Searching Possible Worlds

- Can we estimate the probabilities by only enumerating a few of the possible worlds?
- How can we enumerate just a few of the most probable possible worlds?
- Can we estimate the error in our estimates?
- For what cases does the error get small quickly?
- How fast does it converge to a small error?

Enumerating Possible Worlds

- Impose a total ordering on the variables.
- Ordering consistent with the ordering of the Bayesian network: parents of a node come before the node.
- Suppose the order is X_1, ..., X_n.
- A partial description is a tuple of values $\langle v_1, \cdots, v_j \rangle$ where $v_i \in vals(X_i)$.

Tuple $\langle v_1, \cdots, v_j \rangle$ corresponds to the variable assignment $X_1 = v_1 \wedge \cdots \wedge X_j = v_j$.

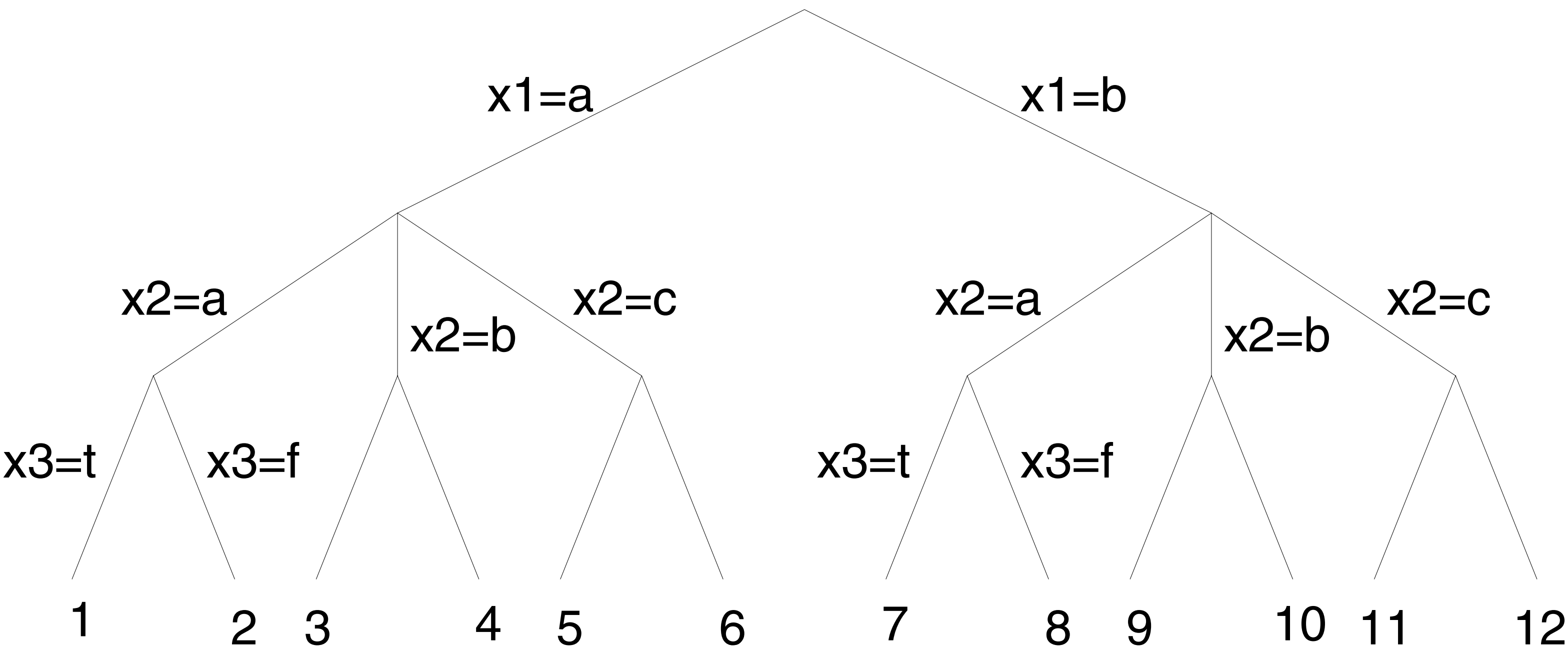
Search tree

The search tree has nodes labeled with partial descriptions, and is defined as follows:

- The root of the tree is labeled with the empty tuple $\langle\rangle$ (where $j = 0$).
- The children of node labeled with $\langle v_1, \cdots, v_j \rangle$ are the nodes $\langle v_1, \cdots, v_j, v \rangle$ for each $v \in vals(X_{j+1})$.
- The leaves of the tree are tuples of the form $\langle v_1, \cdots, v_n \rangle$. These correspond to possible worlds.

Example search tree

Suppose we have 3 variables, $x1$ with domain $\{a, b\}$, $x2$ with domain $\{a, b, c\}$, and $x3$ with domain $\{t, f\}$:



Basic Search Algorithm

$Q := \{\langle\rangle\};$

$W := \{\};$

While $Q \neq \{\}$ do

  choose and remove $\langle v_1, \cdots, v_j \rangle$ from Q;

  if $j = n$

    then $W := W \cup \{\langle v_1, \cdots, v_j \rangle\}$

    else $Q := Q \cup \{\langle v_1, \cdots, v_j, v \rangle : v \in vals(X_{j+1})\}$

Q is a priority queue of partial descriptions.

W is a set of generated possible worlds.

Properties of the Algorithm

- Each partial description can only be generated once. There is no need to check for multiple paths or loops in the search.
- The probability of a node is equal to the sum of the probabilities of the leaves that are descendents of the node.
- Algorithm is independent of the search strategy.

Estimating the Probabilities

Use W, at the start of an iteration of the while loop, as an approximation to the set of all possible worlds.

Let

$$P_W^g = \sum_{w \in W \wedge w \models g} P(w)$$

$$P_Q = \sum_{t \in Q} P(t)$$

Then

$$P_W^g \leq P(g) \leq P_W^g + P_Q$$

Posterior Probabilities

Given the definition of conditional probability:

$$P(g|obs) = \frac{P(g \wedge obs)}{P(obs)}$$

We estimate the probability of a conditional probability:

$$\frac{P_W^{g \wedge obs}}{P_W^{obs} + P_Q} \leq P(g|obs) \leq \frac{P_W^{g \wedge obs} + P_Q}{P_W^{obs} + P_Q}$$

If we choose the midpoint as an estimate:

$$\text{Error} \quad \leq \quad \frac{P_Q}{2(P_W^{obs} + P_Q)}$$

As the computation progresses, the probability mass in the queue P_Q approaches zero.

Complexity

- Approximating probabilities in Bayesian networks in NP-hard.
- The search algorithm is exponential in worst case.
- The algorithm is exponential on average (mass of each possible world is exponentially small in size of network).
- Niche algorithm: skewed distributions (conditional probabilities close to zero or one).

How well does it work for skewed distributions?

Skewed Distributions

Parents of X_i are $\Pi_{X_i} = \left\langle X_{i_1}, \cdots, X_{i_{k_i}} \right\rangle$,

For each value v_j,

$$(X_i = v_i | X_{i_1} = v_{i_1} \wedge \cdots \wedge X_{i_{k_i}} = v_{i_{k_i}}) \begin{cases} \geq p \approx 1 \\ \leq f \approx 0 \end{cases}$$

For each value $v_{i_1} \cdots v_{i_{k_i}}$ one value v_i of X_i is close to one (a *normality* value) and the others are close to zero (*faults*).

Error Convergence

Suppose there are n binary variables in the Bayesian network.

Suppose queue is implemented as a priority queue and we are about to choose the first k-fault ($k \ll n$) partial description.

There are at most $\mathbf{C}_k^n$ elements of the queue.

Each element of the queue has probability less than f^k.

$$\begin{aligned} P_Q &\leq \mathbf{C}_k^n f^k \\ &\leq \frac{n^k}{k!} f^k = \frac{(nf)^k}{k!}. \end{aligned}$$

Thus, we have convergence (as the computation proceeds and k increases) when $nf < 1$.

What is nf?

$$\begin{aligned} p^n &= (1-f)^n \\ &= 1 - nf + \mathbf{C}_2^n f^2 - \cdots \\ &\approx 1 - nf \end{aligned}$$

If nf is small then $nf \approx 1 - p^n$.

p^n is the prior probability of no faults.

$\delta = nf$ is the prior probability that there is some error in the system.

E.g., δ is small when diagnosing a system that works most of the time.

How fast is convergence?

How quickly can we get to the stage of choosing the first k-fault partial description from the queue?

There are at most $(n+1)^k$ ways of choosing k or fewer faults.

For each of these we go through the while loop at most n times. Each time we add and remove elements from the priority queue.

We can reach the stage where we are taking the first k-fault partial description off the queue in $O(n^{k+1} \log n)$ time.

Convergence of Priors

For fixed k, if $\delta < 1$ we can attain an accuracy of $\frac{\delta^k}{k!}$ in the estimate of prior probabilities in $O(n^{k+1}\log n)$ time.

We can attain

- an accuracy of δ in $O(n^2 \log n)$ time;
- an accuracy of $\frac{\delta^2}{2}$ in $O(n^3 \log n)$ time;
- an accuracy of $\frac{\delta^3}{6}$ in $O(n^4 \log n)$ time.

We can obtain an accuracy of ϵ in time

$$O\left(n^{1+\left(\frac{\log \epsilon}{\log \delta}\right)} \log n\right).$$

Refinements

We only need to consider the ancestors of the variables we are interested in. We can prune the rest before the search.

When computing $P(\alpha)$, we prune partial descriptions if it can be determined whether α is true or false in that partial description.

When computing $P(\bullet|OBS)$, we prune partial descriptions in which OBS is false.

We do not need to find the most likely possible worlds *in order* to sum the most likely worlds. One good search strategy is a depth-first depth-bounded search. We estimate the bound to produce the desired accuracy; increasing it when necessary.