Exact Inference in Bayes Nets – Pseudocode

The pseudocode and explanations for the inference algorithms in Russell and Norvig are confusing, so here we provide clearer versions.

The Enumeration Algorithm

function ENUMERATION-ASK(X, **e**, bn) **returns** a distribution over X
  inputs: X, the query variable
    e, observed values for some set of variables **E**
    bn, a Bayes net
  Q ← a distribution over X, where **Q**(x_i) is P(X=x_i)
  for each value x_i that X can have **do**
    Q(x_i) ← ENUMERATE-ALL(bn.VARS, $\mathbf{e}_{x_i}$), where $\mathbf{e}_{x_i}$ is the evidence **e** plus the assignment X=x_i
  return NORMALIZE(**Q**)

function ENUMERATE-ALL($vars$, **e**) **returns** a probability (a real number in [0,1])
  inputs: $vars$, a list of all the variables
    e, observed values for some set of variables **E**
  if EMPTY($vars$) **then return** 1.0
  Y ← FIRST($vars$)
  if Y is assigned a value (call it y) in **e then**
    return P(Y=y | values assigned to Y's parents in **e**) × ENUMERATE-ALL(REST($vars$), **e**)
  else
    return $\sum_{y_i}$ [P(Y=y_i | values assigned to Y's parents in **e**) × ENUMERATE-ALL(REST($vars$), $\mathbf{e}_{y_i}$)],
      where $\mathbf{e}_{y_i}$ is the evidence **e** plus the assignment Y=y_i

The enumeration algorithm is a simple, brute-force algorithm for computing the distribution of a variable in a Bayes net. The ENUMERATION-ASK function takes a variable X and returns a distribution over X, given some evidence **e**.

The evidence **e** is whatever values you already know about the variables in the Bayes net. Evidence simplifies your work because instead of having to consider those variables' whole distributions, you can assign them particular values, so they are no longer variables, they are constants. In the most general case, there is no evidence.

The ENUMERATION-ASK function has to compute a distribution over X, which, because X is a discrete variable, means computing the probability that X takes on each of its possible values (the values in its *domain*). The algorithm does this simply by looping through all of the possible values, and computing the probability for each one. Note that if there is no evidence, then it is literally just computing the probabilities P(X=x_i) for each x_i in X's domain. If there *is* evidence, then it is computing P(**e**, X=x_i) for each x_i in X's domain – that is, it is computing the probability that X has the given value (x_i) *and* the evidence is true – so in that case, we use the he multiplication law of probability, which says that P(X=x_i | **e**) = P(**e**, X=x_i) / P(**e**),
and the fact that P(**e**) is constant: once we have computed P(**e**, X=x_i) for all x_i, we can just normalize those values to get the correct distribution P(X | **e**).

The helper function ENUMERATE-ALL is uninformatively named, but basically what it is doing is

computing (something proportional to) P(**e**), the probability of the evidence **e** (actually, it is only computing it for the variables in **e** that are in *vars*, so in the top-level call where *vars* is all the variables in the Bayes net, it is computing it for **e**). The arguments are *vars* (a list of all the variables we have left to look at) and **e** (a list of all the variables that already have assigned values, along with those values). At each call of this function, we look at the next variable *Y* from *vars* (or in the base case, there are no variables left to look at, so we're done). There are two cases. In the first case, *Y* is already assigned in **e** to some value *y*, so P(**e**) is just P(*Y*=*y* | the rest of **e**) × P(the rest of **e**). In the second case, *Y* is not assigned, so we have to sum over all possible values y_i in *Y*'s domain.

An important fact that is not clear from the book's pseudocode is that the variables passed to ENUMERATE-ALL have to be in *topological order* – that is, a variable cannot come before its parents in the list. This is because to compute the probabilities for a variable, the values of its parents must already be known.

The Variable-Elimination Algorithm

function ELIMINATION-ASK(*X*, **e**, *bn*) **returns** a distribution over *X*
  inputs: *X*, the query variable
    e, observed values for some set of variables **E**
    bn, a Bayes net
  factors ← [for each variable *v* in *bn*.VARS, the CPT for *v* given **e**]
  for each *var* **in** *bn***.vars if** *var* **is not in e and** *var* **is not** *X* **do**
    relevant-factors ← [all factors that contain *var*]
    factors.remove(*relevant-factors*)
    factors.append(SUM-OUT(*var*, POINTWISE-PRODUCT(*relevant-factors*)))
  return NORMALIZE(POINTWISE-PRODUCT(*factors*))

The point of the variable-elimination algorithm is that it is more bottom-up than top-down: instead of figuring out the probabilities we need to compute and then computing all the other probabilities that each one depends on, we try to compute probabilities and then compute the other terms that depend on them, and repeatedly simplify the expression until we have something that is in terms of only the variable we're looking for.

The variable-elimination algorithm uses things called *factors*. A factor is basically a conditional probability table, except that the entries are not necessarily probabilities (but they would be if you normalized them). You can think of a factor as a matrix with a dimension for each variable, where Factor[VAL1][VAL2][...] is (proportional to) a probability such as P(VAR1=VAL1, VAR2=VAL2, ...); or you can think of it as a table with one row for each possible combination of assignments of values to the variables.

The ELIMINATION-ASK function, like ENUMERATION-ASK, takes a variable *X* and returns a distribution over *X*, given some evidence **e**. First it initializes the list of factors; prior to any simplification, this is just the conditional probability tables for each variable given the evidence **e**. Then, it *sums out* each variable from the list of factors. The summing-out process takes all the factors that depend on a given variable and replaces them with a single new factor that does not depend on that variable (by summing over all possible values of the variable). By the end of the loop, all the variables have been summed out except the query variable *X*, so then we can just multiply the factors together (see the textbook [page 526] for an explanation of the *pointwise product*) and normalize to get the distribution.