AI : PRINCIPLES & TECHNIQUES
BAYESIAN NETWORKS

INFERENCING QUERIES USING VARIABLE ELIMINATION IN BAYESIAN NETWORKS

By:
Juan Navarro — s1097545
Luca Laber — s1089333

Table of Contents

INTRODUCTION

In this report we will explore the implementation of the Variable Elimination (VE) algorithm for inference in Bayesian Networks. For context, Bayesian Networks "is a probabilistic graphical model that represents a set of variables and their conditional dependencies" *(Wikipedia contributors, n.d.)*. In other words, Bayesian Networks provide a structured representation of probabilistic relationships, enabling efficient computation of conditional probabilities and reasoning under uncertainty.

As shown in the slides, a simple Bayesian Network can represent relationships between variables such as **Tampering**, **Fire**, **Alarm**, **Smoke**, and **Leaving**. In this network:

- **Tampering** and **Fire** influence whether the **Alarm** sounds.
- **Fire** also impacts whether there is **Smoke**.
- Hearing the **Alarm** or seeing **Smoke** might make someone to **Leave**.

These networks enable reasoning under uncertainty by computing probabilities like P(Fire∣Alarm=True). The primary objective of this assignment is to implement and apply the VE algorithm to solve such inference queries effectively.

Our assignment is structured in two main phases. To form the basis for the VE operations we have to conclude the first phase which involves constructing and manipulating multidimensional factors. The second phase implements the VE algorithm itself, focusing on eliminating variables in a predefined order to compute desired probabilities. The effectiveness of the algorithm is evaluated by applying it to various inference queries on real-world and synthetic Bayesian Networks.

METHODS

In order to facilitate the implementation of the Variable Elimination algorithm, multi-dimensional factors were designed to model the conditional probability tables within the Bayesian Network. These factors form the foundation for the inference computations performed by the VE algorithm. The key operations performed on these factors include:

- **Reduction:** Removing variable states that are inconsistent with the given evidence. For example, if evidence is Alarm=True, the factor associated with P(Alarm) is reduced to only rows where Alarm=True.
- **Product:** Merging factors to calculate joint distributions over common variables.
- **Marginalization:** Eliminating variables by summing their contributions within the factors

Variable Elimination Explanation

The VE algorithm is an efficient method for performing inference in Bayesian Networks by systematically summing out irrelevant variables. As stated above this is achieved by reducing, combining, and marginalizing factors in a structured manner. The process involves the following:

1. Identifying **query variables**, which represent the probabilities to be calculated, and **evidence variables**, which are observed and fixed in the computation.

2. Selecting an **elimination order**, which determines the sequence in which variables are marginalized out.
3. Iteratively processing variables in the elimination order, performing reduction, product, and marginalization operations to compute intermediate factors.
4. Normalizing the final factor to produce a valid probability distribution for the query variables.

This approach ensures that computations are localized to the relevant parts of the network, minimizing redundant calculations.

Variable Elimination Algorithm Implementation

The implementation of the Variable Elimination (VE) algorithm follows a systematic approach to efficiently compute inference queries in Bayesian Networks. The process is structured as follows:

1. **Initialization:**
    - Define query and evidence variables based on the inference task.
    - Analyze the Bayesian Network to extract conditional probability tables and initialize the relevant factors.
2. **Factor Manipulations:**
    - **Reduction:** Apply evidence to reduce the dimensionality of the factors.
    - **Product:** Combine factors containing shared variables to compute joint distributions.
    - **Marginalization:** Eliminate variables by summing out their contributions from the factors.
3. **Iterative Elimination:**
    - Use a predetermined elimination order to systematically process variables. The choice of order significantly impacts computational efficiency.
    - For each variable in the order:
        - Multiply all factors containing the variable.
        - Marginalize the variable from the resulting factor.
        - Replace the processed factors with the newly created factor.
4. **Normalization:**
    - Normalize the final factor to ensure the resulting probability distribution is valid.

This structured approach ensures that the computations remain localized to the relevant parts of the network, avoiding unnecessary complexity and redundant calculations. The algorithm's design enables it to handle queries in both small and large networks efficiently, making it a widely used inference method.

The following visual taken from the slides (Lecture Slides, 2425 AI: Principles & Techniques, Bayesian Networks 2, pg. 30) shows the key steps of the Variable Elimination process:

Variable Elimination

a) What are the query variables?
b) What are the observed variables?
c) Write down:
   1) The product formula to compute the query
   2) The reduced formula based on the network structure
d) Identify factors and reduce observed variables
e) Fix an elimination ordering
f) For every variable Z in this ordering:
   a) Multiply factors containing Z
   b) Sum out Z to obtain new factor f_Z
   c) Remove the multiplied factors form the list and add f_Z
g) Normalize the result to make it a probability distribution

By staying true to this structure, the implemented VE algorithm effectively computes the desired probabilities for a variety of Bayesian Network inference tasks.

ANALYSIS

Complexity Analysis

The Variable Elimination (VE) algorithm's complexity depends on the Bayesian Network structure and the chosen elimination order. We can use the log from the execution to provide with any clear insights into the computations:

- **Elimination Order Impact**:
  - In the log, the elimination order is ['Burglary', 'Earthquake', 'Alarm', 'JohnCalls', 'MaryCalls']. As we know from everything stated above; this order significantly affects the size of intermediate factors.

    - For example, when eliminating ‘Earthquake', two relevant factors are combined, resulting in a new factor with four rows. This therefore demonstrates the exponential growth in the size of factors with bad elimination order choices.
- **Factor Size**:
    - Each variable elimination step requires:
        - **Factor product**: Merging factors with shared variables. For example, combining factors for ‘Earthquake’ and ‘Alarm’ results in a new joint factor with probabilities recalculated for all possible variable combinations.
        - **Marginalization**: Summing out a variable from a factor. For example, marginalizing ‘Earthquake’ results in a reduced factor for ‘Alarm’ with probabilities adjusted accordingly.
    - These operations scale exponentially with the number of variables in the largest factor.

Validation

The algorithm was cross-checked with various tools to verify that thorough testing had been done on the network:

- **Query**: P(Alarm∣Burglary=True)
- **Logs Validation**:
    - The intermediate steps align with expected behavior:
        - After applying evidence (Burglary=True), the algorithm correctly filtered irrelevant rows.
        - Other factors were then combined and marginalized step by step, just as we expected.
    - The final output probabilities were normalized correctly to sum to 1, therefore resulting in the data below:
        - P(Alarm=False)=0.0598
        - P(Alarm=True)=0.9402

Moreover when truly analyzing the log we can highlight areas for potential optimization For example, the given elimination order, we can see that multiple factors were combined and marginalized, therefore leading to an increase in runtime and memory usage. We know that as the network grows, these operations become increasingly computationally expensive.

DISCUSSION

Using everything said above, the implementation of the Variable Elimination (VE) algorithm gave us valuable insights into performing inference on Bayesian Networks. The complexity analysis underscored the significant impact of elimination order on computational performance, with badly chosen orders leading to exponential growth in factor sizes. This highlights the importance of incorporating heuristics to optimize the order and improve efficiency.

The validation process demonstrated the accuracy of the algorithm, with results aligning closely with known outputs and external tools. However, testing also revealed areas for potential optimization, particularly in memory usage and runtime scalability for larger networks. We must come to admit that improvements in error handling and dynamic elimination order selection could enhance the robustness of the overall code.

This assignment had its challenges as well. For instance, managing intermediate factors and scaling for complexity, can come to mirror the practical considerations of producing probabilistic reasoning in AI systems.

CONCLUSION

In conclusion, the Variable Elimination algorithm was implemented successfully and validated on Bayesian Networks, thus demonstrating its effectiveness in solving inference queries. In other words, by correctly applying reduction, product, and marginalization operations, the algorithm produced accurate probability distributions, even under varying conditions.

As stated above, determining the computational efficiency of the algorithm lies heavily on the elimination order. We can even state that while the current implementation is robust for smaller networks, scaling to larger, densely connected networks may present its challenges that mandate further optimization.

ACKNOWLEDGEMENTS

On behalf of Group 44, we want to thank the teacher, TAs for their guidance, and the content provided in brightspace that helped us understand what was needed. We also want to thank Ahmad Alansary for their assistance and collaboration when working on the project.

REFERENCES

- Wikipedia contributors. (n.d.). *Bayesian network*. In *Wikipedia*. Taken on December 16th, 2024, from https://en.wikipedia.org/wiki/Bayesian_network
- Lecture Slides, 2425 AI: Principles & Techniques (SEM1 V), Bayesian Networks 2, pg, Inference in BNs, pg. 7. https://brightspace.ru.nl/d2l/le/content/502381/viewContent/2722692/View
- Lecture Slides, 2425 AI: Principles & Techniques (SEM1 V), Bayesian Networks 2, pg, Variable Elimination, pg. 30. https://brightspace.ru.nl/d2l/le/content/502381/viewContent/2722692/View