Bayesian Networks and Probabilistic Inference

Wes Robbins wesley.robbins@student.montana.edu
Department of Computer Science
Montana State University
Bozeman, MT 59717, USA

Dave Miller david.miller26@student.montana.edu
Department of Computer Science
Montana State University
Bozeman, MT 59715, USA

Editor: Asad Noor

Abstract

Bayesian Networks offer a natural way to represent probabilistic information for many practical problems. Inference can be performed with Bayesian networks by extracting marginal probabilities for any subset of random variables. Furthermore, inference methods account for observed states of random variables, known as evidence. While inference can be performed exactly, it becomes intractable for large networks. Another approach is to approximate marginal probabilities which can significantly decrease time complexity. In this paper, we implement a Bayesian network and two inference methods. We implement one exact inference method, variable elimination, and one approximate inference method, Gibbs sampling. We perform experiments on five predefined networks of various sizes. In our experiments, we find that Gibbs sampling can run more efficiently, while having an average of 11% error for inferred probabilities. Code for this project can be found at https://github.com/WesRobbins/AI-CSCI-446.

1. Introduction

Probabilistic reasoning methods are an impactful field within Artificial Intelligence, and have been shown to be effective in many real-world settings. Probabilistic reasoning methods offer a flexible approach for storing information about the world, which is much less restrictive than only storing information that can be determined absolutely. Due to this, probabilistic methods are often far more powerful than strictly logic based methods.

Graph models are an effective way to store and process probabilistic information. A well-known probabilistic graphical model is the Bayesian network. A Bayesian network is directed acyclic graph where nodes represent random variables and directed edges represent conditionality between variables (Russell and Norvig (2021)). Probabilities for each possible value of each node given the node's parents must be provided to use the network. These probabilities can be given by expert knowledge, or learned if sufficient data exists.

As in any probabilistic inference system, a Bayesian network performs inference by extracting posterior probability distributions for any subset of the random variables, known as query variables. These marginal distributions can be calculated exactly with general exact inference methods. During exact inference, non-query variables are marginalized out out of the network. Because naively marginalizing is exponential with respect to network depth, it is common to store submarginalizations in the inference process (dynamic programming). One such method is variable elimination. However, even dynamic programming approaches may not be efficient enough if inference must be fast or the network is very large. Another approach is to approximate marginal probabilities for the query variables. Several algorithms have proposed for approximating marginal

Network Name	Nodes	Edges
Alarm	37	46
Child	20	25
Hailfinder	56	66
Insurance	27	52
Win95pts	76	112

Table 1: Information about the five networks used in our experiments.

probabilities from a Bayesian network including: direct sampling, importance sampling, and Gibbs sampling.

For this project, we implemented a Bayesian network framework along with two inference methods: variable elimination and Gibbs sampling. We use five given networks to test our implementation. We compare the difference in performance between our exact inference method (ground truth) and our approximate inference method. Additionally, we compare the computational efficiency of the methods. Our hypothesis for this work is below.

Hypothesis Variable Elimination is an exact inference method, so we expect variable elimination to give us correct marginal probabilities for all problems. We hypothesize that Gibbs sampling will give near-exact probabilities for all problems. Specifically, we hypothesize that Gibbs sampling will have less than 5% error for $n = 100$ samples. In terms of efficiency, we expect Gibbs sampling to be require 5-10x less computation due to the fact that it is approximating the marginal probabilities instead of computing all marginalizations.

2. Problem & Approach

As previously mentioned, in this project, we implement a Bayesian network and two inference methods. The goal of inference is to calculate marginal probabilities of all query variables. Inference must account for evidence and the non-query variables. More explicitly, we try to find marginal probabilities of query variables Q given the non-query variables X and evidence E:

$$P(Q|X, E). \tag{1}$$

We answer this questions for several networks, several query variables, and several sets of evidence. Information about the five networks that are used can be found in Table 1. More information about the specific problems can be sound in Section 4.

Variable Elimination

The first inference method we implement is variable elimination, an exact inference method. All exact inference methods must marginalize all non-query variables X in order to obtain the marginal probabilities. Variable elimination is more efficient than naive marginalization, as it implicitly stores sub-marginalizations during the elimination process.

From conditional probability, it is given that

$$P(Q|E) = \alpha \sum_{X} P(Q, E, x) \tag{2}$$

where α is a normalizing scalar. To find the marginal probability, we *must* marginalize not-query variables. In the naive approach, marginalizing each variable (i.e calculating each term $P(Q, E, x)$) may require a long chain rule operation with exponential time complexity.

Variable elimination significantly improves efficiency by storing intermediate marginalizations of non-query variables. The variable elimination algorithm iterates through all non-query variables, and eliminates each variable v. As we go through the elimination process, we no longer have true probability distributions, but rather factors of the random variables. By performing a pointwise multiplication over all factors that contain v and then summing v from the pointwise product (this is also know as a sum-reduction), we effectively marginalize v from any f. This can be seen in the following equation

$$\tau(x_a, x_b, x_c) = \sum_v \phi_1(v, x_a)...\phi_n(v, x_b, x_c) \tag{3}$$

where n is the number of relevant factors to v and τ is the new factor created from this operation (Cozman et al. (2000)). The new factor τ is added to the list of factors and all factors are removed from the factors list. Intuitively, we view this as merging probabilities of v into all relevant probabilities.

After eliminating all non-query variables, we are left with a factor that only contains the query variables. Once we normalize this factor (multiply by α) we have obtained the marginal distributions.

Ordering As a heuristic for variable elimination order, we first eliminate variables with the minimum number of children and parents. We tried several different heuristics and found this to be the most efficient. We find the ordering to be massively important, which we discuss further in Section 4.

Implementation Details We implement variable elimination with a single for loop over an numpy einsum operation for the sum-reduction of each variable.

Gibbs Sampling

Gibbs Sampling is an approximate inference method that uses a Markov chain Monte Carlo algorithm to sample from the conditional probability tables given by a Bayesian network. It is useful when direct sampling is difficult or when the size of a network restricts efficient exact inference. A drawback of Gibbs Sampling js that it is difficult to predict how many samples to take to ensure some amount of precision. Additionally, it can fail under certain conditions.

Gibbs Sampling works by creating a Markov chain of samples where each sample is drawn by conditioning on the values of the last sample (GENG et al. (2000)). Thus, instead of any marginalization, we can directly look up conditional probabilities using the Bayesian Network.

$$P(x_t|x_{t-1}) \tag{4}$$

The first sample is formed using forward sampling, moving through a topological sort of the nodes and sampling a value for each parent before their child. When evidence is given, we do not sample any of the evidence nodes, thereby reducing the complexity even further. Once our sampler converges, we can calculate the marginal probability for each value q of the query variables as shown.

$$P(q \in Q|E) = \frac{\text{\# of samples with q}}{\text{total \# of samples}} \tag{5}$$

As stated above, it is difficult to guess how many samples to take in order to get acceptable accuracy, and is dependent on the network complexity. Furthermore, sometimes the literature recommends the first n samples are discarded in a *burn-in* period. In this period we wait for the Markov chain to reach a stationary distribution. In our experience, averaging after a *burn-in* period did not significantly change the distributions and was set to false by default in the code. Interestingly, we added a setting to only sample from the set of the union of the Markov blankets of the query variables as opposed to the entire network and saw that the distributions were quite similar. This could be because the query variables are independent from the rest of the network given their Markov blankets, but we expected that the approximations would be somewhat less accurate on account of the variables outside the blanket only being sampled once at the beginning.

3. Experimental Approach

For our analysis on these inference algorithms, we are given five different Bayesian Networks of varying size (see Table 1). We calculate marginal probability distributions for some set of variables given 3-5 different sets of evidence. We compare the efficiency of the two algorithms using a decision metric. We test the precision of Gibbs Sampling by comparing it to the results of variable elimination. If the implementation is correct, variable elimination is guaranteed to give correct marginal distributions because it is a general exact inference algorithm. We use a third party library[1] to check that our variable elimination implementation generates correct marginal probabilities.

Decision Metric In order to compare the efficiency of Gibbs sampling and variable elimination, we use the following to track decision for each algorithm. For Gibbs sampling, each sample is of cost 1. For variable elimination each floating point multiplication is of cost 1. Under a standard model of computation, both these operations are of equal cost. Furthermore, these are the two primary operations for each of their respective algorithms. While we believe this metric offers reasonable efficiency comparison between the two algorithms, it is not perfect.

4. Results

Performance

First, we compare the results of our variable elimination algorithm with a third party library. We find that our algorithm correctly infers exact marginal probabilities. With these ground truth marginal probabilities, we can test the precision of Gibbs sampling.

We find that Gibbs sampling decent inference precision. As seen in Table 4, Gibbs sampling has an 11% error on average, with lowest performance on queries with evidence. This is lower performance than we hypothesized. However, the error was greatly impaced by the Win95pts network, on which Gibbs Sampling performed extremely poorly. The conditions of this network could be a failure case of Gibbs, where the sampling becomes trapped in the high probability zones of boolean variables (Russell and Norvig (2021)). Without the Win95pts network, our error becomes 7% on average and 2% without evidence.

Efficiency

We find that according to our decision metric, variable elimination is on average 68% more computationally expensive than variable elimination. While this is not insignificant, we hypothesized Gibbs sampling would offer a greater magnitude of efficiency improvement. Additionally, we find that both algorithms are relatively fast over all, running in less than a .01 of a second. In Table 4, we show an evaluation of the efficiency of the algorithms on each network.

Effect of Elimination Order We find that the ordering of the variables in variable elimination is quite significant. When running variable elimination without ordering the variables, the program would accumulate factors with many variable resulting in high-dimensional matrices. This quickly used up massive amounts of space (up to 90GB) and often crashed the program. On the other hand, once the variables were sorted the program ran much faster and used way less memory.

We found that sorting the variables in ascending order based on the number of parents *and* children provided the fastest run-time, followed closely by a sort considering just the number parents, and considering just the number of children was the worst of the orderings we used.

1. https://github.com/erdogant/bnlearn

Variable	Algorithm	No Evidence	Low Evidence	Medium Evidence
HYPOVOLEMIA	Var Elim	(.200, .800)	(.554, .446)	(.219 .781)
HYPOVOLEMIA	Gibbs	(.216, .784)	(.190, .810)	(.199, .801)
LVFAILURE	Var Elim	(.050, .950)	(.250, .750)	(.962, .038)
LVFAILURE	Gibbs	(.052, .948)	(.052, .948)	(.844, .156)
ERRLOWOUTPUT	Var Elim	(.050, .950)	(.009, .991)	(.604, .396)
ERRLOWOUTPUT	Gibbs	(.040, .960)	(.010, .990)	(.020, .980)

Table 2: Marginal Probabilities for each query variable from the *Child* network. For all marginal probabilities refer to `results/results.txt` file in our repository.

Network Name	Low Evidence	Medium Evidence	High Evidence	Average
Alarm	0.009	0.18	0.24	0.14
Child	0.02	0.08	0.08	0.06
Hailfinder	0.01	0.06	0.05	0.04
Insurance	0.07	0.09	0.06	0.07
Win95pts	0.35	0.25	0.16	0.23
Total Average	**0.09**	**0.12**	**0.14**	**0.11**

Table 3: Percent error of Gibbs sampling for each network and each level of evidence. Percent error obtained from comparing to variable elimination. Only 3 of 6 evidence levels are shown for Win95pts.

Network Name	Variable Elimination	Gibbs Sampling
Alarm	943	640
Child	732	400
Hailfinder	2,716	1,120
Insurance	907	540
Win95pts	1,800	1,520
Average	**1420**	**844**

Table 4: Efficiency of the two algorithms - scored by the decision metric discussed in section 3.

5. Summary

This paper compares the performances and run-times of two inference methods used on various Bayesian Networks. Variable Elimination is an exact inference method that marginalizes out unwanted variables and keeps the complexity down by using dynamic programming. Gibbs Sampling is an approximate inference algorithm that samples directly from the given conditional probability tables using a Markov chain Monte Carlo method. We compare the performance of our approximate method against the true values given by our exact inference method and explore the advantages and drawbacks of these algorithms. We calculated marginal probability distributions using various sets of evidence on five different Bayesian Networks and found that while Variable Elimination gives exact values, it is somewhat slow and has high space overhead, while Gibbs Sampling computes the same values with around 11% error rather quickly.

References

Fabio Gagliardi Cozman et al. Generalizing variable elimination in bayesian networks. In *Workshop on probabilistic reasoning in artificial intelligence*, pages 27–32. Citeseer, 2000.

Zhi GENG, Kai-Can LI, et al. Graphical models with missing data and related algorithms. In *Proceedings of the annual meeting of Japanese Society of Computational Statistics 14*, pages 104–107. Japanese Society of Computational Statistics, 2000.

Stuart Russell and Peter Norvig. Artificial intelligence: a modern approach, 2021.