where the "x" operator is not ordinary matrix multiplication but instead the \textit{pointwise product} operation, to be described shortly.

The process of evaluation is a process of summing out variables (right to left) from pointwise products of factors to produce new factors, eventually yielding a factor that is the solution, i.e., the posterior distribution over the query variable. The steps are as follows:

- 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 x 2 factor $\mathbf{f}_6(B, E)$ whose indices range over just $B$ and $E$:

\[
\mathbf{f}_6(B, E) = (\mathbf{f}_3(a, B, E) \times \mathbf{f}_4(a) \times \mathbf{f}_5(a)) + (\mathbf{f}_3(a, B, E) \times \mathbf{f}_4(a) \times \mathbf{f}_5(a)).
\]

Now we are left with the expression

\[
\mathbf{P}(B_m) = \mathbf{f}_6(B, E) \times \mathbf{f}_2(E).
\]

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

\[
\mathbf{f}_7(B) = \sum \mathbf{f}_2(E) \times \mathbf{f}_6(B, E) = \mathbf{f}_2(e) \times \mathbf{f}_6(B, e) + \mathbf{f}_2(\bar{e}) \times \mathbf{f}_6(B, \bar{e}).
\]

This leaves the expression

\[
\mathbf{P}(B | j, n) = \mathbf{f}_7(B),
\]

which can be evaluated by taking the pointwise product and normalizing the result.

Examining this sequence, we see that two basic computational operations are required: pointwise product of a pair of factors, and summing out a variable from a product of factors. The next section describes each of these operations.

\section*{Operations on factors}

The \textit{pointwise} product of two factors $\mathbf{f}_1$ and $\mathbf{f}_2$ yields a new factor $\mathbf{f}$ whose variables are the \textit{union} of the variables in $\mathbf{f}_1$ and $\mathbf{f}_2$ and whose elements are given by the product of the corresponding elements in the two factors. Suppose the two factors have variables $Y_k$ in common. Then we have

\[
\mathbf{f}(X_1, \ldots, X_l, Y_k, Z_l) = X_j, Y_k, \mathbf{f}_2(Y_l, Y_l, Z_l).
\]

If all the variables are binary, then $\mathbf{f}_1$ and $\mathbf{f}_2$ have $2^{l_1}$ and $2^{l_2}$ entries, respectively, and the pointwise product has $2^{l_1 + l_2}$ entries. For example, given two factors $\mathbf{f}_1(A, B)$ and $\mathbf{f}_2(B, C)$, the \textit{pointwise product} $\mathbf{f}_1 \times \mathbf{f}_2 = \mathbf{f}_1(A, B, C)$ has $2^{1+1+1} = 8$ entries, as illustrated in Figure 14.10. Notice that the factor resulting from a \textit{pointwise} product can contain more variables than any of the factors being multiplied and that the size of a factor is exponential in the number of variables. This is where both space and time complexity arise in the variable elimination algorithm.
Section 14.4. Exact Inference in Bayesian Networks

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>f_1(A, B)</th>
<th>B</th>
<th>C</th>
<th>f_2(B, C)</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>f_3(A, B, C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>.3 x .2 = .06</td>
<td>F</td>
<td>F</td>
<td>.3 x .8 = .24</td>
<td>F</td>
<td>T</td>
<td>.7 x .6 = .42</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>.9 x .2 = .18</td>
<td>T</td>
<td>F</td>
<td>.9 x .8 = .72</td>
<td>T</td>
<td>T</td>
<td>.1 x .6 = .06</td>
<td>F</td>
</tr>
</tbody>
</table>

**Figure 14.10** illustrating pointwise multiplication: \( f_1(A, B) \times f_2(B, C) = f_3(A, B, C) \).

Summing out a variable from a product of factors is done by adding up the submatrices formed by fixing the variable to each of its values in turn. For example, to sum out \( A \) from \( f(A, B, C) \), we write

\[
f(B, C) = A, B, C) = f_3(a, B, C) + f_3(\neg a, B, C)
\]

The only trick is to notice that any factor that does not depend on the variable to be summed out can be moved outside the summation. For example, if we were to sum out \( E \) first in the burglary network, the relevant part of the expression would be

\[
\sum_{E} f_1(E) \times f_2(A, E) \times f_4(A) \times f_5(A) = f_3(A, B, C)
\]

Now the pointwise product inside the summation is computed, and the variable is summed out of the resulting matrix.

Notice that matrices are not multiplied until we need to sum out a variable from the accumulated product. At that point, we multiply just those matrices that include the variable to be summed out. Given functions for pointwise product and summing out, the variable elimination algorithm itself can be written quite simply, as shown in Figure 14.11.

**Variable ordering and variable relevance**

The algorithm in Figure 14.11 includes an unspecified ORDER function to choose an ordering for the variables. Every choice of ordering yields a valid algorithm, but different orderings cause different intermediate factors to be generated during the calculation. For example, in the calculation shown previously, we eliminated \( A \) before \( E \); if we did it the other way, the calculation becomes

\[
\sum_{A} f_2(A) \times f_4(A) \times f_5(A) \times \sum_{E} f_3(E) \times f_6(A, B, E)
\]

during which a new factor \( f_6(A, B) \) will be generated.

In general, the time and space requirements of variable elimination are dominated by the size of the largest factor constructed during the operation of the algorithm. This in turn