Lab4 Tutorial

Exact Inference

高源

DATA130008.01

Dec.21st

Contents

Bayesian Networks

- Nodes for random variables
- Edges for conditional probability distributions
- DAG - Directed Acyclic Graph
- $\Rightarrow$ Conditional Independence Relationships

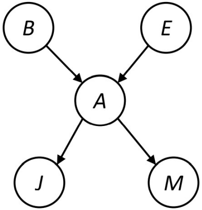


Figure 1: Sample Bayesian Network

Relevant Equations

$$P(\text{ Event } \mid \text{ Condition }) = \frac{P(\text{ Event }, \text{ Condition })}{P(\text{ Condition })}$$

$$P(\text{ Events }_A) = \sum_B P(\text{ Events }_A, \text{ Events }_B)$$

$$P(X_1, X_2, \ldots, X_n) = P(X_1 \mid X_1^p) \cdot P(X_2 \mid X_2^p) \cdot \ldots \cdot P(X_n \mid X_n^p)$$

Burglary Problem

(Chap14 P511) Burglary Problem: Query: P(Burglary| JohnCalls= True, MaryCalls= True)

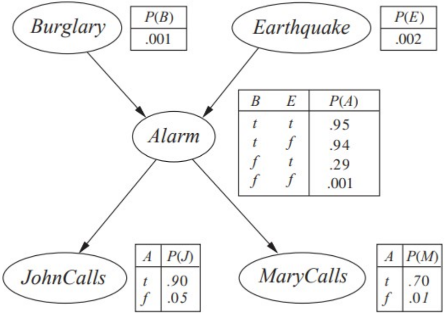


Figure 2: BN of burglary problem

Enumeration VS Elimination

Query: P(Burglary| JohnCalls= True, MaryCalls= True)

$$P(B|j,m) = \alpha P(B,j,m)$$

$$= \alpha \sum_e \sum_a P(B,j,m,e,a)$$

Enumeration

$$= \alpha \sum_e \sum_a P(b)P(e)P(a|b,e)P(j|a)P(m|a)$$

Elimination

$$= \alpha P(b) \boxed{\sum_e} P(e) \boxed{\sum_a} P(a|b,e)P(j|a)P(m|a)$$

Figure 3: Preparation

Enumeration

Query: P(Burglary| JohnCalls= True, MaryCalls= True)

$$\alpha \sum_{e} \sum_{a} P(b)P(e)P(a \mid b, e)P(j \mid a)P(m \mid a)$$

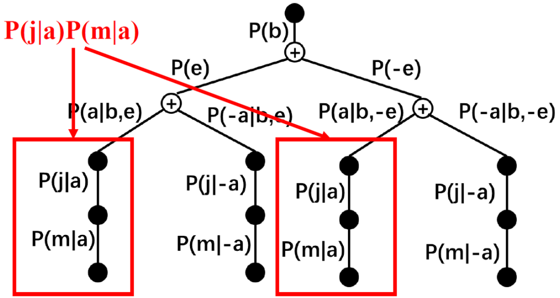
$$
\begin{aligned}
&P(B \mid +j, +m) \\
&= \sum_{e,a} P(B)P(e)P(a|B,e)P(+j|a)P(+m|a) \\
&= P(B)P(+e)P(+a|B,+e)P(+j|+a)P(+m|+a) + P(B)P(+e)P(-a|B,+e)P(+j|-a)P(+m|-a) \\
&\quad P(B)P(-e)P(+a|B,-e)P(+j|+a)P(+m|+a) + P(B)P(-e)P(-a|B,-e)P(+j|-a)P(+m|-a)
\end{aligned}
$$

Figure 4: Inference by Enumeration

Enumeration

Query: P(Burglary| JohnCalls= True, MaryCalls= True)

$$\alpha \sum_{e} \sum_{a} P(b)P(e)P(a \mid b, e)P(j \mid a)P(m \mid a)$$



;

Figure 5: Inference by Enumeration

Enumeration

Query: P(Burglary| JohnCalls= True, MaryCalls= True)

$$\alpha \sum_{e} \sum_{a} P(b)P(e)P(a \mid b,e)P(j \mid a)P(m \mid a)$$

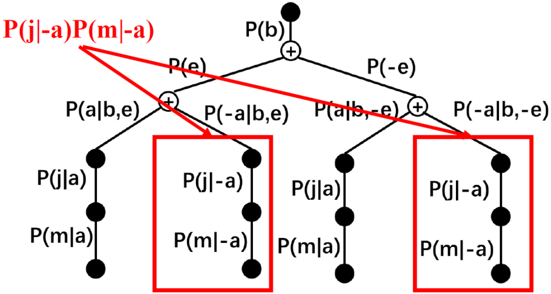


Figure 6: Inference by Enumeration

Elimination

Query: P(Burglary| JohnCalls= True, MaryCalls= True)

$$\mathbf{P}(B \mid j, m) = \alpha \underbrace{\mathbf{P}(B)}_{\mathbf{f}_1(B)} \sum_e \underbrace{P(e)}_{\mathbf{f}_2(E)} \boxed{\sum_a} \underbrace{\mathbf{P}(a \mid B, e)}_{\mathbf{f}_3(A,B,E)} \underbrace{P(j \mid a)}_{\mathbf{f}_4(A)} \underbrace{P(m \mid a)}_{\mathbf{f}_5(A)}$$

Step1: Make factors

Step2: Join factors &
eliminate hidden vars

Figure 7: Variable Elimination Process

Elimination

Query: P(Burglary| JohnCalls= True, MaryCalls= True)

- First, we sum out A from the product of $\mathbf{f}_3$, $\mathbf{f}_4$, and $\mathbf{f}_5$. This gives us a new 2×2 factor $\mathbf{f}_6(B, E)$ whose indices range over just B and E:

$$\begin{aligned} \mathbf{f}_6(B,E) &= \sum_a \mathbf{f}_3(A,B,E) \times \mathbf{f}_4(A) \times \mathbf{f}_5(A) \\ &= (\mathbf{f}_3(a,B,E) \times \mathbf{f}_4(a) \times \mathbf{f}_5(a)) + (\mathbf{f}_3(\neg a,B,E) \times \mathbf{f}_4(\neg a) \times \mathbf{f}_5(\neg a)) \,. \end{aligned}$$

Now we are left with the expression

$$\mathbf{P}(B \mid j, m) = \alpha\, \mathbf{f}_1(B) \times \sum_e \mathbf{f}_2(E) \times \mathbf{f}_6(B,E) \,.$$

- Next, we sum out E from the product of $\mathbf{f}_2$ and $\mathbf{f}_6$:

$$\begin{aligned} \mathbf{f}_7(B) &= \sum_e \mathbf{f}_2(E) \times \mathbf{f}_6(B,E) \\ &= \mathbf{f}_2(e) \times \mathbf{f}_6(B,e) + \mathbf{f}_2(\neg e) \times \mathbf{f}_6(B,\neg e) \,. \end{aligned}$$

This leaves the expression

$$\mathbf{P}(B \mid j, m) = \alpha\, \mathbf{f}_1(B) \times \mathbf{f}_7(B)$$

Figure 8: Variable Elimination Algorithm

Elimination

Query: P(Burglary| JohnCalls= True, MaryCalls= True)

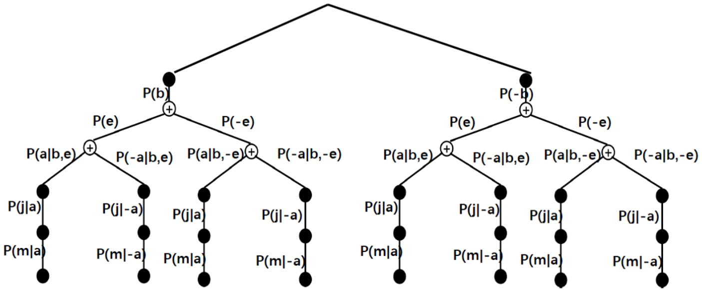


Figure 9: Inference by Elimination

Elimination

Query: P(Burglary| JohnCalls= True, MaryCalls= True)

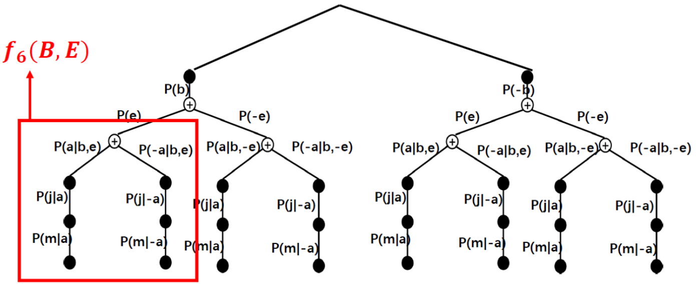


Figure 10: Inference by Elimination

Elimination

Query: P(Burglary| JohnCalls= True, MaryCalls= True)

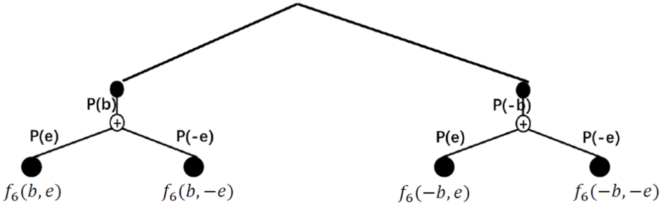


Figure 11: Inference by Elimination

Elimination

Query: P(Burglary| JohnCalls= True, MaryCalls= True)

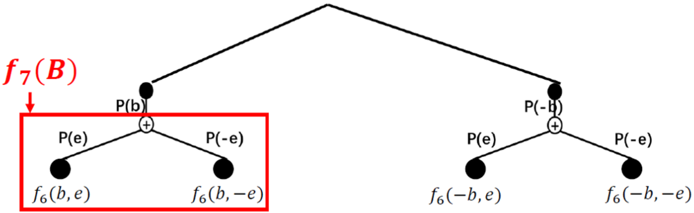


Figure 12: Inference by Elimination

Elimination

Query: P(Burglary| JohnCalls= True, MaryCalls= True)

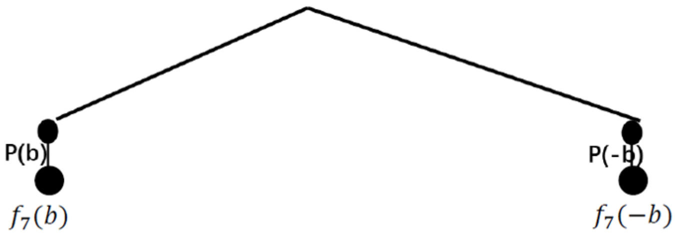


Figure 13: Inference by Elimination

Elimination

How to join factors?

e	$\neg e$
0.002	0.998

$\cdot$

	e		$\neg e$	
	a	$\neg a$	a	$\neg a$
b	0.95	0.05	0.94	0.06
$\neg b$	0.29	0.71	0.001	0.999

Figure 14: P(E),P(A|B,E)

Elimination

How to join factors?

e			$\neg e$		
	a	$\neg a$		a	$\neg a$
b	0.002	0.002	b	0.998	0.998
$\neg b$	0.002	0.002	$\neg b$	0.998	0.998

$\cdot$

e			$\neg e$		
	a	$\neg a$		a	$\neg a$
b	0.95	0.05	b	0.94	0.06
$\neg b$	0.29	0.71	$\neg b$	0.001	0.999

Figure 15: Expand

Elimination

How to join factors?

e			$\neg e$		
	a	$\neg a$		a	$\neg a$
b	0.0019	0.0001	b	0.938	0.05988
$\neg b$	5.8 E-4	1.42 E-4	$\neg b$	9.98 E-4	0.997

Figure 16: Point-wise Multiplication

Elimination

How to join factors?

	a	$\neg a$	
b	0.9399	0.05998	~ 1
$\neg b$	0.00158	0.9971	~ 1

Figure 17: Sum out

Summary

- Enumeration -
  - Step 1: Select the entries consistent with the evidence
  - Step 2: Sum out hidden vars to get joint of Query and evidence
  - Step 3: Normalize
- Elimination -
  - Step 1: Make factors
  - Step 2: Join all factors and eliminate all hidden vars
  - Step 3: Normalize

Summary

- Enumeration -
  - Step 1: Select the entries consistent with the evidence
  - Step 2: Sum out hidden vars to get joint of Query and evidence
  - Step 3: Normalize
- Elimination -
  - Step 1: Make factors
  - Step 2: Join all factors and eliminate all hidden vars
  - Step 3: Normalize

Problem Description

- Still, the burglary problem
- Input: Query and CPTs
- Output: Query Probability, and **Evidence Probability**
- Time limit: 1000ms
- Memory limit: 263MB
- Implement both methods:
  Enumeration and Elimination

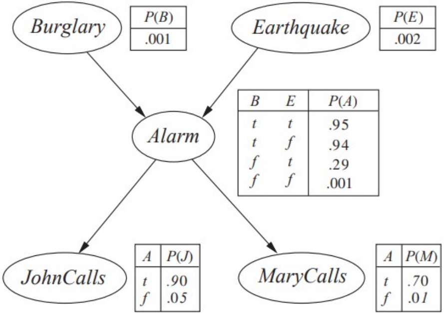


Figure 18: BN for burglary problem

Problem Description

- Still, the burglary problem
- Input: Query and CPTs
- Output: Query Probability, and **Evidence Probability**
- Time limit: 1000ms
- Memory limit: 263MB
- Implement both methods:
  Enumeration and Elimination

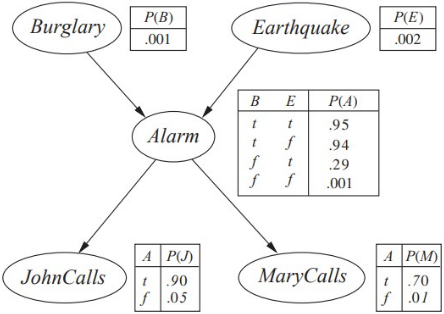


Figure 18: BN for burglary problem

Problem Description

- Still, the burglary problem
- Input: Query and CPTs
- Output: Query Probability, and **Evidence Probability**
- Time limit: 1000ms
- Memory limit: 263MB
- Implement both methods:
  Enumeration and Elimination

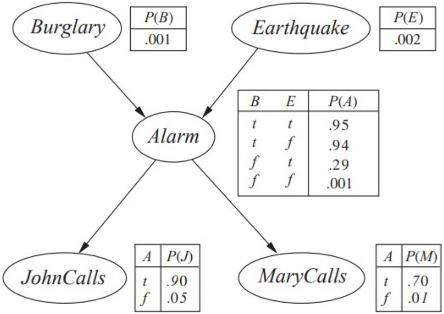


Figure 18: BN for burglary problem

In/Out Samples

```
P(Earthquake = -)
P(Burglary = + | John = +, Mary = +)
******
Burglary
0.001
***
Earthquake
0.002
***
Alarm | Burglary Earthquake
0.95 + +
0.94 + -
0.29 - +
0.001 - -
***
John | Alarm
0.9 +
0.05 -
***
Mary | Alarm
0.7 +
0.01 –
```

Figure 19: Input

```
probability by enumeration: 0.998
probability by elimination: 0.998
probability of evidence   : 1.000
**********
probability by enumeration: 0.284
probability by elimination: 0.284
probability of evidence   : 0.002
**********
```

Figure 20: Output

Functions to be implemented

- Enumeration algorithm:
  - Def enumeration_ask(X, e, bn):
  - Def enumeration_all(X, e, bn):
- Elimination algorithm:
  - Def elimination_ask(X, e, bn):
- You may change a few lines on joint_probability,conditional_probability and process_P_Query to get evidence probability.

Classes

Classes:

```
class BayesNet:                      used in building the BayesNet
    def __init__(self, node_specs=[]):
    def add(self, node_spec):
    def variable_node(self, var):
    def variable_values(self, vars):

class BayesNode:                     used in building the BayesNode
    """A conditional probability distribution for a boolean variable, P(X | parents).
Part of a BayesNet."""
    def __init__(self, x, parents, cpt):
    def p(self, value, event):
```

Classes:

```
class ProbDist:                      used in the computation for probability distribution
    def __init__(self, varname='?', freqs=None):
    def normalize(self):

class Factor:                        used in elimination algorithm
    def __init__(self, variables, cpt):
    def pointwise_product(self, other):
    def sum_out(self, var):
    def p(self, e):
    def normalize(self):
```

Pseudo Code

function ENUMERATION-ASK(X, **e**, bn) **returns** a distribution over X
  inputs: X, the query variable
    e, observed values for variables **E**
    bn, a Bayes net with variables $\{X\} \cup \mathbf{E} \cup \mathbf{Y}$ /* **Y** = *hidden variables* */

  $\mathbf{Q}(X) \leftarrow$ a distribution over X, initially empty
  for each value x_i of X **do**
    $\mathbf{Q}(x_i) \leftarrow$ ENUMERATE-ALL(bn.VARS, $\mathbf{e}_{x_i}$)
      where $\mathbf{e}_{x_i}$ is **e** extended with $X = x_i$
  return NORMALIZE($\mathbf{Q}(X)$)

---

function ENUMERATE-ALL($vars$, **e**) **returns** a real number
  if EMPTY?($vars$) **then return** 1.0
  $Y \leftarrow$ FIRST($vars$)
  if Y has value y in **e**
    then return $\boxed{P(y \mid parents(Y))}$ $\times$ ENUMERATE-ALL(REST($vars$), **e**)
    else return $\sum_y P(y \mid parents(Y)) \times$ ENUMERATE-ALL(REST($vars$), $\mathbf{e}_y$)
      where $\mathbf{e}_y$ is **e** extended with $Y = y$

You can use Ynode.p()

Figure 14.9 The enumeration algorithm for answering queries on Bayesian networks.

Figure 21: Enumeration

Pseudo Code

```
function ELIMINATION-ASK(X, e, bn) returns a distribution over X
  inputs: X, the query variable
          e, observed values for variables E
          bn, a Bayesian network specifying joint distribution P(X_1, ..., X_n)

  factors ← []
  for each var in ORDER(bn.VARS) do
      factors ← [MAKE-FACTOR(var, e)|factors]
      if var is a hidden variable then factors ← SUM-OUT(var, factors)
  return NORMALIZE(POINTWISE-PRODUCT(factors))
```

Actually, this is an “append” operation

How to express this?

Figure 22: Elimination

PJ4 Out

- Project 4 - Car to be released tonight
- **Offline Discussion** on Jan.8th
- **Due date** on Jan.9th

Good Luck