function **likelihood-weighting**($X, E, bn, N$) returns an estimate of $P(X \mid e)$
inputs: $X$, the query variable
$E$, observed values for variables $E$
$bn$, a Bayesian network specifying joint distribution $P(X_1, \ldots, X_n)$
$N$, the total number of samples to be generated
local variables: $W$, a vector of weighted counts for each value of $X$, initially zero
for $j = 1$ to $N$
do
$x, w \leftarrow \text{weighted-sample}(bn, e)$
$W[x] \leftarrow W[x] + w$ where $x$ is the value of $X$ in $x$
return $\text{normalize}(W)$

function **weighted-sample**($bn, e$) returns an event and a weight
w $\leftarrow 1$; $x \leftarrow \text{an event with n elements initialized from } e$
foreach variable $X_i$ in $X_1, \ldots, X_n$ do
  if $X_i$ is an evidence variable with value $x_i$ in $e$
    then $w \leftarrow w \times P(X_i = x_i \mid \text{parents}(X_i))$
  else $x_i \leftarrow \text{a random sample from } P(X_i \mid \text{parents}(X_i))$
return $x$, $w$

Figure 14.15 The likelihood-weighting algorithm for inference in Bayesian networks. In **weighted-sample**, each nonevidence variable is sampled according to the conditional distribution given the values already sampled for the variable’s parents, while a weight is accumulated based on the likelihood for each evidence variable.

with values $e$. We call the nonevidence variables $Z$ (including the query variable $X$). The algorithm samples each variable in $Z$ given its parent values:

$$S_{\text{WS}}(z, e) = \prod P(z_i \mid \text{parents}(z_i)). \tag{14.7}$$

Notice that $P_{\text{parents}}(Z_i)$ can include both nonevidence variables and evidence variables. Unlike the prior distribution $P(z)$, the distribution $S_{\text{WS}}$ pays some attention to the evidence: the sampled values for each $Z_i$ will be influenced by evidence among $Z_i$’s ancestors. For example, when sampling Sprinkler the algorithm pays attention to the evidence Cloudy = true in its parent variable. On the other hand, $S_{\text{US}}$ pays less attention to the evidence than does the true posterior distribution $P(z \mid e)$, because the sampled values for each $Z_i$ ignore evidence among $Z_i$’s non-ancestors. For example, when sampling Sprinkler and Rain the algorithm ignores the evidence in the child variable WetGrass = true; this means it will generate many samples with Sprinkler = false and Rain = false despite the fact that the evidence actually rules out this case.

Ideally, we would like to use a sampling distribution equal to the true posterior $P(z \mid e)$, to take all the evidence into account. This cannot be done efficiently, however. If it could, then we could approximate the desired probability to arbitrary accuracy with a polynomial number of samples. It can be shown that no such polynomial-time approximation scheme can exist.
The likelihood weight $w$ makes up for the difference between the actual and desired sampling distributions. The weight for a given sample $x$, composed from $z$ and $e$, is the product of the likelihoods for each evidence variable given its parents (some or all of which may be among the $Z_i$s):

$$w(z, e) = \prod_{i=1}^{m} P(e_i | \text{parents}(E_i)).$$

Multiplying Equations (14.7) and (14.8), we see that the weighted probability of a sample has the particularly convenient form:

$$P(z, e) = \frac{\sum_{y \in \mathcal{Y}} \sum_{e \in \mathcal{E}} P(y, e) w(z, y, e) \cdot \sum_{x \in \mathcal{X}} P(x, y, e) \cdot w(x, y, e) \cdot \text{likelihood-weighting}}{\sum_{y \in \mathcal{Y}} \sum_{e \in \mathcal{E}} P(y, e) \cdot \text{likelihood-weighting}}$$

$$= \frac{\sum_{y \in \mathcal{Y}} \sum_{e \in \mathcal{E}} P(y, e) \cdot \text{likelihood-weighting}}{\sum_{y \in \mathcal{Y}} \sum_{e \in \mathcal{E}} P(y, e) \cdot \text{likelihood-weighting}}$$

because the two products cover all the variables in the network, allowing us to use Equation (14.2) for the joint probability.

Now it is easy to show that likelihood weighting estimates are consistent. For any particular value $x$ of $X$, the estimated posterior probability can be calculated as follows:

$$\hat{P}(x | e) = \sum_{y \in \mathcal{Y}} \sum_{e \in \mathcal{E}} P(y, e) P(z, e)$$

$$= \sum_{y \in \mathcal{Y}} \sum_{e \in \mathcal{E}} P(y, e)$$

$$= \sum_{y \in \mathcal{Y}} \sum_{e \in \mathcal{E}} P(x, y, e)$$

$$= P(x)$$

Hence, likelihood weighting returns consistent estimates.

Because likelihood weighting uses all the samples generated, it can be much more efficient than rejection sampling. It will, however, suffer a degradation in performance as the number of evidence variables increases. This is because most samples will have very low weights and hence the weighted estimate will be dominated by the tiny fraction of samples that accord more than an infinitesimal likelihood to the evidence. The problem is exacerbated if the evidence variables occur late in the variable ordering, because then the nonevidence variables will have no evidence in their parents and ancestors to guide the generation of samples. This means the samples will be simulations that bear little resemblance to the reality suggested by the evidence.

### 14.5.2 Inference by Markov chain simulation

Markov chain Monte Carlo (MCMC) algorithms work quite differently from rejection sampling and likelihood weighting. Instead of generating each sample from scratch, MCMC algorithms generate each sample by making a random change to the preceding sample. It is therefore helpful to think of an MCMC algorithm as being in a particular current state specifying a value for every variable and generating a next state by making random changes to the