complexity results on the SDP. We then conclude our paper in Section 8.

2. Related Work

When making decisions under uncertainty, it may be difficult to finalize a decision in the presence of unobserved variables. Given these unobserved variables, there are two fundamental questions. The first question is whether, given the current observations, the decision maker is ready to commit to a decision. We will refer to this as the *stopping criterion* for making a decision. Assuming the stopping criterion is not met, the second question is what additional observations should be made before the decision maker is ready to make a decision. This typically requires a *selection criterion* based on some measure for quantifying an observation's *value of information* (VOI). In this section, we first introduce some necessary notation, and then review some commonly used stopping and selection criteria.

2.1 Notation

Throughout this paper, we use standard notation for variables and their instantiations, where variables are denoted by upper case letters X and their instantiations by lower case letters x. Additionally, sets of variables are denoted by bold upper case letters $\mathbf{X}$ and their instantiations by bold lower case letters $\mathbf{x}$. We assume that the state of the world is described over random variables $\mathbf{X}$, where the evidence $\mathbf{E} \subseteq \mathbf{X}$ includes **all** known variables, and where hidden variables $\mathbf{U} \subseteq \mathbf{X}$ include **all** unknown variables. By definition, $\mathbf{E} \cap \mathbf{U} = \emptyset$ and $\mathbf{E} \cup \mathbf{U} = \mathbf{X}$. We often discuss the ramifications of observing a subset of hidden variables $\mathbf{H} \subseteq \mathbf{U}$ on decision making. Furthermore, we use $D \in \mathbf{U}$ to denote the main hypothesis variable that forms the basis for a decision.[1]

2.2 Stopping Criterion

Given that there are hidden variables in our model and we have the choice of whether or not to observe some subset, a stopping criterion determines when we stop the process of information gathering and commit to a decision. Note that we are concerned with making a decision based on some hypothesis variable, such as the state of a patient's health. For a stopping criterion, the most basic approach used in a variety of domains is to commit to a decision once the belief about a certain event crosses some threshold, as is done by Pauker and Kassirer (1980), Kruegel et al. (2003), and Lu and Przytula (2006). However, this approach may not be robust, as further observations may cause the belief about the event to fall below the threshold. Van der Gaag and Bodlaender (2011) note the possibility of this and pose the `STOP` problem, which asks whether or not the present evidence gathered is sufficient for diagnosis, or if there exists further relevant evidence that can and should be gathered.

Other approaches involve ensuring that the uncertainty surrounding the decision variable is sufficiently reduced. For instance, Gao and Koller (2011) stop information gathering when 1) the conditional entropy of the interest variable is reduced beyond some threshold or 2) the margin between the first and second most likely states of the interest variable is above some threshold. In any case, it is clear that threshold-based stopping criteria are ubiquitous for decision making under uncertainty.

Alternatively, there are also several stopping criteria that involve the existence of a *budget*, which can be an abstract quantity to represent the available resources that can be used for information gathering. The budget may be representative of the number of observations that are allowed (Modelo-Howard et al., 2008; Munie & Shoham, 2008; Yu, Krishnapuram, Rosales, & Rao, 2009; Chen, Low, Tan, Oran, Jaillet, Dolan, & Sukhatme, 2012a), or in terms of a "monetary" amount that may be spent on observations of varying cost (Greiner, Grove, & Roth, 2002; Krause & Guestrin, 2009; Bilgic & Getoor, 2011). In the context of a budget, the general stopping criterion is then to continue to make observations until the budget is completely expended, as is done by Modelo-Howard et al. (2008) and Munie and Shoham (2008). Krause and Guestrin (2009) and Bilgic and Getoor (2011) note that the budget should be expended with the caveat that the *value of information* of an observation is at *least* the *cost* of the observation.

2.3 Selection Criterion: Value of Information

Should the stopping criterion determine that further observations are necessary, a *selection criterion* is then used to determine which variables should be selected for observation. Ideally, we want to observe all variables that will give us additional information with regards

1. The work presented in this paper can be extended to the case of multiple hypothesis variables, but we focus here on the case of one hypothesis variable for simplicity.

to our decision variable. However, due to resource constraints (such as a limited budget) this is often not possible. In this basic approach, a common selection criterion is to then select observations that will minimize the conditional entropy of the decision variable (Vomlel, 2004; Lu & Przytula, 2006; Krause & Guestrin, 2009; Yu et al., 2009; Zhang & Ji, 2010; Gao & Koller, 2011; Ognibene & Demiris, 2013; Shann & Seuken, 2013). The entropy of a variable X is defined as:

$$H(X) = -\sum_{x} Pr(x) \log Pr(x) \tag{1}$$

and is a measure of the uncertainty of the variable's state — if the entropy of a variable is high, that means that there is much uncertainty on what value that variable takes.[2] The uncertainty of the decision variable's true state makes it difficult to make a decision. Thus, a natural selection criterion is to observe variables to minimize the conditional entropy of the decision variable, where the conditional entropy of variable D *given* variable X is defined as:

$$H(D \mid X) = \sum_{x} H(D \mid x) Pr(x) \tag{2}$$

The conditional entropy is thus an *expectation* of what the entropy would be after observing X. A similar selection criterion is to observe variables that will most greatly increase the margin between the posterior probabilities of the first and second most-likely states of the decision variable (Krause & Guestrin, 2009).

These selection criteria involve utilizing the notion of *value of information (VOI)* in order to quantify the value of various observations (Lindley, 1956; Stratonovich, 1965; Howard, 1966; Raiffa, 1968). The VOI of a set of variables can depend on various measures. In the two example selection criteria we have discussed, those measures would be entropy and margins of confidence. For instance, if observing a variable X would reduce the conditional entropy $H(D \mid X)$ more than observing variable X' would ($H(D \mid X) < H(D \mid X')$), then the value of observing X would be higher.

Krause and Guestrin (2009) define a general notion of VOI that is based on different reward functions. In particular, given an arbitrary reward function R,[3] a hypothesis variable D, and evidence $\mathbf{e}$, the VOI of observing hidden variables $\mathbf{H}$ is:

$$\mathcal{V}(R, D, \mathbf{H}, \mathbf{e}) = \mathcal{ER}(R, D, \mathbf{H}, \mathbf{e}) - R(Pr(D \mid \mathbf{e})) \tag{3}$$

where

$$\mathcal{ER}(R, D, \mathbf{H}, \mathbf{e}) = \sum_{\mathbf{h}} R(Pr(D \mid \mathbf{h}, \mathbf{e})) Pr(\mathbf{h} \mid \mathbf{e}) \tag{4}$$

is the expected reward of observing variables $\mathbf{H}$ and $R(Pr(D \mid \mathbf{e}))$ is the reward had we not observed variables $\mathbf{H}$. By this definition, the reward function used by Lu and

2. In information theory, the logarithm is typically assumed to be base-2 (Cover & Thomas, 1991), which we also assume throughout the paper for convenience.
3. A reward function is assumed to take as input the probability distribution of the hypothesis variable, $Pr(D)$, and return some numeric value. We discuss reward functions further in Section 7.2.

D	$Pr(D \mid E_1 = +, E_2 = +)$
$+$	0.880952
$-$	0.119048

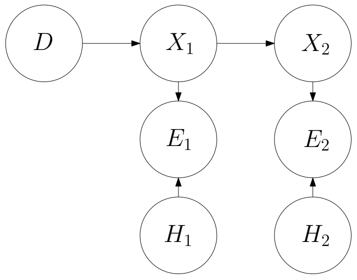

Figure 1: A simple Bayesian network, under sensor readings $\{E_1 = +, E_2 = +\}$. Variables H_1 and H_2 represent the health of sensors E_1 and E_2. On the left is the posterior on the decision variable D. Network CPTs can be found in Appendix C in Figure 13.

Przytula (2006) and Krause and Guestrin (2009) to select variables in order to minimize the conditional entropy is then $R(Pr(D \mid \mathbf{e})) = -H(D \mid \mathbf{e})$, so maximizing the expected reward of observing variables $\mathbf{H}$ is then equivalent to minimizing the conditional entropy $H(D \mid \mathbf{H})$. Some other possible reward functions involve utility-based reward functions or threshold-based reward functions (Munie & Shoham, 2008).[4]

Note that the vast majority of selection criteria use a myopic approach, in which out of all possible observations, just one observation is considered at a time, and the observation with the highest VOI is selected each time. This approach is greedy and short-sighted — the *optimal* VOI can only be computed by computing it non-myopically (Bilgic & Getoor, 2011). We discuss the usage of non-myopic VOI in Appendix A.1.

3. Same-Decision Probability

The *Same-Decision Probability* (SDP) was initially introduced by Darwiche and Choi (2010) as a confidence measure for threshold-based decisions in Bayesian networks under noisy sensor readings. Prior to formally defining the SDP, we first show an example to provide intuition. Consider now the Bayesian network in Figure 1, which models a scenario involving a hypothesis variable D, and two noisy sensors E_1 and E_2 that influence our belief in some hypothesis d. Networks such as this are typically used to compute the belief in the hypothesis given some sensor readings, $Pr(d \mid \mathbf{e})$. The basis of whether or not to make a decision often then depends on whether or not the posterior probability of that hypothesis d surpasses some threshold T (Hamscher, Console, & de Kleer, 1992; Heckerman et al., 1995; Kruegel et al., 2003; Lu & Przytula, 2006).

Figure 1 shows a particular reading of two sensors and the resulting belief $Pr(D = + \mid E_1 = +, E_2 = +)$. Suppose our threshold is $T = 0.6$, then as $Pr(d \mid \mathbf{e}) \geq T$, we would make a certain decision. Notice in Figure 1 that the *health* of those sensors is modeled by variables H_1 and H_2. A sensor can either be truthful, stuck positive (readings always display as $+$), or lying (readings show the opposite value of the actual value) (Darwiche & Choi, 2010). If those variables had been observed, they could have informed us of the trustworthiness of

4. For more on reward functions, see the list provided by Krause and Guestrin (2009).

H_1	H_2	$Pr(\mathbf{h} \mid \mathbf{e})$	$Pr(d \mid \mathbf{h}, \mathbf{e})$
t	t	0.781071	**0.90**
p	t	0.096429	**0.82**
l	t	0.001071	0.10
t	p	0.096429	**0.90**
p	p	0.021429	0.50
l	p	0.001190	0.10
t	l	0.001071	**0.90**
p	l	0.001190	0.18
l	l	0.000119	0.10

Table 1: Scenarios $\mathbf{h}$ for sensor readings $\mathbf{e} = \{E_1 = +, E_2 = +\}$ for the network in Figure 1, where $\mathbf{H} = \{H_1, H_2\}$. Cases above the threshold $T = 0.6$ are in bold. Note t, p, l respectively represent a truthful, stuck positive, and lying sensor.

the sensors E_1 and E_2 and thus allow us to make a better decision. We want to make a more-informed decision based on the probability $Pr(d \mid \mathbf{h}, \mathbf{e})$ instead of making a decision based on just $Pr(d \mid \mathbf{e})$.

Consider Table 1, which enumerates all of the possible health states of the sensors. In only four of these cases does the probability of the hypothesis pass the threshold (in bold), leading to the same decision. In the other five scenarios, a different decision would have been made. The SDP is thus the probability of the four scenarios in which the same decision would have been made. For this example, the SDP is:

$$0.781071 + 0.096429 + 0.096429 + 0.001071 = 0.975$$

indicating a relatively robust decision.

Choi et al. (2012) define the SDP formally as:

Definition 1 (*Same-Decision Probability*)**.** *Let $\mathcal{N}$ be a Bayesian network that is conditioned on evidence $\mathbf{e}$, where we are further given a hypothesis d, a threshold T, and a set of unobserved variables $\mathbf{H}$. Suppose we are making a decision that is confirmed by the threshold $Pr(d \mid \mathbf{e}) \geq T$. The* ***Same-Decision Probability*** *in this scenario is*

$$SDP(d, \mathbf{H}, \mathbf{e}, T) = \sum_{\mathbf{h}} [Pr(d \mid \mathbf{e}, \mathbf{h}) \geq T] Pr(\mathbf{h} \mid \mathbf{e}), \tag{5}$$

where $[Pr(d \mid \mathbf{h}, \mathbf{e}) \geq T]$ is an indicator function such that

$$[Pr(d \mid \mathbf{h}, \mathbf{e}) \geq T] = \begin{cases} 1 & \text{if } Pr(d \mid \mathbf{e}, \mathbf{h}) \geq T \\ 0 & \text{otherwise.} \end{cases}$$

The SDP is notably hard to compute. Choi et al. (2012) prove that computing the SDP is in general $\mathrm{PP}^{\mathrm{PP}}$-complete.[5] From previous work on the SDP (Darwiche & Choi, 2010; Choi et al., 2012), the two options for computing the SDP are

5. The class $\mathrm{PP}^{\mathrm{PP}}$ can be thought of as a counting variant of the $\mathrm{NP}^{\mathrm{PP}}$ class, which contains the polynomial time hierarchy PH and for which the MAP problem is complete (Park & Darwiche, 2004).

D	$Pr(D)$
$+$	0.5
$-$	0.5

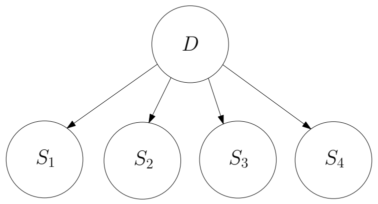

Figure 2: A Bayesian network for intrusion detection, with its CPTs given in Table 2

1. An *approximate* algorithm developed by Choi et al. (2012). This algorithm uses an augmented variable elimination algorithm that produces a potentially weak bound based on the one-sided Chebyshev inequality.

2. A *naive* brute-force method that enumerates over all possible instantiations.

4. Applying the Same-Decision Probability

We investigate the use of the SDP as a stopping criterion and a selection criterion. We contrast the usage of the SDP with traditional methods discussed in Section 2, and find that using the SDP can provide more insight to a decision maker in some scenarios.

4.1 SDP as a Stopping Criterion

By the definition of SDP, we can see that calculating a high SDP, in contrast to calculating a low SDP, would indicate a higher degree of readiness to make a decision, as the chances of the decision changing given further evidence gathering is lower. In this section we show that computing the SDP can provide additional insight and thus can distinguish scenarios that are otherwise indistinguishable based on standard stopping criteria.

The threshold-based decision is a classical notion in decision making under uncertainty, and it is commonly used as it requires no utilities to be elicited. Examples of threshold-based decisions are very prevalent in educational diagnosis (Gertner, Conati, & VanLehn, 1998; Conati, Gertner, & VanLehn, 2002; Butz et al., 2004; Xenos, 2004; Arroyo & Woolf, 2005; Munie & Shoham, 2008), intrusion detection (Kruegel et al., 2003; Modelo-Howard et al., 2008), fault diagnosis (Heckerman et al., 1995; Lu & Przytula, 2006), and medical diagnosis (Pauker & Kassirer, 1980; Kahn et al., 1997; van der Gaag & Coupé, 1999).

Consider the sensor network in Figure 2, which may correspond to an intrusion detection application as discussed by Kruegel et al. (2003). Here, the hypothesis variable is $D = \{+, -\}$ with $D = +$ implying an intrusion. Suppose we commit to a decision, and stop performing observations, when our belief in the event $D = +$ surpasses some threshold T, say $T = 0.55$. There are four sensors in this model, S_1, S_2, S_3 and S_4, whose readings may affect this decision.

Consider the two following scenarios:

1. $S_1 = +$ and $S_2 = +$.

2. $S_3 = +$ and $S_4 = +$.

D	S_1	$Pr(S_1 \mid D)$
$+$	$+$	0.55
$+$	$-$	0.45
$-$	$+$	0.45
$-$	$-$	0.55

D	S_3	$Pr(S_3 \mid D)$
$+$	$+$	0.60
$+$	$-$	0.40
$-$	$+$	0.40
$-$	$-$	0.60

D	S_2	$Pr(S_2 \mid D)$
$+$	$+$	0.55
$+$	$-$	0.45
$-$	$+$	0.45
$-$	$-$	0.55

D	S_4	$Pr(S_4 \mid D)$
$+$	$+$	0.65
$+$	$-$	0.35
$-$	$+$	0.35
$-$	$-$	0.65

Table 2: CPTs for the network in Figure 2. Parameterization 1.

Since $Pr(D = + \mid S_1 = +, S_2 = +) = 0.60 > 0.55$ and $Pr(D = + \mid S_3 = +, S_4 = +) = 0.74 > 0.55$, it is clear that in both cases that the threshold has been crossed. We deem that no further observations are necessary based on our beliefs surpassing our threshold. Hence, when using thresholds as a stopping criterion (as is commonly done, see Kruegel et al., 2003; Lu & Przytula, 2006; Gao & Koller, 2011), the two scenarios are identical in that no more information is gathered and a decision is made.

From the viewpoint of SDP, however, these two scenarios are very different. In particular, the first scenario leads to an SDP of 52.97%. This means that there is a 47.03% chance that a different decision would be made if we were to further observe the two unobserved sensors S_3 and S_4. The second scenario, however, leads to an SDP of 100%. That is, we would with certainty know that we would make the same decision if we were to also observe the two unobserved sensors S_1 and S_2: no matter what the readings of S_1 and S_2 could be, our beliefs in the event $D = +$ would always surpass our threshold 0.55. Indeed, as we can see in Table 2, the sensors S_1 and S_2 are not as strong as sensors S_3 and S_4, and in this example, they are not strong enough to reverse our decision.

This example provides a clear illustration of the utility of the SDP as a stopping criterion. However, some may argue that it is clear that in the second case, we should stop gathering information as $Pr(D = + \mid S_3 = +, S_4 = +) = 0.74$ has a larger margin from the threshold than $Pr(D = + \mid S_1 = +, S_2 = +) = 0.60$.[6] However, we show with the following example that deciding to stop based solely on the margin is not robust. Consider once again the sensor network in Figure 2 and the parameterizations of the sensor network shown in Table 5 and Table 6 (found in Appendix C), which we respectively refer to as Case 1 and Case 2.[7] Note that in this example, we use a threshold of $T = 0.5$.

In both cases, when $S_3 = +$ and $S_4 = +$ are observed, $Pr(D = + \mid S_3 = +, S_4 = +) \geq T$. In particular,

1. Case 1: $Pr(D = + \mid S_3 = +, S_4 = +) = 0.775$.

6. Thus using the aforementioned *margins of confidence* stopping criterion used by Gao and Koller (2011).
7. Note that the exact numbers of the CPTs are not necessary to grasp these examples — CPTs are provided so that readers may reconstruct these networks.

2. Case 2: $Pr(D = + \mid S_3 = +, S_4 = +) = 0.599$.

By using the previously discussed margin stopping criterion, it would seem that in Case 1 we could stop information gathering, whereas for Case 2 more information gathering is necessary. However, we can compute the SDP for both cases for more insights on the nature of robustness in these settings. For Case 1, we find that the SDP is 0.781, whereas for Case 2, we find that the SDP is 1.0 — even though in Case 1 the margin is higher, there is a greater chance that the decision would change given further information. This demonstrates that we cannot use solely the margin to determine whether or not to stop information gathering.

It is clear from these examples that SDP is a useful stopping criterion. First, the SDP can pinpoint situations where further observations are unnecessary as they would never reverse the decision under consideration. Second, the SDP can also identify situations where the decision to be made is not robust, and is likely to change upon making further observations. In addition to these examples, in Appendix A.2 we show how the SDP can be a useful stopping criterion in the context of utility-based decisions (e.g. influence diagrams).

4.2 SDP as a Selection Criterion

We now turn our attention to the use of SDP as a criterion for deciding which variables to observe next, assuming that some stopping criterion indicates that further observations are necessary. Our proposal is based on using VOI as the selection criterion (see Equation 3), while choosing the SDP as the reward function. We call this the *SDP gain*, and it is formally defined as:

Definition 2. *Given Definition 4 of an SDP, the* ***SDP gain*** *of observing variables* $\mathbf{G}$ *out of variables* $\mathbf{H}$ *is defined as the* ***expected SDP*** *of observing* $\mathbf{G} \subseteq \mathbf{H}$ *subtracted by the SDP over* $\mathbf{H}$*:*

$$\mathcal{G}(\mathbf{G}) = \mathcal{E}(\mathbf{G}, \mathbf{H}, \mathbf{e}, T) - SDP(d, \mathbf{H}, \mathbf{e}, T), \tag{6}$$

where the expected SDP is defined as:

$$\mathcal{E}(\mathbf{G}, \mathbf{H}, \mathbf{e}, T) = \sum_{\mathbf{g}} SDP(d, \mathbf{H} \setminus \mathbf{G}, \mathbf{g}\mathbf{e}, T) Pr(\mathbf{g}|\mathbf{e}) \tag{7}$$

and d is defined as the decision made given the current evidence.

Note that if we observe some variables $\mathbf{G} \subseteq \mathbf{H}$ such that the expected SDP is 1.0, that indicates that after observing $\mathbf{G}$ and making a decision, the remaining variables $\mathbf{H} \setminus \mathbf{G}$ will be rendered completely redundant — their observation will have no effect on the decision. The goal of using the SDP gain as a selection criterion is to observe those variables which, on average, will allow for the most stable decision given the collected observations.

We will next provide an example of using SDP as a selection criterion, contrasting it with two other selection criteria: One based on reducing entropy of the hypothesis variable D, and another based on maximizing the gap between the decision probability $Pr(d|\mathbf{e})$ and the given threshold T (Krause & Guestrin, 2009). While both criteria can be motivated as reducing uncertainty, we show that both can indeed lead to less stable decisions than if the SDP were to be used.

D	$Pr(D)$
$+$	0.5
$-$	0.5

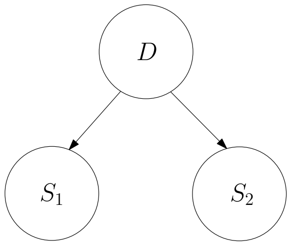

Figure 3: A Bayesian network with its CPTs given in Appendix C.

The example is given by the Bayesian network in Figure 3, where D is the hypothesis variable and S_1/S_2 are sensors. A decision is triggered when $Pr(D = + \mid \mathbf{e}) \geq .80$, where evidence $\mathbf{e}$ is over sensors S_1 and S_2. With no observations (empty evidence $\mathbf{e}$), the SDP is 0.595, suggesting that further observations may be needed. Assuming a limited number of observations (Heckerman et al., 1995), and using a myopic approach of observing one variable at a time (Dittmer & Jensen, 1997), we need now to select the next variable to observe.

Note that maximizing VOI with negative entropy as the reward function amounts to maximizing mutual information, as $H(D, X) = H(D) - H(D \mid X)$ (Cover & Thomas, 1991; Krause & Guestrin, 2005). The mutual information between variable D and sensor S_2 is 0.53 whereas the mutual information between D and sensor S_1 is 0.278. Hence, observing S_2 will reduce the entropy of D the most. In terms of margin of confidence, another reward function used by Krause and Guestrin (2009), observing S_2 will on average lead to a 0.7 margin between the states $D = +$ and $D = -$, whereas observing S_1 will only lead to a 0.6 margin between the two states.

However, if we compute the corresponding SDP gains, $\mathcal{G}(S_1)$ and $\mathcal{G}(S_2)$, we find that observing S_1 will, on average, lead to improving the decision stability the most. In particular, observing S_1 would give us an SDP of either 1 or 0.81 — for an expected SDP of 0.905, whereas observing S_2 would give us an SDP of either 0.7625, 0.5, or 1 — for an expected SDP of 0.805. Therefore, $\mathcal{G}(S_1) = 0.31$ and $\mathcal{G}(S_2) = 0.21$. Hence, observing S_1 will on average allow us to make a decision that is less likely to change due to additional information (beyond S_1).

Some intuition to why this occurs is that although observing S_2 leads to greater information gain than observing S_1, it is *superfluous information.* Note that $Pr(D = + \mid S_2 = -) = 0.0625$, whereas $Pr(D = + \mid S_1 = -) = 0.2$. Clearly, we can see that observing S_2 can lead to a more skewed distribution with minimal conditional entropy. However, in the context of threshold-based decisions, we make a decision based solely on whether $Pr(D = + \mid \mathbf{e})$ is above or below the threshold, meaning that we may not put as much emphasis on how much below or above the threshold $Pr(D = + \mid \mathbf{e})$ is. In this case, although observing S_2 can on average lead to a more extreme distribution, observing $S_2 = \mathsf{o}$ leads to making an extremely nonrobust decision (a decision that would change 50% of the time with observation of S_1). Observing S_1 before making a decision leads to a much more robust decision. This example demonstrates the usefulness of SDP as a selection criterion for threshold-based decisions, as the SDP can be used to select observations that lead to more robust decisions.

D	$Pr(D)$
$+$	0.3
$-$	0.7

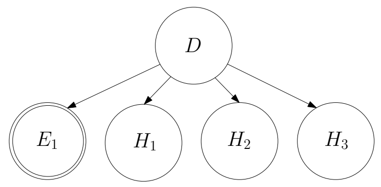

Figure 4: A Naive Bayes network (CPTs defined in Appendix C).

5. Computing the Same-Decision Probability

Computing the SDP involves computing an expectation over the hidden variables $\mathbf{H}$. The naive brute-force algorithm would enumerate and check whether $Pr(d \mid \mathbf{h}, \mathbf{e}) \geq T$ for all instantiations $\mathbf{h}$ of $\mathbf{H}$. We now present an algorithm that can save us the need to explore every possible instantiation of $\mathbf{h}$. To make the algorithm easier to understand, we will first describe how to compute the SDP in a Naive Bayes network. This is no trivial problem — we show in Section 7 that computing the SDP in a Naive Bayes network is NP-hard. We then generalize our algorithm to arbitrary networks.

5.1 Computing the SDP in Naive Bayes Networks

We will find it more convenient to implement the test $Pr(d \mid \mathbf{h}, \mathbf{e}) \geq T$ in the *log-odds* domain, where:

$$\log O(d \mid \mathbf{h}, \mathbf{e}) = \log \frac{Pr(d \mid \mathbf{h}, \mathbf{e})}{Pr(\overline{d} \mid \mathbf{h}, \mathbf{e})} \tag{8}$$

We then define the *log-odds threshold* as $\lambda = \log \frac{T}{1-T}$ and, equivalently, test whether $\log O(d \mid \mathbf{h}, \mathbf{e}) \geq \lambda$.

In a Naive Bayes network with D as the class variable, $\mathbf{H}$ and $\mathbf{E}$ as the leaf variables, and $\mathbf{Q} \subseteq \mathbf{H}$, the posterior log-odds after observing a *partial instantiation* $\mathbf{q} = \{h_1, \ldots, h_j\}$ can be written as:

$$\log O(d \mid \mathbf{q}, \mathbf{e}) = \log O(d \mid \mathbf{e}) + \sum_{i=1}^{j} w_{h_i} \tag{9}$$

where w_{h_i} is the *weight of evidence* h_i and defined as:

$$w_{h_i} = \log \frac{Pr(h_i \mid d, \mathbf{e})}{Pr(h_i \mid \overline{d}, \mathbf{e})} \tag{10}$$

The weight of evidence w_{h_i} is then the contribution of evidence h_i to the quantity $\log O(d \mid \mathbf{q}, \mathbf{e})$ (Chan & Darwiche, 2003). Note that all weights can be computed in time and space linear in $|\mathbf{H}|$ using a floating point representation.[8] Table 3 depicts the weights of evidence for the network in Figure 4.

8. Additionally, note that for Equation 10, since in Naive Bayes networks H_i is d-separated from $\mathbf{E}$ given d, the term $\mathbf{e}$ can be dropped from the equation. We leave the term in because for general networks, H_i may not be d-separated from $\mathbf{E}$.

i	w_{h_i}	$w_{\overline{h}_i}$
1	3.0	-2.17
2	1.22	-1.22
3	1.22	-1.22

Table 3: Weights of evidence for the attributes in Figure 4.

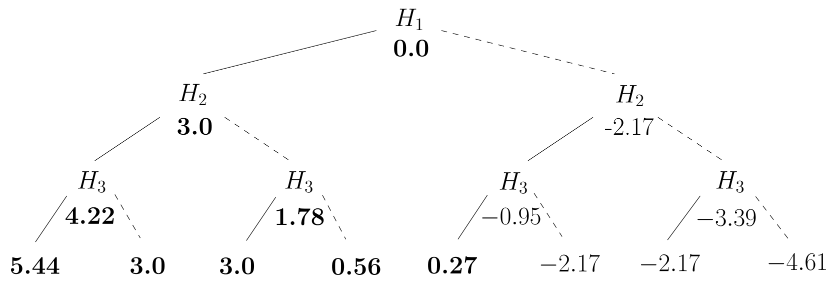

Figure 5: The search tree for the network of Figure 4. A solid line indicates + and a dashed line indicates −. The quantity $\log O(d \mid \mathbf{q}, \mathbf{e})$ is displayed next to each node $\mathbf{q}$ in the tree. Nodes with $\log O(d \mid \mathbf{q}, \mathbf{e}) \geq \lambda = 0$ are shown in bold.

One can then compute the SDP by enumerating the instantiations of variables $\mathbf{H}$ and then using Equation 9 to test whether $\log O(d \mid \mathbf{h}, \mathbf{e}) \geq \lambda$. Figure 5 depicts a search tree for the Naive Bayes network in Figure 4, which can be used for this purpose. The leaves of this tree correspond to instantiations $\mathbf{h}$ of variables $\mathbf{H}$. More generally, every node in the tree corresponds to an instantiation $\mathbf{q}$, where $\mathbf{Q} \subseteq \mathbf{H}$.

A brute-force computation of the SDP would then entail:

1. Initializing the total SDP to 0.
2. Visiting every leaf node $\mathbf{h}$ in the search tree.
3. Checking whether $\log O(d \mid \mathbf{h}, \mathbf{e}) \geq \lambda$ and if so, adding $Pr(\mathbf{h} \mid \mathbf{e})$ to the total SDP.

Figure 5 depicts the quantity $\log O(d \mid \mathbf{q}, \mathbf{e})$ for each node $\mathbf{q}$ in the tree, indicating that five leaf nodes (i.e., five instantiations of variables $\mathbf{H}$) will indeed contribute to the SDP.

We now state the key observation underlying our proposed algorithm. Consider the node corresponding to instantiation $H_1 = +$ in the search tree, with $\log O(d \mid H_1 = +, \mathbf{e}) = 3.0$. All four completions $\mathbf{h}$ of this instantiation (i.e., the four leaf nodes below it) are such that $\log O(d \mid \mathbf{h}, \mathbf{e}) \geq \lambda = 0$. Hence, we really do not need to visit all such leaves and add their contributions $Pr(\mathbf{h}|\mathbf{e})$ individually to the SDP. Instead, we can simply add $Pr(H_1 = +|\mathbf{e})$ to the SDP, which equals the sum of $Pr(\mathbf{h}|\mathbf{e})$ for these leaves. More importantly, we can detect that all such leaves will contribute to the SDP by computing a lower bound using the weights depicted in Table 3. That is, there are two weights for variable H_2, the *minimum* of which is -1.22. Moreover, there are two weights for variable H_3, the *minimum* of which -1.22. Hence, the lowest contribution to the log-odds made by any leaf below node $H_1 = +$

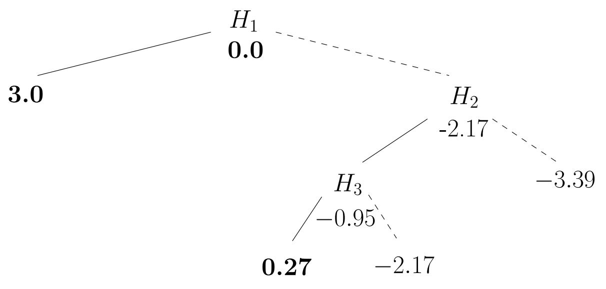

Figure 6: The reduced search tree for the network of Figure 5.

will be $-1.22 - 1.22 = -2.44$. Adding this contribution to the current log-odds of 3.0 will lead to a log-odds of .56, which still passes the given threshold.

A similar technique can be used to compute upper bounds, allowing us to detect nodes in the search tree where no leaf below them will contribute to the SDP. Consider for example the node corresponding to instantiation $H_1 = -, H_2 = -$, with $\log O(d \mid H_1 = -, H_2 = -, \mathbf{e}) = -3.39$. Neither of the leaves below this node will contribute to the SDP as their log-odds do not pass the threshold. This can be detected by considering the weights of evidence for variable H_3 and computing the *maximum* of these weights (1.22). Adding this to the current log-odds of -3.39 gives -2.17, which is still below the threshold. Hence, no leaf node below $H_1 = -, H_2 = -$ will contribute to the SDP and this part of the search tree can also be pruned.

If we apply this pruning technique based on lower and upper bounds, we will actually end up exploring only the portion of the tree shown in Figure 6. The pseudocode of our final algorithm is shown in Algorithm 1. Note that it takes linear time to compute the upper and lower bounds. Additionally, note that the specific ordering of $\mathbf{H}$ in which the search tree is constructed is directly linked to the amount of pruning. We use an ordering heuristic that ranks each query variable H_i by the difference of its corresponding upper and lower bound — $\mathbf{H}$ is then ordered from greatest difference to lowest difference as to allow for earlier pruning.

5.2 Computing the SDP in Arbitrary Networks

We will generalize our algorithm to arbitrary networks by viewing such networks as Naive Bayes networks but with aggregate attributes. For this, we first need the following notion.

Definition 3. *A partition of* $\mathbf{H}$ *given* D *and* $\mathbf{E}$ *is a set* $\mathbf{S}_1, \ldots, \mathbf{S}_k$ *such that:* $\mathbf{S}_i \subseteq \mathbf{H}$*;* $\mathbf{S}_i \cap \mathbf{S}_j = \emptyset$*;* $\mathbf{S}_1 \cup \ldots \cup \mathbf{S}_k = \mathbf{H}$*; and* $\mathbf{S}_i$ *is independent (d-separated) from* $\mathbf{S}_j$*,* $i \neq j$*, given* D *and* $\mathbf{E}$*.*

Figure 7 depicts an example partition.

The intuition behind a partition is that it allows us to view an arbitrary network as a Naive Bayes network, with class variable D and aggregate attributes $\mathbf{S}_1, \ldots, \mathbf{S}_k$. That is, each aggregate attribute $\mathbf{S}_i$ is viewed as a variable with states $\mathbf{s}_i$, allowing us to view each instantiation $\mathbf{h}$ as a set of values $\mathbf{s}_1, \ldots, \mathbf{s}_k$. We now have:

Algorithm 1 Computing the SDP in a Naive Bayes network. Note: For $\mathbf{q} = \{h_1, \ldots, h_j\}$, $w_{\mathbf{q}}$ is defined as $\sum_{i=1}^{j} w_{h_i}$.

```
input:
 $\mathcal{N}$:  Naive Bayes network with class variable $D$
 $\mathbf{H}$:  attributes $\{H_1, \ldots, H_k\}$
 $\lambda$:  log-odds threshold
 $\mathbf{e}$:  evidence
output: Same-Decision Probability $p$
main:
  global $p \leftarrow 0.0$ (initial probability)
  $\mathbf{q} \leftarrow \{\}$ (initial instantiation is empty set)
  $depth \leftarrow 0$ (initial depth of search tree)
  DFS_SDP($\mathbf{q}$, $\mathbf{H}$, $depth$)
  return $p$
1: procedure DFS_SDP($\mathbf{q}$, $\mathbf{H}$, $depth$)
2:     $UpperBound \leftarrow \log O(d \mid \mathbf{e}) + w_{\mathbf{q}} + \sum_{i=depth+1}^{k} \max_{h_i} w_{h_i}$
3:     $LowerBound \leftarrow \log O(d \mid \mathbf{e}) + w_{\mathbf{q}} + \sum_{i=depth+1}^{k} \min_{h_i} w_{h_i}$
4:     if ($UpperBound < \lambda$) then return
5:     else if ($LowerBound \geq \lambda$) then
6:         add $Pr(\mathbf{q} \mid \mathbf{e})$ to $p$, return
7:     else
8:         if $depth < k$ then
9:             for each value $h_{depth+1}$ of attribute $H_{depth+1}$ do
10:                DFS_SDP($\mathbf{q}h_{depth+1}$, $\mathbf{H} \setminus H_{depth+1}$, $depth + 1$)
```

Proposition 1. *For a partial instantiation* $\mathbf{q} = \{\mathbf{s}_1, \ldots, \mathbf{s}_j\}$,

$$\log O(d \mid \mathbf{q}, \mathbf{e}) = \log O(d \mid \mathbf{e}) + \sum_{i=1}^{j} w_{\mathbf{s}_i}, \tag{11}$$

where

$$w_{\mathbf{s}_i} = \log \frac{Pr(\mathbf{s}_i, \mid d, \mathbf{e})}{Pr(\mathbf{s}_i \mid \overline{d}, \mathbf{e})} \tag{12}$$

Proof.

$$\begin{aligned}
\log O(d \mid \mathbf{q}, \mathbf{e}) &= \log \frac{Pr(d \mid \mathbf{q}, \mathbf{e})}{Pr(\overline{d} \mid \mathbf{q}, \mathbf{e})} \\
&= \log \frac{Pr(d \mid \mathbf{e}) Pr(\mathbf{s}_1 \mid d, \mathbf{e}) \ldots Pr(\mathbf{s}_j \mid d, \mathbf{e})}{Pr(\overline{d} \mid \mathbf{e}) Pr(\mathbf{s}_1 \mid \overline{d}, \mathbf{e}) \ldots Pr(\mathbf{s}_j \mid \overline{d}, \mathbf{e})} \\
&= \log O(d \mid \mathbf{e}) + \sum_{i=1}^{j} w_{\mathbf{s}_i}
\end{aligned}$$

□

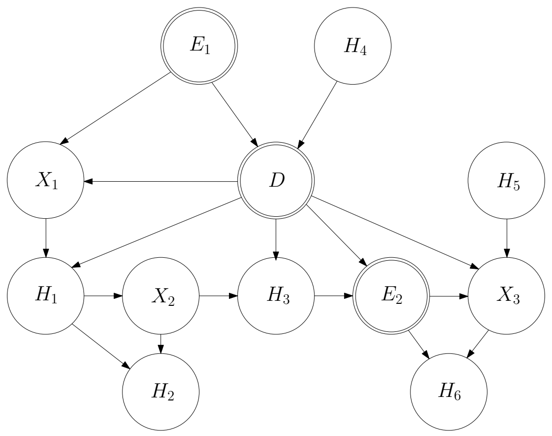

Figure 7: The partition of $\mathbf{H}$ given D and $\mathbf{E}$ is: $\mathbf{S}_1 = \{H_1, H_2, H_3\}$ $\mathbf{S}_2 = \{H_4\}$, $\mathbf{S}_3 = \{H_5, H_6\}$.

Since Equations 11 and 12 are analogous to Equations 9 and 10, we can now use Algorithm 1 on an arbitrary network. This usage, however, requires some auxiliary computations that were not needed or were readily available for Naive Bayes networks. We discuss these computations next.

5.2.1 Finding a Partition

We first need to compute a partition $\mathbf{S}_1, \ldots, \mathbf{S}_k$, which is done by pruning the network structure as follows: we delete edges outgoing from nodes in evidence $\mathbf{E}$ and hypothesis D, and delete (successively) all leaf nodes that are neither in $\mathbf{H}$, $\mathbf{E}$ or D. We then identify the components $\mathbf{X}_1, \ldots, \mathbf{X}_k$ of the resulting network and define each non-empty $\mathbf{S}_i = \mathbf{H} \cap \mathbf{X}_i$ as an element of the partition. This guarantees that in the original network structure, $\mathbf{S}_i$ is d-separated from $\mathbf{S}_j$ by D and $\mathbf{E}$ for $i \neq j$ (see (Darwiche, 2009)). In Figure 7, network pruning leads to the components $\mathbf{X}_1 = \{X_1, X_2, E_2, H_1, H_2, H_3\}$, $\mathbf{X}_2 = \{D, E_1, H_4\}$ and $\mathbf{X}_3 = \{X_3, H_5, H_6\}$.

5.2.2 Computing Posterior Log-Odds, Probability and Weights of Evidence

The quantities $O(d \mid \mathbf{e})$, $Pr(\mathbf{q} \mid \mathbf{e})$ and $w_{\mathbf{s}_i}$, which are referenced on Lines 2, 3, and 6 of the algorithm, have simple closed forms in Naive Bayes networks. For arbitrary networks, however, computing these quantities requires inference which we do using the algorithm of variable elimination as described by Darwiche (2009). Note that the network pruning of deleting edges and removing the leaf nodes, as discussed above, guarantees that each factor used by variable elimination will have all its variables in some component $\mathbf{X}_i$. Hence, variable elimination can be applied to each component $\mathbf{X}_i$ in isolation, which is sufficient to obtain all needed quantities.

5.2.3 Computing the Min and Max of Evidence Weights

We finally show how to compute the upper and lower bounds, $\max_{\mathbf{s}_i} w_{\mathbf{s}_i}$ and $\min_{\mathbf{s}_i} w_{\mathbf{s}_i}$, which are referenced on Lines 2 and 3 of the algorithm. These quantities can also be computed using variable elimination, applied to each component $\mathbf{X}_i$ in isolation. In this case, however, we must eliminate variables $\mathbf{X}_i \setminus \mathbf{S}_i$ first and then variables $\mathbf{S}_i$. Moreover, the first set of variables is summed-out, while the second set of variables is max'd-out or min'd-out, depending on whether we need $\max_{\mathbf{s}_i} w_{\mathbf{s}_i}$ or $\min_{\mathbf{s}_i} w_{\mathbf{s}_i}$. Finally, this elimination process is applied twice, once with evidence $d, \mathbf{e}$ and a second time with evidence $\overline{d}, \mathbf{e}$.

More precisely, for every component $\mathbf{X}_i$ we have a set of factors for the case where $D = d$ and where $D = \overline{d}$. Using the same variable ordering, we perform variable elimination on both sets of factors to eliminate any nonquery (intermediary) variables so that we are left with a set of factors ψ^i_d where $\prod \psi^i_d = Pr(\mathbf{S}_i, d, \mathbf{e})$, and a set of factors $\phi^i_{\overline{d}}$ where $\prod \phi^i_{\overline{d}} = Pr(\mathbf{S}_i, \overline{d}, \mathbf{e})$. Since the elimination order was the same, there is thus a one-to-one matching between factors from both sets, and we can define a new set of factors $\chi_i = \frac{\psi^i_d}{\phi^i_{\overline{d}}} = \frac{Pr^i_d(\mathbf{e}^i,\mathbf{S}_i)}{Pr^i_{\overline{d}}(\mathbf{e}^i,\mathbf{S}_i)}$. We can then calculate $w_{\overline{\mathbf{s}}_i}$ and $w_{\underline{\mathbf{s}}_i}$ by respectively maximizing and minimizing out variables. Note that summing out variables and then maximizing variables is the variable elimination algorithm used by Dechter (1999) in order to solve MAP. Our algorithm here differs as we perform both maximization and minimization (to both calculate $w_{\overline{\mathbf{s}}_i}$ and $w_{\underline{\mathbf{s}}_i}$), and do so on the set of factors χ_i instead of on the factors (ψ^i_d or $\phi^i_{\overline{d}}$) that result from simply summing out the intermediary variables.

Note that similarly to Dechter (1999), as we are first summing out variables and then performing some maximization (and minimization in our case), the elimination order in this case is constrained, meaning that we may be forced to use a poor ordering for variable elimination that results in a high treewidth.

5.3 Complexity Analysis

Let n be the number of variables in the network, $h = |\mathbf{H}|$, and $w = \max_i w_i$, where w_i is the width of constrained elimination order used on component $\mathbf{X}_i$. The best-case time complexity of our algorithm is then $\mathrm{O}\big(n \exp w\big)$ and the worst-case time complexity is $\mathrm{O}\big(n \exp{(w + h)}\big)$. The intuition behind these bounds is that computing the maximum and minimum weights for each aggregate attribute takes time $\mathrm{O}\big(n \exp w\big)$. This also bounds the complexity of computing $O(d|\mathbf{e})$, $Pr(\mathbf{q}|\mathbf{e})$ and corresponding weights $w_{\mathbf{s}_i}$. Moreover, depending on the weights and the threshold T, traversing the search tree can take anywhere from constant time to $\mathrm{O}\big(\exp h\big)$. Since depth-first search can be implemented with linear space, the space complexity is $\mathrm{O}\big(n \exp w\big)$.

6. Experimental Results

We performed several experiments on both real and synthetic networks to test the performance of our algorithm across a wide variety of network structures, ranging from simple Naive Bayes networks to highly connected networks. Real networks were either learned from datasets provided by the UCI Machine Learning Repository (Bache & Lichman, 2013) or

Network	source	$\|\mathbf{H}\|$	$\|\mathbf{h}\|$	*naive*	*approx*	*new*
car	UCI	6	144	0.131	0.118	0.049
emdec6g	HRL	8	256	0.407	0.245	0.294
tcc4e	HRL	9	512	0.470	0.257	0.149
ttt	UCI	9	19683	6.234	0.133	0.091
caa	CRESST	14	16384	6.801	0.145	0.167
voting	UCI	16	65536	21.35	0.176	0.128
nav	CRESST	20	1572864	642.88	0.856	0.178
fire	CRESST	24	16777216	ϕ	0.183	0.508
chess	UCI	30	1610612736	ϕ	*	15.53

Table 4: Algorithm comparison on real networks. We show the time, in seconds, it takes each algorithm, *naive*, *approx*, and *new* to compute the SDP in different networks. Note that ϕ indicates that the computation did not complete in the 20 minute time limit that we constrained. Moreover, * indicates that there was not sufficient memory to complete the computation.

provided by HRL Laboratories and CRESST.[9] The majority of the real networks used were diagnostic networks, which made it clear which variable should be selected as the decision variable as it would either be the "knowledge" or "fault" variable. For the unclear cases, the decision variable was picked at random. Both query and evidence variables were selected at random for all real networks.

Besides this algorithm, there are two other options available to compute the SDP: 1. the *naive* method to brute-force the computation by enumerating over all possible instantiations or 2. the *approximate* algorithm developed by Choi et al. (2012). To compare our algorithm with these two other approaches, we compute the SDP over the real networks. For each network we selected at least 80% of the total network variables to be query variables so that we could emphasize how the size of the query set greatly influences the computation time. Each computation was given 20 minutes to complete. As we believe that the value of the threshold can greatly affect running time, we computed the SDP with thresholds $T = [0.01, 0.1, 0.2, \ldots, 0.8, 0.9, 0.99]$ and took the worst-case time. The results of our experiments with the three algorithms are shown in Table 4. Note that $|\mathbf{H}|$ is the number of query variables and $|\mathbf{h}|$ is the number of instantiations the naive algorithm must enumerate over. Moreover, ϕ indicates that the computation did not complete in the 20 minute time limit and * indicates that there was not sufficient memory to complete the computation. The networks $\{$*car,ttt,voting,nav,chess*$\}$ are Naive Bayes networks whereas the networks $\{$*caa,fire*$\}$ are polytree networks and the others are more general networks.

Given the real networks that we tested our algorithm on, it is clear that the algorithm outperforms both the naive implementation and the approximate algorithm for both Naive Bayes networks and polytree networks. Note that the approximation algorithm is based on variable elimination but can only use certain constrained orders. For a Naive Bayes

9. `http://www.cse.ucla.edu/`

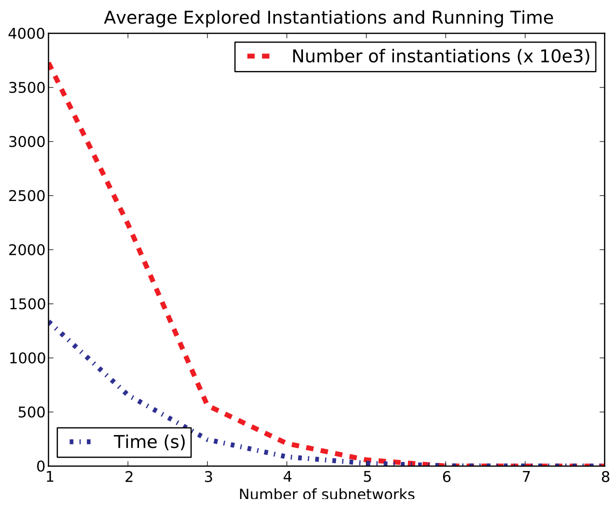

Figure 8: Synthetic network average running time and average number of instantiations explored by number of connected components.

network with hypothesis D being the root, the approximation algorithm will be forced to use a particularly poor ordering, which explains its failure on the `chess` network.

To analyze how a more general network structure and the selected threshold affects the performance of our algorithm, we generated synthetic networks with 100 variables and varying treewidth using *BNGenerator* (Ide, Cozman, & Ramos, 2004). For each network, we randomly selected the decision variable, 25 query variables, and evidence variables.[10] We then generated a partition for each network and grouped the networks by the size of obtained partition (k). Our goal was to test how our algorithm's running time and ability to prune the search-space depends on k. The average time and average number of instantiations explored are shown in Figure 8.

In general, we can see that as k increases, the number of instantiations explored by the algorithm decreases and its runtime improves. The network becomes more similar to a Naive Bayes structure with increasing k. Moreover, the larger k is, the more levels there are in the search tree, which means that our algorithm will have more opportunities to prune. In the worst case, a network may be unable to be disconnected at all ($k = 1$). However, even in this case our algorithm is still, on average, more efficient compared to the brute-force implementation — for some cases, after computing the maximum and minimum weight of observing $\mathbf{H}$, it will find that there does not exist any $\mathbf{h}$ that will change the decision. We found that, given a time limit of 2 hours, the brute-force algorithm could not solve any synthetic networks, whereas our approach solved more than 70% of such networks.

We also test how the threshold affects computation time. Here, we calculate the posterior probability of the decision variable and then run repeatedly our algorithm with thresholds that are varying increments away. The average running time for all increments can be seen in Figure 9. It is evident that when the threshold is set to be further away from the initial

10. As the synthetic networks are binary, a brute-force approach would need to explore 2^{25} instantiations.

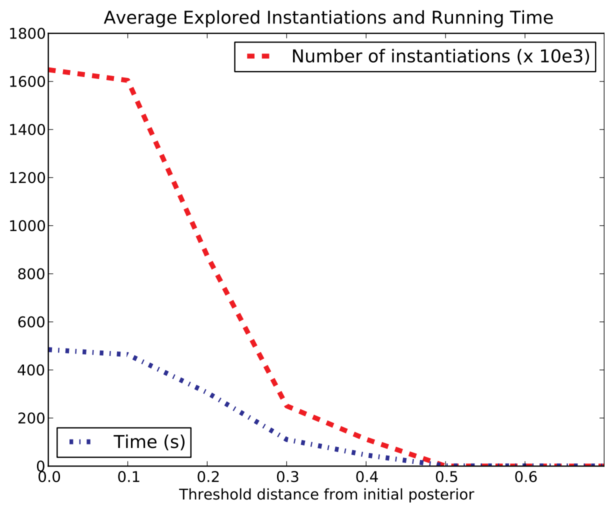

Figure 9: Synthetic network average running time and average number of instantiations explored by threshold distance from the initial posterior probability.

posterior probability, the algorithm finishes much faster, which is perhaps expected since the usage of more extreme thresholds would allow for more search space pruning.

Overall, our experimental results show that our algorithm is able to solve many SDP problems that are out of reach of existing methods. We also confirm that our algorithm completes much faster when the network can be disconnected or when the threshold is far away from the initial posterior probability of the decision variable.

7. The Complexity of Computing the Same-Decision Probability

We present here new complexity results for the SDP. We first prove that the complexity of computing the SDP in Naive Bayes structures is NP-hard. We then show that the general complexity of computing the SDP lies in the same complexity class as a general expectation computation problem that is applicable to a wide variety of queries in graphical models, such as the computation of non-myopic value of information.

7.1 Computing the SDP in Naive Bayes

SDP is known to be PP^{PP}-complete (Choi et al., 2012). We now show that SDP remains hard for Naive Bayes networks.

Theorem 1. *Computing the Same-Decision Probability in a Naive Bayes network is* NP-*hard.*

Proof. We reduce the number partition problem defined by Karp (1972) to computing the SDP in a Naive Bayes model. Suppose we are given a set of positive integers $c_1, \ldots, c_n$, and we wish to determine whether there exists $I \subseteq \{1, \ldots, n\}$ such that $\sum_{j \in I} c_i = \sum_{j \notin I} c_j$. We can solve this by considering a Naive Bayes network with a binary class variable D having uniform probability, and binary attributes $H_1, \ldots, H_n$ having CPTs leading to weights of evidence $w_{H_i=T} = c_i$ and $w_{H_i=F} = -c_i$. The construction of these CPTs can be done by solving the following system of equations:

$$\begin{aligned} c_i &= \log \frac{Pr(H_i = T \mid D = T)}{Pr(H_i = T \mid D = F)} \\ -c_i &= \log \frac{Pr(H_i = F \mid D = T)}{Pr(H_i = F \mid D = F)} \\ 1 &= Pr(H_i = T \mid D = T) + Pr(H_i = F \mid D = T) \\ 1 &= Pr(H_i = T \mid D = F) + Pr(H_i = F \mid D = F) \end{aligned}$$

We leave the exact derivations out (see Exercise 3.27 in Darwiche, 2009). We get the result that:

$$\begin{aligned} Pr(H_i = T \mid D = F) = Pr(H_i = F \mid D = T) &= \frac{1}{2^{c_i} + 1} \\ Pr(H_i = T \mid D = T) = Pr(H_i = F \mid D = F) &= 1 - \frac{1}{2^{c_i} + 1} \end{aligned}$$

Note that given that these CPTs have been defined such that $w_{H_i=T} = c_i$ and $w_{H_i=F} = -c_i$, the set of integers can be partitioned if there is an instantiation $\mathbf{h} = \{h_1, \ldots, h_n\}$ with $\sum_{i=1}^n w_{h_i} = 0$ since I would then include all indices i with $h_i = T$ in this case.

The Naive Bayes network satisfies a number of properties that we shall use next. First, $\sum_{i=1}^n w_{h_i}$ is either 0, ≥ 1, or ≤ -1 since all weights w_{h_i} are integers. Next, if $\sum_{i=1}^n w_{h_i} = c$, then $\sum_{i=1}^n w_{h'_i} = -c$ where $h'_i \neq h_i$. Finally, $Pr(h_1, \ldots, h_n) = Pr(h'_1, \ldots, h'_n)$ when $h'_i \neq h_i$, as D has a uniform probability distribution and each leaf H_i has been defined with a symmetric CPT.

Consider now the following SDP (the last step below is based on the above properties):

$$\begin{aligned} & SDP(D = T, \{H_1, \ldots, H_n\}, \{\}, 2/3) \\ &= \sum_{h_1,\ldots,h_n} [Pr(D = T \mid h_1, \ldots, h_n) \geq 2/3] Pr(h_1, \ldots, h_n) \\ &= \sum_{h_1,\ldots,h_n} [\log O(D = T \mid h_1, \ldots, h_n) \geq 1] Pr(h_1, \ldots, h_n) \\ &= \sum_{h_1,\ldots,h_n} \left[\sum_{i=1}^n w_{h_i} \geq 1 \right] Pr(h_1, \ldots, h_n) \\ &= \frac{1}{2} \sum_{h_1,\ldots,h_n} \left[\sum_{i=1}^n w_{h_i} \neq 0 \right] Pr(h_1, \ldots, h_n) \end{aligned}$$

We then have $\sum_{i=1}^n w_{h_i} = 0$ for some instantiation $h_1, \ldots, h_n$ iff

$$\sum_{h_1,\ldots,h_n} \left[\sum_{i=1}^n w_{h_i} \neq 0 \right] Pr(h_1, \ldots, h_n) < 1$$

Hence, the partitioning problem can be solved iff

$$SDP(D = T, \{H_1, \ldots, H_n\}, \{\}, 2/3) < 1/2$$

$\square$

7.2 The Complexity of Computing the Non-myopic VOI

The SDP was shown to be a $\mathrm{PP}^{\mathrm{PP}}$-complete problem by Choi et al. (2012). The class $\mathrm{PP}^{\mathrm{PP}}$ is essentially the counting variant of the $\mathrm{NP}^{\mathrm{PP}}$ class, which contains the polynomial hierarchy PH and for which the MAP problem is complete (Park & Darwiche, 2004). We show in this section that a general problem of computing expectations is also $\mathrm{PP}^{\mathrm{PP}}$-complete, with the non-myopic VOI and SDP being an instance of such an expectation. Thus, the development of algorithms to compute the SDP will be beneficial to problems in the $\mathrm{PP}^{\mathrm{PP}}$ class, which in turn benefits computing an assortment of expectations, including non-myopic VOI.

The proposed expectation computation is based on using a reward function R with some properties that we review next. In particular, the function R is assumed to map a probability distribution $Pr(D \mid \mathbf{e})$ to a numeric value. We also assume that the minimum l and maximum u of this range are polytime computable. These assumptions are not too limiting—for example, both entropy and utility can be expressed using reward functions that fall in this category (Krause & Guestrin, 2009).

We now consider the following computation of expectations.

D-EPT: Given a polynomial-time computable reward function R, hypothesis variable D, unobserved variables $\mathbf{H}$, evidence $\mathbf{e}$, a real number N, and a distribution Pr induced by a Bayesian network over variables $\mathbf{X}$,[11] the **expectation decision problem** asks: Is

$$\mathbb{E} = \sum_{\mathbf{h}} R(Pr(D \mid \mathbf{h}, \mathbf{e}))Pr(\mathbf{h} \mid \mathbf{e})$$

greater than N?

Note that the SDP falls as a special case when the reward function R is the SDP indicator function (see Definition 4). For example, in the definition used by Choi et al. (2012), the decision function outputs one of two decisions depending on whether $Pr(d|\mathbf{e}) > T$ for some value d of D and some threshold T.

We now have the following theorems, with proofs in Appendix B.

Theorem 2. ***D-EPT*** *is* $\mathrm{PP}^{\mathrm{PP}}$*-hard.*

Theorem 3. ***D-EPT*** *is in* $\mathrm{PP}^{\mathrm{PP}}$*.*

This shows that **D-EPT** is $\mathrm{PP}^{\mathrm{PP}}$-complete and implies that computational problems such as computing the non-myopic VOI using a variety of reward functions is also $\mathrm{PP}^{\mathrm{PP}}$-complete.

11. This proof also holds for influence diagrams constrained to have only one decision node.

8. Conclusion

In this paper, we have discussed some commonly used information gathering criteria for graphical models such as value of information and have reviewed the recently introduced notion of the Same-Decision Probability (SDP). In this paper, we have proposed the usage of the SDP as a decision making tool by showing concrete examples of its usefulness as both a stopping criterion and a selection criterion. As a stopping criterion, the SDP can allow us to determine when no further observations are necessary. As a selection criterion, usage of the SDP can allow us to select observations that allow us to increase decision robustness.

As we have justified the usage of the SDP, we have proposed an exact algorithm for its computation. Experimental results show that this algorithm has comparable running time to the previous approximate algorithm and is also much faster than the naive brute-force algorithm. Finally, we have presented several new complexity results.

Acknowledgements

This paper combines and extends the work presented by Chen, Choi, and Darwiche (2012b, 2013). This work has been partially supported by ONR grant #N00014-12-1-0423, NSF grant #IIS-1118122, and NSF grant #IIS-0916161. We would also like to thank the National Center for Research on Evaluation, Standards, & Student Testing and Hughes Research Lab for contributing sample diagnostic networks.

Appendix A. Miscellaneous Topics

In this section we go into more details about notions that were mentioned earlier in the paper. In particular, we continue our discussion from Section 2.3 and go over the notion of non-myopic value of information. Additionally, we also continue from where we left off in Section 4.1 and expand upon the notion of SDP as a stopping criterion in the context of utility-based decisions.

Appendix A.1 Non-myopic Value of Information

Myopic value of information is often used in many applications as it is easy to compute (Dittmer & Jensen, 1997; Vomlel, 2004; Gao & Koller, 2011). However, the problem with myopic selection is that it is not optimal, as at times the "whole is greater than the sum of its parts", as each individual observation in a set $\mathbf{H}$ seemingly may not provide significant value, but the VOI of observing $\mathbf{H}$ can be very high. For instance, take the function $D = X_1 \oplus X_2$, where alone neither X_1 nor X_2 is useful, but together they are determinative of D (Bilgic & Getoor, 2011). Only by computing the non-myopic VOI can the the optimal VOI be obtained.

Due to the aforementioned problems with using myopic VOI, more recently, researchers have recently suggested using the non-myopic VOI instead of myopic VOI and have proposed various methods to compute the non-myopic VOI (Heckerman, Horvitz, & Middleton, 1993; Liao & Ji, 2008; Krause & Guestrin, 2009; Zhang & Ji, 2010; Bilgic & Getoor, 2011). Computing the non-myopic VOI of some hidden variables $\mathbf{H}$ is difficult as it involves computing an expectation over the possible values of $\mathbf{H}$, which quickly becomes intractable as $\mathbf{H}$ becomes larger.

Existing algorithms for computing the non-myopic VOI are approximate algorithms (Heckerman et al., 1993; Liao & Ji, 2008) or relatively limited algorithms that are restricted to tree networks with few leaf variables (Krause & Guestrin, 2009). Bilgic and Getoor (2011) have developed the Value of Information Lattice (VOILA), a framework in which **all subsets** of hidden variables $\mathbf{H}$ are examined, and the optimal subset of features can be found to increase classification accuracy while meeting some budget constraint.

Appendix A.2 SDP as a Stopping Criterion for Utility-based Decisions

In some cases the expected *utility* of different decisions, as well as the cost of reducing uncertainty (making observations), is quantified. This is common in the decision-theoretic setting (Howard, 1966; Howard & Matheson, 1984), where *influence diagrams* are commonly used. Influence diagrams can be seen as Bayesian networks that incorporate decision and utility nodes (Howard & Matheson, 1984; Zhang, 1998; Kjærulff & Madsen, 2008). The selection criterion for the decision-theoretic setting is clear: the observations that lead to the greatest increase of expected utility are selected. The usage of utilities and observation costs is now prevalent; however, numerous researchers have noted the difficulty of coming up with the actual numerical quantities (Glasziou & Hilden, 1989; Lu & Przytula, 2006; Bilgic & Getoor, 2011).

To show how the SDP can be used as a stopping criterion in the decision-theoretic context of expected-utility decisions and influence diagrams (Howard & Matheson, 1984), we extend the definition of the SDP to a more general setting to allow for more applications. In particular, we assume that $\mathcal{F}$ is a polytime computable decision function that outputs a decision d based on the distribution $Pr(D \mid \mathbf{e})$. For instance, the decision function most commonly used in classification is to select the class with the highest posterior probability $\arg\max_d Pr(d \mid \mathbf{e})$ (Friedman et al., 1997), whereas for threshold-based decisions, the decision function would simply be to select a decision d if $Pr(D = d \mid \mathbf{e}) \geq T$.

SDP is thus defined as the probability that the same decision would be made if the hidden states of variables $\mathbf{H}$ were known (Chen et al., 2012b).

Definition 4 (*Same-Decision Probability — Generalized*)**.** *Given a decision function $\mathcal{F}$, hypothesis variable D, unobserved variables $\mathbf{H}$, and evidence $\mathbf{e}$, the* ***Same-Decision Probability*** *(SDP) is defined as*

$$SDP(\mathcal{F}, D, \mathbf{H}, \mathbf{e}) = \sum_{\mathbf{h}} [\mathcal{F}(Pr(D \mid \mathbf{h}, \mathbf{e}))]_{\mathbf{h}} Pr(\mathbf{h} \mid \mathbf{e}) \tag{13}$$

where $[\mathcal{F}(Pr(D \mid \mathbf{h}, \mathbf{e}))]_{\mathbf{h}}$ is an indicator function that

$$= \begin{cases} 1 & \text{if } \mathcal{F}(Pr(D \mid \mathbf{h}, \mathbf{e})) = \mathcal{F}(Pr(D \mid \mathbf{e})) \\ 0 & \text{otherwise.} \end{cases}$$

The original SDP definition, however, assumed that D is a binary variable, where $\mathcal{F}(Pr(D \mid \mathbf{e})) = d$ when $Pr(d \mid \mathbf{e}) \geq T$ for some threshold T (Darwiche & Choi, 2010).

We now consider the use of SDP as a stopping criterion in the context of expected-utility decisions and influence diagrams (Howard & Matheson, 1984). In particular, we

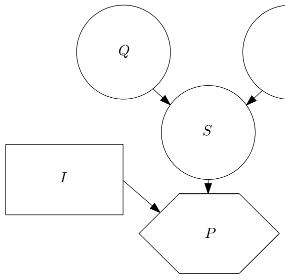

Figure 10: An influence diagram for an investment problem.

show that by using the SDP, we can distinguish high-risk, high-reward scenarios from low-risk, low-reward scenarios that are otherwise indistinguishable when we consider the usage of VOI/utilities alone.

Consider the influence diagram in Figure 10, which consists of a Bayesian network with three variables (C, Q and S), a decision node I, and a utility node P that is a direct function of the utility function u. This influence diagram models an investment problem in which a venture capital firm is deciding whether to invest an amount of \$5 million in a tech startup ($I = T$) or allowing the money to collect interest in the bank ($I = F$). In this example, the profit of the investment (P) depends on the decision (I) and the success of the company (S), which in turn depends on two factors: (1) whether the existing competitor companies are successful (C) and (2) whether the the co-founders of the startup have a high quality, original idea (Q). Both C and Q are unobserved initially and independent of each other. Variable S is the latent hypothesis variable in this case and thus cannot be observed. Variables C and Q, however, can be observed for a price.

The goal here is to choose the decision $I = i$ with the maximum expected utility:

$$\mathcal{EU}(i \mid \mathbf{e}) = \sum_s Pr(s \mid \mathbf{e}) u(i, s),$$

where $u(i, s)$ is the utility of decision $I = i$ given evidence $\mathbf{e}$ on variables C and Q.

Figures 11 and 12 contain two different parameterizations of the influence diagram in Figure 10. We will refer to these as different scenarios of the investment problem.

In both scenarios, given no evidence on variables C and Q, the best decision is $I = F$, with an expected utility of \$500K. A decision maker may commit to this decision or decide to observe variables C and Q, with the hope of finding a better decision in light of the additional information. The classical stopping criterion here is to compute the *maximum* expected utility given that we observe variables C and Q (Heckerman et al., 1993; Dittmer & Jensen, 1997):

$$\max_i \sum_{c,q} \mathcal{EU}(i \mid c, q) Pr(c, q).$$

In both scenarios, the maximum expected utility comes out to \$1,180K, showing that further observations may lead to a better decision.[12]

12. According to the formulation of Krause and Guestrin (2009), we have computed the VOI for variables C and Q using the reward function.

Q	$Pr(Q)$
T	0.4
F	0.6

Q	C	$Pr(S = T \mid .)$
T	T	0.60
T	F	0.90
F	T	0.20
F	F	0.30

C	$Pr(C)$
T	0.6
F	0.4

I	S	$u(I, S)$
T	T	$\$5 \times 10^6$
T	F	$-\$5 \times 10^6$
F	T	$\$5 \times 10^5$
F	F	$\$5 \times 10^5$

Figure 11: A parameterization of the influence diagram in Figure 10.

Q	$Pr(Q)$
T	0.1
F	0.9

Q	C	$Pr(S = T \mid .)$
T	T	0.05
T	F	0.98
F	T	0.01
F	F	0.05

C	$Pr(C)$
T	0.9
F	0.1

I	S	$u(I, S)$
T	T	$\$7 \times 10^7$
T	F	$-\$5 \times 10^6$
F	T	$\$5 \times 10^5$
F	F	$\$5 \times 10^5$

Figure 12: A parameterization of the influence diagram in Figure 10.

Up to this point, the above two scenarios are indistinguishable from the viewpoint of classical decision making tools. Remember that Krause and Guestrin (2009) and Bilgic and Getoor (2011) remark that the budget for observations should be expended so long as the value of information of the observation is greater than the cost of observation. According to those selection criteria, both of the variables should thus be observed, as the expected financial gain could very well increase.

The SDP, however, finds that these two scenarios are very different. In particular, with respect to variables C and Q, the SDP is 60% in the first scenario and is 99% in the second scenario. That is, even though we stand to make a better decision in both scenarios upon observing variables C and Q (at least with respect to financial gain), and even though the expected benefit from such observations is the same in both scenarios, it is very unlikely that we would change the current decision of $I = F$ in the second scenario in comparison to the first. Hence, given the additional information provided by the SDP, a decision maker may act quite differently in these two scenarios. Indeed, when we take a closer look at the second scenario, there is a state of the world (when $S = T$) where deciding to invest would yield a very large financial gain. However, the chance of this state manifesting itself is extremely small (analogous to a lottery), meaning that a risk-conscious decision maker may be more averse to "gamble" in the second scenario and even waste resources to observe the variables C and D. Note that for this example we have assumed that the utility does not incorporate any "risk-factor", as if it did a rational decision maker would then always choose to gather more information despite a low probability of changing the current decision.

This illustrates the usefulness of SDP as a stopping criterion in the context of expected-utility decisions and influence diagrams. Namely, using SDP, we can distinguish between two very different scenarios, that are otherwise indistinguishable when we consider utilities alone.

Appendix B. Proofs

In this section we provide proofs for Theorems 2 and 3.

Proof of Theorem 2. We show **D-EPT** is $\mathrm{PP}^{\mathrm{PP}}$-hard by reduction from the following decision problem **D-SDP**, which corresponds to the originally proposed notion of same-decision probability for threshold-based decisions (Darwiche & Choi, 2010).

D-SDP: Given a decision based on probability $Pr(d \mid \mathbf{e})$ surpassing a threshold T, a set of unobserved variables $\mathbf{H}$, and a probability p, is the same-decision probability:

$$\sum_{\mathbf{h}} [Pr(d \mid \mathbf{h}, \mathbf{e}) \geq T] Pr(\mathbf{h} \mid \mathbf{e}) \tag{14}$$

greater than p?

Here, [.] denotes an indicator function which evaluates to 1 if the enclosed expression is satisfied, and 0 otherwise. **D-SDP** was shown to be $\mathrm{PP}^{\mathrm{PP}}$-complete by Choi et al. (2012).

This same-decision probability corresponds to an expectation with respect to the distribution $Pr(\mathbf{H} \mid \mathbf{e})$, using the reward function:

$$R(Pr(D \mid \mathbf{h}, \mathbf{e})) = \begin{cases} 1 & \text{if } Pr(d \mid \mathbf{h}, \mathbf{e}) \geq T \\ 0 & \text{otherwise.} \end{cases}$$

Thus the same-decision probability is $\geq T$ iff this expectation is $\geq T$. $\square$

Proof of Theorem 3. To show that **D-EPT** is in $\mathrm{PP}^{\mathrm{PP}}$, we provide a probabilistic polynomial-time algorithm, with access to a PP oracle, that answers the decision problem **D-EPT** correctly with probability greater than $\frac{1}{2}$. This proof generalizes and simplifies the proof given by Choi et al. (2012) for **D-SDP**.

Consider the following probabilistic algorithm that determines if $\mathbb{E} > N$:

1. Sample a complete instantiation $\mathbf{x}$ from the Bayesian network, with probability $Pr(\mathbf{x})$. We can do this in linear time, using forward sampling (Henrion, 1986).

2. If $\mathbf{x}$ is compatible with $\mathbf{e}$, we can use a PP-oracle to compute $t = R(Pr(D \mid \mathbf{h}, \mathbf{e}))$. First, the reward function R can be computed in polynomial time, by definition. Second, $Pr(D \mid \mathbf{h}, \mathbf{e})$ can be computed using a PP-oracle, since the inference is #P-complete (Roth, 1996), and since $\mathrm{P}^{\mathrm{PP}} = \mathrm{P}^{\#\mathrm{P}}$.

3. Define a function $a(t) = \frac{1}{2} + \frac{1}{2}\frac{t-N}{u-l}$, which defines a probability used by our probabilistic algorithm to guess whether $\mathbb{E} > N$ (see Lemma 1).

4. Declare that $\mathbb{E} > N$ with probability:

   - $a(t)$ if $\mathbf{x}$ is compatible with $\mathbf{e}$;
   - $\frac{1}{2}$ if $\mathbf{x}$ is not compatible with $\mathbf{e}$.

The probability of declaring $\mathbb{E} > N$ is:

$$r = \sum_{\mathbf{h}} a(t) Pr(\mathbf{h}, \mathbf{e}) + \frac{1}{2}(1 - Pr(\mathbf{e})) \tag{15}$$

which is greater than $\frac{1}{2}$ iff the following set of equivalent statements hold:

$$\begin{aligned}
\sum_{\mathbf{h}} a(t) Pr(\mathbf{h}, \mathbf{e}) &> \frac{Pr(\mathbf{e})}{2} \\
\sum_{\mathbf{h}} a(t) Pr(\mathbf{h} \mid \mathbf{e}) &> \frac{1}{2} \\
\sum_{\mathbf{h}} \left(\frac{1}{2} + \frac{1}{2}\frac{t-N}{u-l}\right) Pr(\mathbf{h} \mid \mathbf{e}) &> \frac{1}{2} \\
\sum_{\mathbf{h}} \left(\frac{1}{2}\frac{t-N}{u-l}\right) Pr(\mathbf{h} \mid \mathbf{e}) &> 0 \\
\sum_{\mathbf{h}} (t-N) Pr(\mathbf{h} \mid \mathbf{e}) &> 0 \\
\sum_{\mathbf{h}} R(Pr(D \mid \mathbf{h}, \mathbf{e})) Pr(\mathbf{h} \mid \mathbf{e}) &> N.
\end{aligned}$$

Thus $r > \frac{1}{2}$ iff $\mathbb{E} > N$. □

Lemma 1. *The function $a(t) = \frac{1}{2} + \frac{1}{2}\frac{t-N}{u-l}$ maps a reward t to a probability in* $[0, 1]$.

Proof. Values u and l are given, and denote upper and lower bounds on the reward t, but also the threshold N. Thus $\frac{t-N}{u-l}$ is in $[-1, 1]$. □

Note that $a(t)$ denotes a probability used by our algorithm to declare whether $\mathbb{E} > N$, which is higher or lower depending on the value of the reward $t = R(Pr(D \mid \mathbf{h}, \mathbf{e}))$.

Appendix C. Conditional Probability Tables

In this section we provide conditional probability tables for the networks in Figures 1, 2, 3, and 4.

D	$Pr(D)$
$+$	0.5
$-$	0.5

D	X_1	$Pr(X_1 \mid D)$
$+$	$+$	0.9
$+$	$-$	0.1
$-$	$+$	0.1
$-$	$-$	0.9

X_1	X_2	$Pr(X_2 \mid X_1)$
$+$	$+$	0.9
$+$	$-$	0.1
$-$	$+$	0.1
$-$	$-$	0.9

H_i	X_i	E_i	$Pr(E_i \mid H_i, X_i)$
t	$+$	$+$	1.0
t	$-$	$+$	0.0
p	$+$	$+$	1.0
p	$-$	$+$	1.0
n	$+$	$+$	0.0
n	$-$	$+$	0.0
l	$+$	$+$	0.0
l	$-$	$+$	1.0

H_i	$Pr(H_i)$
t	0.81
p	0.09
n	0.09
l	0.01

Figure 13: The CPTs for the Bayesian network given in Figure 1. Note that for the CPTs of variables E_i, only the lines for the case $E_i = +$ are given, since $Pr(E_i = - | H_i, X_i) = 1 - Pr(E_i = + | H_i, X_i)$.

D	S_1	$Pr(S_1 \mid D)$
$+$	$+$	0.65
$+$	$-$	0.35
$-$	$+$	0.30
$-$	$-$	0.70

D	S_3	$Pr(S_3 \mid D)$
$+$	$+$	0.65
$+$	$-$	0.35
$-$	$+$	0.35
$-$	$-$	0.65

D	S_2	$Pr(S_2 \mid D)$
$+$	$+$	0.60
$+$	$-$	0.40
$-$	$+$	0.30
$-$	$-$	0.70

D	S_4	$Pr(S_4 \mid D)$
$+$	$+$	0.65
$+$	$-$	0.35
$-$	$+$	0.35
$-$	$-$	0.65

Table 5: CPTs for the network in Figure 2. Parameterization 2.

D	S_1	$Pr(S_1 \mid D)$
$+$	$+$	0.50
$+$	$-$	0.50
$-$	$+$	0.40
$-$	$-$	0.60

D	S_3	$Pr(S_3 \mid D)$
$+$	$+$	0.55
$+$	$-$	0.45
$-$	$+$	0.45
$-$	$-$	0.55

D	S_2	$Pr(S_2 \mid D)$
$+$	$+$	0.50
$+$	$-$	0.50
$-$	$+$	0.40
$-$	$-$	0.60

D	S_4	$Pr(S_4 \mid D)$
$+$	$+$	0.55
$+$	$-$	0.45
$-$	$+$	0.45
$-$	$-$	0.55

Table 6: CPTs for the network in Figure 2. Parameterization 3.

D	S_1	$Pr(S_1 \mid D)$
$+$	$+$	0.8
$+$	$-$	0.2
$-$	$+$	0.2
$-$	$-$	0.8

D	S_2	$Pr(S_2 \mid D)$
$+$	$+$	0.75
$+$	$\circ$	0.2
$+$	$-$	0.05
$-$	$+$	0.05
$-$	$\circ$	0.2
$-$	$-$	0.75

Table 7: CPTs for the Bayesian network in Figure 3.

D	H_1	$Pr(H_1 \mid D)$
$+$	$+$	0.80
$+$	$-$	0.20
$-$	$+$	0.10
$-$	$-$	0.90

D	H_2	$Pr(H_2 \mid D)$
$+$	$+$	0.70
$+$	$-$	0.30
$-$	$+$	0.30
$-$	$-$	0.70

Table 8: CPTs for the network in Figure 4. $Pr(H_3 \mid D)$, $Pr(E_1 \mid D)$ and $Pr(H_2|D)$ are equal.

References

Arroyo, I., & Woolf, B. (2005). Inferring learning and attitudes from a Bayesian network of log file data. In *Proceedings of the 12th International Conference on Artificial Intelligence in Education*, pp. 33–40.

Bache, K., & Lichman, M. (2013). UCI machine learning repository..

Bilgic, M., & Getoor, L. (2011). Value of information lattice: Exploiting probabilistic independence for effective feature subset acquisition. *Journal of Artificial Intelligence Research (JAIR)*, *41*, 69–95.

Butz, C. J., Hua, S., & Maguire, R. B. (2004). A web-based intelligent tutoring system for computer programming. In *Web Intelligence*, pp. 159–165. IEEE Computer Society.

Chan, H., & Darwiche, A. (2003). Reasoning about Bayesian network classifiers. In *Proceedings of the 19th Conference in Uncertainty in Artificial Intelligence*, pp. 107–115.

Chen, J., Low, K. H., Tan, C. K.-Y., Oran, A., Jaillet, P., Dolan, J., & Sukhatme, G. (2012a). Decentralized data fusion and active sensing with mobile sensors for modeling and predicting spatiotemporal traffic phenomena. In *Proceedings of the Twenty-Eighth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-12)*, pp. 163–173, Corvallis, Oregon. AUAI Press.

Chen, S., Choi, A., & Darwiche, A. (2012b). The Same-Decision Probability: A new tool for decision making. In *Proceedings of the Sixth European Workshop on Probabilistic Graphical Models*, pp. 51–58.

Chen, S., Choi, A., & Darwiche, A. (2013). An exact algorithm for computing the Same-Decision Probability. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pp. 2525–2531.

Choi, A., Xue, Y., & Darwiche, A. (2012). Same-Decision Probability: A confidence measure for threshold-based decisions. *International Journal of Approximate Reasoning (IJAR)*, *2*, 1415–1428.

Conati, C., Gertner, A., & VanLehn, K. (2002). Using Bayesian networks to manage uncertainty in student modeling. *User Modeling and User-Adapted Interaction*, *12*(4), 371–417.

Cover, T. M., & Thomas, J. A. (1991). *Elements of Information Theory.* Wiley-Interscience.

Darwiche, A. (2009). *Modeling and Reasoning with Bayesian Networks* (1st edition). Cambridge University Press.

Darwiche, A., & Choi, A. (2010). Same-Decision Probability: A confidence measure for threshold-based decisions under noisy sensors. In *Proceedings of the Fifth European Workshop on Probabilistic Graphical Models*, pp. 113–120.

Dechter, R. (1999). Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, *113*(1), 41–85.

Dittmer, S., & Jensen, F. (1997). Myopic value of information in influence diagrams. In *Proceedings of the Thirteenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-97)*, pp. 142–149.

Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, *29*(2-3), 131–163.

Gao, T., & Koller, D. (2011). Active classification based on value of classifier. In *Advances in Neural Information Processing Systems (NIPS 2011)*.

Gertner, A. S., Conati, C., & VanLehn, K. (1998). Procedural help in Andes: Generating hints using a Bayesian network student model. In *Proceedings of the National Conference on Artificial Intelligence*, pp. 106–111.

Glasziou, P., & Hilden, J. (1989). Test selection measures. *Medical Decision Making*, *9*(2), 133–141.

Greiner, R., Grove, A. J., & Roth, D. (2002). Learning cost-sensitive active classifiers. *Artificial Intelligence*, *139*(2), 137–174.

Hamscher, W., Console, L., & de Kleer, J. (Eds.). (1992). *Readings in Model-Based Diagnosis*. Morgan Kaufmann Publishers Inc.

Heckerman, D., Breese, J. S., & Rommelse, K. (1995). Decision-theoretic troubleshooting. *Communications of the ACM*, *38*(3), 49–57.

Heckerman, D., Horvitz, E., & Middleton, B. (1993). An approximate nonmyopic computation for value of information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *15*(3), 292–298.

Henrion, M. (1986). Propagating uncertainty in Bayesian networks by probabilistic logic sampling. In *Proceedings of the Second Annual Conference on Uncertainty in Artificial Intelligence (UAI-86)*, pp. 149–163.

Howard, R. A. (1966). Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, *2*(1), 22–26.

Howard, R. A., & Matheson, J. E. (Eds.). (1984). *Readings on the Principles and Applications of Decision Analysis*. Strategic Decision Group.

Ide, J. S., Cozman, F. G., & Ramos, F. T. (2004). Generating random Bayesian networks with constraints on induced width. In *Proceedings of the 16th European Conference on Artificial Intelligence*, pp. 323–327.

Jordan, A. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. *Advances in Neural Information Processing Systems*, *14*, 841.

Kahn, C. E., Roberts, L. M., Shaffer, K. A., & Haddawy, P. (1997). Construction of a Bayesian network for mammographic diagnosis of breast cancer. *Computers in Biology and Medicine*, *27*(1), 19–29.

Karp, R. M. (1972). Reducibility among combinatorial problems. In *Complexity of Computer Computations*. Springer.

Kjærulff, U. B., & Madsen, A. L. (2008). *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*. Springer.

Krause, A., & Guestrin, C. (2005). Near-optimal nonmyopic value of information in graphical models. In *21st Conference on Uncertainty in Artificial Intelligence*, pp. 324–331.

Krause, A., & Guestrin, C. (2009). Optimal value of information in graphical models. *Journal of Artificial Intelligence Research (JAIR)*, *35*, 557–591.

Kruegel, C., Mutz, D., Robertson, W., & Valeur, F. (2003). Bayesian event classification for intrusion detection. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*.

Liao, W., & Ji, Q. (2008). Efficient non-myopic value-of-information computation for influence diagrams. *International Journal of Approximate Reasoning*, *49*(2), 436–450.

Lindley, D. V. (1956). On a measure of the information provided by an experiment. *Annals of Mathematical Statistics*, *27*(4), 986–1005.

Lu, T.-C., & Przytula, K. W. (2006). Focusing strategies for multiple fault diagnosis. In *Proceedings of the 19th International FLAIRS Conference*, pp. 842–847.

Millán, E., Descalco, L., Castillo, G., Oliveira, P., & Diogo, S. (2013). Using Bayesian networks to improve knowledge assessment. *Computers & Education*, *60*(1), 436–447.

Modelo-Howard, G., Bagchi, S., & Lebanon, G. (2008). Determining placement of intrusion detectors for a distributed application through Bayesian network modeling. In *Proceedings of the 11th International Symposium on Recent Advances in Intrusion Detection*, pp. 271–290.

Munie, M., & Shoham, Y. (2008). Optimal testing of structured knowledge. In *AAAI'08: Proceedings of the 23rd National Conference on Artificial intelligence*, pp. 1069–1074.

Ognibene, D., & Demiris, Y. (2013). Towards active event recognition. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pp. 2495–2501.

Park, J. D., & Darwiche, A. (2004). Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research (JAIR)*, *21*, 101–133.

Pauker, S. G., & Kassirer, J. P. (1980). The threshold approach to clinical decision making.. *The New England Journal of Medicine*, *302*(20), 1109–17.

Raiffa, H. (1968). *Decision Analysis – Introductory Lectures on Choices under Uncertainty.* Addison-Wesley.

Ramoni, M., & Sebastiani, P. (2001). Robust Bayes classifiers. *Artificial Intelligence*, *125*(1-2), 209–226.

Roth, D. (1996). On the hardness of approximate reasoning. *Artificial Intelligence*, *82*, 273 – 302.

Shann, M., & Seuken, S. (2013). An active learning approach to home heating in the smart grid. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pp. 2892–2899.

Stratonovich, R. (1965). On value of information. *Izvestiya of USSR Academy of Sciences, Technical Cybernetics*, *5*, 3–12.

van der Gaag, L. C., & Coupé, V. M. H. (1999). Sensitivity analysis for threshold decision making with Bayesian belief networks. In *AI*IA*, pp. 37–48.

Vomlel, J. (2004). Bayesian networks in educational testing. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *12*(supp01), 83–100.

Xenos, M. (2004). Prediction and assessment of student behaviour in open and distance education in computers using Bayesian networks. *Computers & Education*, *43*(4), 345–359.

Yu, S., Krishnapuram, B., Rosales, R., & Rao, R. B. (2009). Active sensing. In *International Conference on Artificial Intelligence and Statistics*, pp. 639–646.

Zhang, N. L. (1998). Probabilistic inference in influence diagrams. In *Computational Intelligence*, pp. 514–522.

Zhang, Y., & Ji, Q. (2010). Efficient sensor selection for active information fusion. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, *40*(3), 719–728.