A Variable Elimination Algorithm for Belief Networks

This handout describes a general algorithm for computing $P(X \mid E)$ in a belief network, where X is a node and E is the evidence (those nodes known to be true or false). The example is computing $P(R \mid W)$ for the network in Figure 14.11, p. 510. The algorithm (named elim-bel) comes from Rina Dechter, Bucket elimination: A unifying framework for probabilistic inference, *Proceedings 1996 Conference on Uncertainty in Artificial Intelligence.*

Conversion of Conditional Probability Tables to λ Tables

Each conditional probability table is converted to a λ (lambda) table. Each possible value assignment to the node and its parents are associated with a λ value, which is initialized to the corresponding conditional probability.

$P(C) = 0.5$ $\Longrightarrow$

C	λ
T	$0.5 = P(C)$
F	$0.5 = P(\neg C)$

C	$P(S)$
T	$0.1 = P(S \mid C)$
F	$0.5 = P(S \mid \neg C)$

$\Longrightarrow$

C	S	λ
T	T	$0.1 = P(S \mid C)$
T	F	$0.9 = P(\neg S \mid C)$
F	T	$0.5 = P(S \mid \neg C)$
F	F	$0.5 = P(\neg S \mid \neg C)$

C	$P(R)$
T	$0.8 = P(R \mid C)$
F	$0.2 = P(R \mid \neg C)$

$\Longrightarrow$

C	R	λ
T	T	$0.8 = P(R \mid C)$
T	F	$0.2 = P(\neg R \mid C)$
F	T	$0.2 = P(R \mid \neg C)$
F	F	$0.8 = P(\neg R \mid \neg C)$

S	R	$P(W)$
T	T	$0.99 = P(W \mid S, R)$
T	F	$0.90 = P(W \mid S, \neg R)$
F	T	$0.90 = P(W \mid \neg S, R)$
F	F	$0.00 = P(W \mid \neg S, \neg R)$

$\Longrightarrow$

S	R	W	λ
T	T	T	$0.99 = P(W \mid S, R)$
T	T	F	$0.01 = P(\neg W \mid S, R)$
T	F	T	$0.90 = P(W \mid S, \neg R)$
T	F	F	$0.10 = P(\neg W \mid S, \neg R)$
F	T	T	$0.90 = P(W \mid \neg S, R)$
F	T	F	$0.10 = P(\neg W \mid \neg S, R)$
F	F	T	$0.00 = P(W \mid \neg S, \neg R)$
F	F	F	$1.00 = P(\neg W \mid \neg S, \neg R)$

Eliminating Evidence Variables

It is given that W is true. In each λ table that contains W, we delete all rows that have W as false, and eliminate the W column. [If W was given as false, then we would delete the rows that had W as true.]

S	R	W	λ
T	T	T	0.99
T	T	F	0.01
T	F	T	0.90
T	F	F	0.10
F	T	T	0.90
F	T	F	0.10
F	F	T	0.00
F	F	F	1.00

$\Longrightarrow$

S	R	λ
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.00

Eliminating Other Variables

Now we want to eliminate S and C, leaving only R, the variable that we are interested in. First, we eliminate S. [Order of elimination does not affect correctness.]

1. First find all the λ tables that reference S. This includes the S, R λ table just created by evidence elimination, and the C, S λ table generated earlier.

2. These tables also reference C and R. The new λ table is for these two variables.

3. To compute an entry in the new C, R λ table (say C false and R true), we add two values. One value is the product of the λ values that are consistent with C false, R true, and S true. The other value is the product of the λ values that are consistent with C false, R true, and S false.

C	S	λ
T	T	0.1
T	F	0.9
F	T	0.5
F	F	0.5

$\times$

S	R	λ
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.00

$\Longrightarrow$

C	R	λ
T	T	$0.909 = 0.1(0.99) + 0.9(0.90)$
T	F	$0.090 = 0.1(0.90) + 0.9(0.00)$
F	T	$0.945 = 0.5(0.99) + 0.5(0.90)$
F	F	$0.450 = 0.5(0.90) + 0.5(0.00)$

To eliminate C, there is one C λ table and two C, R λ tables to combine.

C	λ
T	0.5
F	0.5

$\times$

C	R	λ
T	T	0.8
T	F	0.2
F	T	0.2
F	F	0.8

$\times$

C	R	λ
T	T	0.909
T	F	0.090
F	T	0.945
F	F	0.450

$\Longrightarrow$

R	λ
T	$0.4581 = 0.5(0.8)(0.909) + 0.5(0.2)(0.945)$
F	$0.1890 = 0.5(0.2)(0.090) + 0.5(0.8)(0.450)$

Now, we have one λ table for R alone, and we can compute $P(R \mid W)$ by:

$$P(R \mid W) = \frac{0.4581}{0.4581 + 0.1890} \approx 0.7079$$

If we ended up with more than one λ table for R, we would compute one value p by multiplying all the λ values for R as true, compute a second value q by multiplying all the λ values for R as false, and then perform $p/(p + q)$.

Pseudocode for Elim-Bel

In this pseudocode, I assume the existence of the following subroutines. VARIABLES(a) returns the set of variables in a, which might be a belief network, a value assignment, or a lambda table. PARENTS(x) returns the set of the parents of the variable x. ASSIGNMENTS(S) returns all possible value assignments to a set of variables S; if S has n variables, then ASSIGNMENTS(S) returns 2^n assignments. $P(x \mid v)$ looks up the probability that x is true given value assignment v in the probability table for x. LAMBDA(v, *ltable*) returns the value in the lambda table *ltable* consistent with value assignment v; any variables in v not mentioned by *ltable* are ignored.

ELIM-BEL determines the probability that x is true given value assignment v. v does not assign a value to x. v can be a partial value assignment.

```
function ELIM-BEL(x, v)

  /* Create lambda tables. */
  ltables ← ∅
  for each y in VARIABLES(belief net) do
    S ← PARENTS(y)
    ltable ← new lambda table
    for each u in ASSIGNMENTS(S)
      LAMBDA(u ∪ {y = true}, ltable) ← P(y | u)
      LAMBDA(u ∪ {y = false}, ltable) ← 1 − P(y | u)
    end for
    Add ltable to ltables
  end for

  /* Eliminate evidence variables. */
  for each ltable in ltables do
    Remove all LAMBDA(u, ltable) values where u is inconsistent with v
    Remove VARIABLES(v) from ltable
  end for
```

```
/* Eliminate other variables. */
for each y in Variables(ltables) − {x} do
    ytables ← subset of ltables that refer to y
    S ← Variables(ytables) − {y}
    ltable ← new lambda table
    for each u in Assignments(S) do
        ytrue ← 1
        yfalse ← 1
        for each ytable in ytables do
            ytrue ← ytrue ∗ Lambda(u ∪ {y = true}, ytable)
            yfalse ← yfalse ∗ Lambda(u ∪ {y = false}, ytable)
        end for
        Lambda(u, ltable) ← ytrue + yfalse
    end for
    Remove ytables from ltables
    Add ltable to ltables
end for

/* Calculate conditional probability. */
xtrue ← 1
xfalse ← 1
for each ltable in ltables do
    xtrue ← xtrue ∗ Lambda({x = true}, ltable)
    xfalse ← xfalse ∗ Lambda({x = false}, ltable)
end for
return xtrue/(xtrue + xfalse)
```