Algoritmos de Propagación II. Métodos de Monte Carlo

Antonio Salmerón

Dpto. Estadística y Matemática Aplicada
Universidad de Almería
Almería. 04120
e-mail: asc@stat.ualm.es

Resumen

Es conocido que la propagación exacta de probabilidades en redes bayesianas es un problema NP-duro [6]. Esto quiere decir que si la red es suficientemente complicada, puede que no podamos obtener resultados en un tiempo razonable. Surge entonces la necesidad de emplear métodos aproximados que, a cambio de perder la exactitud de los cálculos, ofrecen resultados en un tiempo menor. En este capítulo estudiamos un grupo de algoritmos aproximados de gran importancia: los basados en métodos de Monte Carlo.

1 Introducción

Los algoritmos aproximados surgieron con el propósito de resolver los casos peores para los métodos exactos en un tiempo más razonable, generalmente mediante técnicas de Monte Carlo, a cambio de la pérdida de la exactitud de los cálculos. La inferencia por métodos de simulación es también un problema NP-duro cuando se requiere una precisión determinada [7]; sin embargo, el conjunto de problemas resolubles es mayor que para los métodos exactos.

En este capítulo describiremos los métodos más importantes de propagación de probabilidades basados en simulación por Monte Carlo.

Comenzaremos planteando el problema en la sección 2. A continuación, en la sección 3, explicaremos el concepto de simulación y veremos cómo se aplica a la estimación de la distribución *a posteriori* de una red bayesiana. En la sección 4 estudiamos el funcionamiento de los métodos de propagación por Monte Carlo más sencillos: los que no utilizan precomputación. Terminaremos el capítulo con un acercamiento a métodos más sofisticados como el muestreo sistemático (sección 5) y muestreo por importancia basado en precomputación aproximada (sección 6).

2 Planteamiento del Problema

Supondremos durante este capítulo una red bayesiana definida sobre un conjunto de variables $X = \{X_1, \ldots, X_n\}$, cada una de ellas tomando valores en un conjunto finito U_i, $i = 1, \ldots, n$ y $N = \{1, \ldots, n\}$. Consideraremos también un conjunto de variables observadas X_E, tomando el valor $X_E = e$ con $e \in U_E$. Al valor e lo llamaremos *evidencia*.

El objetivo que nos proponemos es calcular la distribución *a posteriori* $p(x_k|e)$ para todo $x \in U_k$, correspondiente a cada variable X_k con $k \in N$. Al cálculo de esta probabilidad lo llamamos *propagación de probabilidades*. Esta probabilidad podría obtenerse mediante marginalización a partir de la distribución conjunta de la red,

$$p(x) = \prod_{i \in N} f_i(x^{\downarrow s(f_i)}), \quad \forall x \in U_N, \tag{1}$$

donde $s(f_i)$ representa el conjunto de índices de las variables para las que está definida la función f_i. En este caso, cada función f_i se corresponde con la distribución condicionada de la variable X_i a sus padres Π_{X_i}, es decir, $p(x_i|\pi_{X_i})$, con $x_i \in U_i$, $\pi_{X_i} \in U_{F(i)}$ y $s(f_i) = \{i\} \cup F(i)$, donde $F(i)$ es el conjunto de índices de las variables padre de X_i. Si existen variables observadas, $X_E = e$, entonces la distribución anterior quedará como

$$p(x, e) = \left(\prod_{i \in N} f_i(x^{\downarrow s(f_i)}) \right) \cdot \left(\prod_{j \in E} \delta_{e_j}(x^{\downarrow j}) \right), \quad \forall x \in U_N, \tag{2}$$

donde δ_{e_j} es una función que toma el valor 1 si x es consistente con la evidencia y 0 en otro caso:

$$\delta_{e_j}(y) = \begin{cases} 1 \text{ si } e_j = y, \\ 0 \text{ si } e_j \neq y. \end{cases} \tag{3}$$

Obsérvese que la probabilidad que queremos calcular es

$$p(x_k|e) = \frac{p(x_k, e)}{p(e)}, \tag{4}$$

y, dado que $p(e)$ es constante, ésta es proporcional a $p(x_k, e)$. Por lo tanto, podemos obtener la distribución *a posteriori* si calculamos para cada $x_k \in U_k$ el valor

$p(x_k, e)$ y normalizamos después. Podemos expresar $p(x_k, e)$ como la siguiente suma:

$$S = p(x_k, e) = \sum_{\substack{x \in U_N \\ x^{\downarrow E} = e \\ x^{\downarrow k} = x_k}} p(x) = \sum_{\substack{x \in U_N \\ x^{\downarrow k} = x_k}} p(x, e). \tag{5}$$

Pero suponemos que la distribución $p(x, e)$ es suficientemente complicada como para que los métodos exactos no sean aplicables, y, de igual manera, tampoco será posible calcular la suma anterior en un tiempo razonable. Por lo tanto, nos conformaremos con aplicar un método de simulación para obtener una estimación de la probabilidad que buscamos.

A continuación veremos en qué consiste la simulación y cómo puede ésta aplicarse a nuestro problema.

3 Simulación

Por *simulación* podemos entender la experimentación sobre un modelo de cierto sistema, de cara a predecir el comportamiento del mismo. Si el proceso de simulación conlleva el uso de números aleatorios, se la suele llamar también *simulación por Monte Carlo*. El objetivo de la simulación es extraer conclusiones sobre cierto sistema real sin necesidad de experimentar directamente sobre el sistema en cuestión.

Por ejemplo, supongamos que una empresa está considerando la apertura de un supermercado y nos encarga un informe para decidir el número de cajas registradoras que han de colocar. En este caso, el sistema real es el supermercado. Para decidir el número óptimo de cajas registradoras, podríamos observar el comportamiento del sistema, construyendo el supermercado, poniendo un cierto número de cajas y observando si éstas son suficientes o no. Es evidente que este método es extremadamente costoso. Podríamos recurrir entonces a realizar un modelo de simulación del supermercado y experimentar, en un ordenador, el funcionamiento del mismo. En este caso, sería sencillo hacer pruebas con distintos números de cajas registradoras.

En un modelo de este tipo, necesitamos generar aleatoriamente una población; en este caso, la de los usuarios del supermercado. Se sabe que dicha población puede modelizarse de acuerdo a ciertas distribuciones de probabilidad conocidas: por ejemplo, el número de personas que llegan a una caja registradora para ser atendidos sigue una distribución de Poisson.

Generar individuos de una población no es más que generar valores para una variable aleatoria que sigue una distribución dada. Una forma de hacer esto es mediante el *método de inversión*, fundamentado en el siguiente teorema:

Teorema 1. Sea X una variable aleatoria con función de distribución $F(x)$. Sea $F^{-1}(y)$ la función inversa de F, definida como

$$F^{-1}(y) = \inf\{x \mid F(x) \geq y\}, \quad 0 \leq y \leq 1. \tag{6}$$

Entonces, si U es una v.a. uniformemente distribuida en el intervalo $(0, 1)$, se cumple que la v.a. definida como $Z = F^{-1}(U)$ tiene como función de distribución $F(x)$. □

Este teorema nos dice la forma de generar valores para la variable X. Lo único que hay que hacer es generar un número aleatorio u (entre 0 y 1), y calcular el valor $F^{-1}(u)$. El resultado será un valor para la variable X. Existen numerosas formas de generar números aleatorios [16]. La mayoría de los lenguajes de programación de propósito general ofrecen mecanismos para generarlos. Con esto, el algoritmo para realizar esta tarea es como sigue:

1. Generar un número aleatorio u.
2. $X = F^{-1}(u)$.
3. Devolver X.

El método anterior es válido para variables tanto discretas como continuas. En las redes bayesianas, las variables que manejaremos serán siempre *discretas y finitas*, es decir, solo podrán tomar un número finito de valores. El siguiente ejemplo ilustra el funcionamiento del método de inversión para una variable discreta y finita.

Ejemplo 2. Sea una variable aleatoria X que puede tomar los valores x_1, x_2 y x_3 con probabilidad $P(X = x_1) = 0.2$, $P(X = x_2) = 0.3$ y $P(X = x_3) = 0.5$. La función de distribución F para la variable X puede verse en la figura 1. Supongamos que hemos generado un número aleatorio $u = 0.7$. Para obtener un valor para X a partir de u hemos de evaluar la función $F^{-1}(0.7)$. Obsérvese que en la gráfica 1 esto se puede hacer situando el punto 0.7 en el eje de ordenadas y viendo con qué punto del eje de abscisas se corresponde de acuerdo con el dibujo de F. Puede comprobarse que el valor 0.7 se corresponde con el valor x_3 de acuerdo con la fórmula (6). □

En general, un algoritmo para generar valores para una variable X con n posibles valores, $\{x_1, \ldots, x_n\}$ y con función de probabilidad $P(X = x_1) = p_1$, $P(X = x_2) = p_2, \ldots, P(X = x_n) = p_n$, es el siguiente:

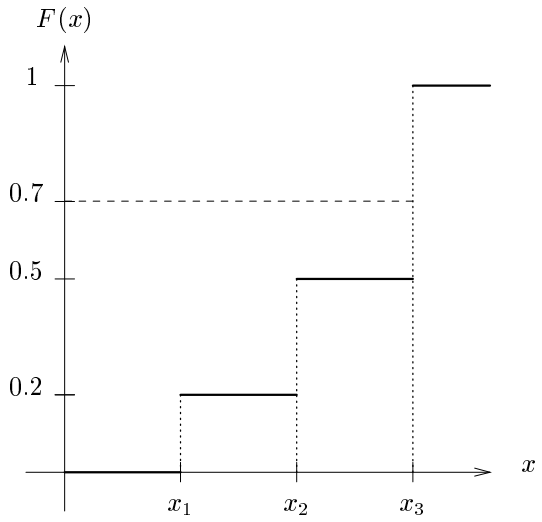

Figura1. Método de inversión.

1. Generar un número aleatorio u.
2. $P = p_1$.
3. $i = 1$.
4. Mientras $i <= n$ y $P < u$,
   (a) $i = i + 1$.
   (b) $P = P + p_i$.
5. $X = x_i$.
6. Devolver X.

3.1 Obtención de la probabilidad a posteriori mediante simulación

Una forma de obtener una estimación de la probabilidad de interés (fórmula (5)) mediante simulación, sería generando una serie de valores para las variables $X_1, \ldots, X_n$ mediante el método de inversión a partir de la distribución $p(x)$. A partir de la muestra generada, para un cierto x_k podríamos estimar su probabilidad como el cociente entre el número de veces en que X_k toma el valor x_k y el número total de individuos en la muestra generada.

Ejemplo 3. Consideremos la red de la figura 2, para la cual hay definida una distribución de probabilidad $p(x_1, x_2, x_3)$. Supongamos que las tres variables son

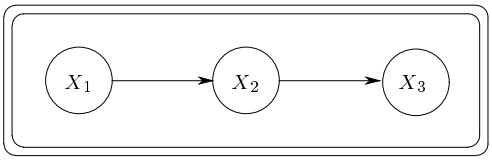

Figura2. Una red bayesiana con tres variables.

$f_1(0) = P(X_1 = 0) = 0.6$
$f_1(1) = P(X_1 = 1) = 0.4$
$f_2(0,0) = P(X_2 = 0 \mid X_1 = 0) = 0.2$
$f_2(0,1) = P(X_2 = 0 \mid X_1 = 1) = 0.5$
$f_2(1,0) = P(X_2 = 1 \mid X_1 = 0) = 0.8$
$f_2(1,1) = P(X_2 = 1 \mid X_1 = 1) = 0.5$
$f_3(0,0) = P(X_3 = 0 \mid X_2 = 0) = 0.2$
$f_3(0,1) = P(X_3 = 0 \mid X_2 = 1) = 0.3$
$f_3(1,0) = P(X_3 = 1 \mid X_2 = 0) = 0.8$
$f_3(1,1) = P(X_3 = 1 \mid X_2 = 1) = 0.7$

Tabla1. Probabilidades condicionadas para la red anterior.

binarias, es decir, pueden tomar los valores 0 ó 1, y que hemos generado la siguiente muestra a partir de la distribución p mediante el método de inversión:

$$(0,1,0), (0,1,1), (0,1,0), (1,1,1), (0,0,1), (1,1,1),$$

donde cada coordenada de cada tripleta representa los valores para X_1, X_2 y X_3 respectivamente. Si, por ejemplo, quisiéramos estimar la probabilidad *a posteriori* de la variable X_1, contaríamos los elementos de la muestra en los que X_1 toma el valor 0 y dividiríamos ese número entre el tamaño de la muestra, y análogamente para $X_1 = 1$. Es decir, estimaríamos dicha probabilidad como:

$$\hat{P}(X_1 = 0) = \frac{4}{6} = \frac{2}{3},$$

$$\hat{P}(X_1 = 1) = \frac{2}{6} = \frac{1}{3}.$$

□

En la práctica, no será posible utilizar la distribución p para generar la muestra, pues ésta será difícil de manejar y por lo tanto su inversa también lo será. Lo

x	$p(x)$
$(0,0,0)$	0.024
$(0,0,1)$	0.096
$(0,1,0)$	0.144
$(0,1,1)$	0.336
$(1,0,0)$	0.040
$(1,0,1)$	0.160
$(1,1,0)$	0.060
$(1,1,1)$	0.140

Tabla2. Probabilidad conjunta para la red anterior.

que se hace es utilizar una distribución modificada más sencilla para simular, y luego se asigna un *peso* o importancia a cada individuo de la muestra. El fundamento de este procedimiento consiste en que podemos expresar la suma (5) como sigue:

$$p(x_k, e) = \sum_{\substack{x \in U_N \\ x^{\downarrow k} = x_k}} p(x, e) = \sum_{\substack{x \in U_N \\ x^{\downarrow k} = x_k}} \frac{p(x, e)}{f^*(x)} f^*(x), \tag{7}$$

donde f^* es una función masa de probabilidad positiva en todos los puntos donde p es positiva. A f^* se le llama *función de muestreo*.

Si f^* se elige de forma que sea sencilla de manejar, podemos utilizarla para generar la muestra de las variables de la red, $\{x^{(j)}\}$, $j = 1, \ldots, m$, con m el tamaño de la muestra. A cada configuración $x^{(j)}$ le asignamos un *peso* o *importancia* w_j definido como

$$w_j = \frac{p(x^{(j)}, e)}{f^*(x^{(j)})}. \tag{8}$$

Entonces, puede estimarse cada probabilidad $p(x_k, e)$ como

$$\hat{p}(x_k, e) = \frac{1}{m} \sum_{j \in J} \frac{p(x^{(j)}, e)}{f^*(x^{(j)})} = \frac{1}{m} \sum_{j \in J} w_j, \tag{9}$$

donde $J \subseteq \{1, \ldots, m\}$ es un conjunto de índices tal que las configuraciones $x^{(j)}$, $j \in J$, verifican que ${x^{(j)}}^{\downarrow k} = x_k$ y ${x^{(j)}}^{\downarrow E} = e$. Es decir, se estima la probabilidad

de cada valor x_k como la media de los pesos de las configuraciones que componen la muestra, considerando que tienen peso cero aquellas configuraciones que no son consistentes con x_k. Puede comprobarse que $\hat{p}(x_k, e)$ es un estimador insesgado de $p(x_k, e)$ (ver [17]).

Para obtener la probabilidad a posteriori, $p(x_k|e)$, basta con normalizar los valores estimados de $p(x_k, e)$, lo que es equivalente a dividir entre la suma de todos los pesos.

Configuración ($x^{(j)}$)	Peso (w_j)
$(0,0,0)$	0.192
$(0,1,1)$	2.688
$(0,1,0)$	1.152
$(1,1,1)$	1.120
$(0,0,1)$	0.768
$(1,1,1)$	1.120

Tabla3. Pesos para la muestra del ejemplo.

Ejemplo 4. Supongamos que queremos estimar la probabilidad *a posteriori* de la variable X_1 de la red de la figura 2. Vamos a utilizar el método de los pesos. Imaginemos que hemos utilizado para obtener la muestra una distribución uniforme, es decir, $f^*(x) = 1/8$ para toda configuración x de las variables X_1, X_2 y X_3, y que hemos obtenido la misma muestra que en el ejemplo 3. La tabla 3 muestra los pesos de cada una de las configuraciones que forman la muestra. Procedemos como en el ejemplo 3, pero ahora sumando los pesos de las configuraciones favorables a cada uno de los valores de X_1. Obtenemos la siguiente estimación:

$$\hat{P}(X_1 = 0) = \frac{0.192 + 2.688 + 1.152 + 0.768}{6} = \frac{4.8}{6} = 0.8,$$
$$\hat{P}(X_1 = 1) = \frac{1.120 + 1.120}{6} = \frac{2.240}{6} \approx 0.37.$$

Normalizando, obtenemos la estimación $\hat{P}(X_1 = 0) = 0.68$ y $\hat{P}(X_1 = 1) = 0.32$. □

El proceso anterior queda reflejado en el siguiente algoritmo:

Algoritmo de simulación

1. Desde $i = 1$ hasta m,
   (a) Generar una configuración $x^{(i)}$ a partir de f^*.
   (b) Calcular
$$w_i = \frac{p(x^{(i)}, e)}{f^*(x^{(i)})}. \tag{10}$$
2. Para cada $x_k \in U_k$, $k = \{1, \ldots, n\}$,
   (a) Estimar $p(x_k, e)$ usando la fórmula (9).
3. Normalizar los valores $p(x_k, e)$ para obtener $p(x_k|e)$.

En este esquema, si todas las configuraciones que forman la muestra se elijen de forma independiente, diremos que realizamos un *muestreo por importancia* [16].

Siguiendo este esquema general de simulación, se han desarrollado diversos esquemas de propagación aproximada. La diferencia entre ellos radica en la forma en que se generan las configuraciones que componen la muestra y también en la distribución de muestreo que se emplea. Estudiaremos los siguientes métodos:

- Muestreo lógico probabilístico.
- Ponderación por verosimilitud.
- Simulación estocástica.
- Muestreo estratificado o sistemático.
- Muestreo por importancia basado en precomputación aproximada.

Los tres primeros no requieren ningún proceso de precomputación para calcular las distribuciones de muestreo antes de la simulación; por ello, los llamaremos *algoritmos de Monte Carlo sin precomputación*. El muestreo sistemático tampoco requiere de dicha precomputación, pero difiere de los anteriores en la forma de obtener las muestras. Por último, el método de muestreo por importancia basado en precomputación aproximada conlleva un cálculo inicial enfocado a mejorar la calidad de las funciones de muestreo.

4 Algoritmos de Propagación por Monte Carlo sin Precomputación

4.1 Muestreo lógico probabilístico

Este método, propuesto por Henrion [10], se engloba dentro de los llamados de *propagación hacia delante*. La idea de las técnicas de propagación hacia delante

consiste en elegir un orden *ancestral*[1] de las variables de la red y obtener una configuración para cada variable en secuencia, muestreando según la distribución condicionada de dicha variable dados sus padres en la red. A cada configuración de las variables obtenida se le asigna un peso que, al final del proceso de simulación, y normalizando, resulta en una estimación de la probabilidad a posteriori de cada variable.

En el método de muestreo lógico probabilístico destaca el hecho de que todos los pesos valen 0 ó 1, dependiendo de que la configuración obtenida sea coherente con las observaciones o no. Esto se debe a que la distribución de muestreo elegida coincide con la original, es decir, que para cada configuración $x^{(j)}$, el peso es:

$$\begin{aligned} w_j &= \frac{p(x^{(j)}, e)}{f^*(x^{(j)})} \\ &= \frac{\left(\prod_{i=1}^n f_i(x^{(j)\downarrow s(f_i)})\right) \cdot \left(\prod_{l\in E} \delta_{e_l}(x^{(j)\downarrow l})\right)}{\prod_{i=1}^n f_i(x^{(j)\downarrow s(f_i)})} \\ &= \prod_{l\in E} \delta_{e_l}(x^{(j)\downarrow l}). \end{aligned}$$

El algoritmo detallado es el siguiente, donde supondremos, sin pérdida de generalidad, que las variables siguen un orden ancestral:

Muestreo Lógico

1. Desde $j = 1$ hasta m (tamaño de la muestra),
   (a) Desde $i = 1$ hasta n,
      i. Obtener un valor $x_i \in U_i$ simulando de acuerdo a la distribución $p(x_i|\pi_{X_i})$, donde π_{X_i} es la configuración ya obtenida para los padres de X_i.
      ii. Si X_i es una variable observada y $x_i \neq e_i$, hacer $w_j = 0$ y volver al paso 1.
   (b) Hacer $w_j = 1$.
2. Para cada $x_k \in U_k$, $k = \{1, \ldots, n\}$,
   (a) Estimar $p(x_k, e)$ usando la fórmula (9).
3. Normalizar los valores $p(x_k, e)$ para obtener $p(x_k|e)$.

[1] Un orden de los nodos de un grafo se dice *ancestral* si cada nodo tiene una posición en dicho orden anterior a cualquier descendiente suyo.

Obsérvese que el problema de este algoritmo es que si la configuración obtenida no concuerda con las observaciones, la iteración no será válida (paso 1.(a).ii. del algoritmo). Este problema no se presenta si todas las observaciones se dan en nodos raíz, dado que en ese caso se puede instanciar cada variable al valor observado y no se simulan. Entonces, la primera variable a simular sería la primera que no estuviera observada, y su distribución de probabilidad estaría restringida a los valores de las variables observadas, luego no se obtendrían configuraciones contradictorias con las observaciones. De cualquier forma, lo normal es que las observaciones se presenten en cualquier parte de la red y no sólo en las raíces, por lo que este método no será aplicable en numerosas ocasiones.

El siguiente ejemplo ilustra el funcionamiento del algoritmo.

Ejemplo 5. Consideremos la red de la figura 2, en la que se ha observado que la variable X_3 toma el valor 1. El orden en que vamos a simular las variables es X_1, X_2, X_3. Veamos:

- **Simulación de** X_1. Para simular un valor para esta variable, generamos un número aleatorio. Supongamos que dicho número es $u = 0.3$. Aplicando el método de inversión a la distribución f_1 (ver tabla 1), obtenemos el valor $X_1 = 0$.
- **Simulación de** X_2. Generamos un nuevo número aleatorio, por ejemplo, $u = 0.7$. Ahora utilizamos la distribución f_2 instanciada al valor $X_1 = 0$ y por el método de inversión obtenemos el valor $X_2 = 1$.
- **Simulación de** X_3. Realizamos el mismo proceso utilizando f_3. Si el número aleatorio generado es $u = 0.4$, obtenemos $X_3 = 1$.

En definitiva, la configuración obtenida es $(0, 1, 1)$, que es consistente con la observación $X_3 = 1$. Si en la simulación de X_3 el número aleatorio hubiera sido, por ejemplo, $u = 0.1$, entonces el valor obtenido para X_3 hubiera sido el 0, lo que produciría la configuración $(0, 1, 0)$ que no es consistente con la evidencia, y, por lo tanto, la simulación no habría sido válida.

□

4.2 Método de ponderación por verosimilitud

El esquema de ponderación por verosimilitud fue desarrollado independientemente por Fung y Chang [9] y Shachter y Peot [18]. El objetivo que persigue es evitar la aparición de configuraciones inconsistentes con la evidencia. Para ello, las variables observadas no se simulan, sino que toman directamente el valor observado. Esto se consigue haciendo que la distribución de muestreo valga 1 para el valor de las variables observadas, de forma que siempre se obtenga ese valor al simular.

Es decir, la función de muestreo será igual al producto de las condicionadas de la red salvo para las variables observadas:

$$\forall x_i \in U_i, \quad f_i^*(x_i) = \begin{cases} p(x_i|\pi_{X_i}) \text{ si } i \notin E, \\ \delta_{e_i}(x_i) \quad \text{si } i \in E. \end{cases} \tag{11}$$

donde f_i^* es la distribución de muestreo para la variable X_i, y π_{X_i} es el valor simulado para las variables Π_{X_i}, con lo que

$$f^*(x) = \prod_{i=1}^{n} f_i^*(x_i) \quad \forall x = (x_i, \ldots, x_n) \in U_N. \tag{12}$$

Obsérvese que al usar las distribuciones condicionadas para simular, es necesario que el orden de simulación de las variables sea ancestral, al igual que en el muestreo lógico probabilístico.

Dado que todas las configuraciones son consistentes con la evidencia, el peso de una configuración $x = (x_1, \ldots, x_n)$ se puede calcular como

$$\begin{aligned} w &= \frac{p(x,e)}{f^*(x)} \\ &= \frac{\left(\prod_{i=1}^{n} f_i(x^{\downarrow s(f_i)})\right) \cdot \left(\prod_{i\in E} \delta_{e_i}(x_i)\right)}{\left(\prod_{i\notin E} f_i(x^{\downarrow s(f_i)})\right) \cdot \left(\prod_{i\in E} \delta_{e_i}(x_i)\right)} \\ &= \prod_{i\in E} f_i(x^{\downarrow s(f_i)}) \\ &= \prod_{i\in E} p(x_i|\pi_{X_i}). \end{aligned}$$

Es decir, el peso de cada configuración viene determinado por la probabilidad de la evidencia dado el resto de las variables, o, lo que es lo mismo, la verosimilitud de la evidencia.

Con esto, el algoritmo de ponderación por verosimilitud es muy similar al de Henrion, y puede enunciarse como sigue:

Ponderación por verosimilitud

1. Desde $j = 1$ hasta m (tamaño de la muestra),
   (a) Desde $i = 1$ hasta n,

   i. Si $i \notin E$, obtener un valor $x_i \in U_i$ simulando de acuerdo a la distribución $p(x_i|\pi_{X_i})$.
   (b) $w_j = \prod_{i \in E} p(x_i|\pi_{X_i})$.
2. Para cada $x_k \in U_k$, $k = \{1, \ldots, n\}$,
   (a) Estimar $p(x_k, e)$ usando la fórmula (9).
3. Normalizar los valores $p(x_k, e)$ para obtener $p(x_k|e)$.

Ejemplo 6. Para ilustrar este método, consideraremos de nuevo la red de la figura 2 y el orden de simulación X_1, X_2, X_3. Supondremos que se ha observado que la variable X_3 toma el valor 1. En estas condiciones, el proceso de simulación sería prácticamente igual que en el ejemplo 5, salvo que la variable X_3 no se simularía, sino que directamente tomaría el valor 1. Luego, si los números aleatorios son los mismos que en el ejemplo 5, la configuración obtenida es $(0, 1, 1)$, y el peso será

$$w = P(X_3 = 1|X_1 = 0, X_2 = 1) = P(X_3 = 1|X_2 = 1) = 0.7.$$

□

El funcionamiento de este método es bueno salvo cuando se presentan probabilidades muy próximas a cero. En este caso es posible que gran parte de las configuraciones simuladas tengan peso nulo [12].

4.3 Método de simulación estocástica

Este método, también llamado de *simulación directa*, fue propuesto por Pearl [15]. Las diferencias más destacadas respecto al algoritmo de ponderación por verosimilitud son:

1. En este caso, las variables no han de simularse en ningún orden en especial.
2. En lugar de simular usando la distribución condicionada de cada variable, se usa la distribución de cada variable condicionada a su envolvente de Markov en la red[2].

El algoritmo detallado queda como sigue.

Simulación estocástica

1. Hacer que todos los nodos de la red a uno de sus posibles valores con probabilidad no nula.

[2] La *envolvente de Markov* de una variable en una red bayesiana es el conjunto de los padres, hijos y padres de los hijos de dicha variable.

2. Para cada variable no observada X_i, $i \in \{1, \ldots, n\}$, hacer $h_i(x_i) = 0$ para todo $x_i \in U_i$.
3. Desde $j = 1$ hasta m (tamaño de la muestra),
   (a) Para cada variable X_i, $i \in \{1, \ldots, n\}$,
      i. Calcular $P(X_i|W_{X_i})$, donde W_{X_i} denota la envolvente de Markov de la variable X_i, de la siguiente manera:

$$p(x_i|w_{X_i}) = \alpha \cdot p(x_i|\pi_{X_i}) \prod_{j \in H(i)} p(x_j|\pi_{X_j}) \quad \forall x_i \in U_i. \tag{13}$$

      donde α es una constante de normalización, $H(i)$ es el conjunto de índices de las variables hijo de X_i y w_{X_i} es la configuración actual de la envolvente de Markov de la variable X_i.
      ii. Simular un valor $x_i^{(j)} \in U_i$ para X_i según la distribución $p(x_i|w_{X_i})$.
      iii. Actualizar h_i según una de la dos siguientes expresiones:

$$h_i(x_i^{(j)}) = h(x_i^{(j)}) + 1,$$
$$h_i(x_i^{(j)}) = h(x_i^{(j)}) + p(x_i^{(j)}|w_{X_i}).$$

4. Normalizar los h_i, $i = 1, \ldots, n$. Cada h_i resultante es la distribución a posteriori de la variable X_i.

Este método presenta dos problemas principales. Por un lado, puede ser difícil encontrar una configuración inicial para las variables de la red que tenga probabilidad positiva. Jensen, Kong y Kjærulff [13] proponen usar inicialmente una técnica de muestreo hacia delante para encontrar la configuración inicial.

Por otro lado, cada configuración depende de la generada inmediatamente antes (ver fórmula (13)). Por eso, puede darse el caso de que, una vez alcanzada una configuración, ésta se repita un gran número de veces, debido a que las dependencias entre las variables sean "casi" funcionales, es decir, las distribuciones generadas en la fórmula 13 tengan valores muy próximos a 0 o a 1. La convergencia de este método hacia la distribución exacta está asegurada, cuando todas las probabilidades son estrictamente positivas, por resultados de la teoría de los procesos de Markov [3,8], pero ésta puede alcanzarse muy lentamente por la razón dicha anteriormente. En el caso de tener probabilidades nulas, puede que no se de la convergencia. El siguiente ejemplo puede aclarar la situación:

Ejemplo 7. Sea una red bayesiana con dos variables binarias conectadas de la forma $X_1 \to X_2$, con $U_1 = \{x_1, \bar{x}_1\}$, $U_2 = \{x_2, \bar{x}_2\}$ y tales que $p(x_2|x_1) = p(\bar{x}_2|\bar{x}_1) = \delta \simeq 1$. Supongamos que $p(x_1) = 0.5$ y que $X_1 = x_1$, entonces

$p(x_2|w_{X_2}) = p(x_2|x_1) = \delta$. Si en una simulación obtenemos $X_2 = x_2$, en la próxima simulación la distribución usada para simular X_1 será

$$\begin{aligned} p(x_1|w_{X_1}) &= p(x_1|x_2) \\ &= \alpha \cdot p(x_2|x_1) \cdot p(x_1) \\ &= \alpha \cdot 0.5 \cdot \delta = \delta, \end{aligned}$$

dado que, por la regla de Bayes, $\delta = 1/P(X_2 = x_2)$, y

$$\begin{aligned} p(x_2) &= p(x_2|x_1) \cdot p(x_1) + p(x_2|\bar{x}_1) \cdot p(\bar{x}_1) \\ &= \delta \cdot 0.5 + (1 - \delta) \cdot 0.5 = 0.5. \end{aligned}$$

Si continuamos así, obtendremos la configuración (x_1, x_2) con probabilidad muy próxima a 1, y, en el momento en que una de las dos variables cambiara de valor, la otra también lo haría, repitiéndose entonces muchas veces la configuración $(\bar{x}_1, \bar{x}_2)$. Obsérvese, por lo tanto, que la configuración que se obtenga en una simulación puede depender fuertemente de la obtenida en la simulación anterior. □

Tratando de resolver este problema, surgió el denominado *muestreo de Gibbs por bloques*, desarrollado por Jensen, Kong y Kjærulff [13]. Estos autores se dan cuanta de que los problemas de la simulación estocástica se deben a la dependencia entre las configuraciones de una muestra, en el sentido de que, en cada momento, sólo se cambia el valor de una variable. Esto no ocurre en el muestreo hacia delante, en el que todas las variables pueden cambiar de valor de una configuración a la siguiente en una muestra.

El muestreo de Gibbs por bloques es un sofisticado método que se basa en buscar un compromiso entre dependencia entre las configuraciones y coste computacional, partiendo de los dos casos extremos:

1. Simular una sola variable cada vez dada su envolvente de Markov es computacionalmente simple, pero las muestras pueden ser muy dependientes.
2. Simular todas las variables a la vez hace que las muestras sean independientes, pero el coste computacional puede ser intratable.

El método consiste en dividir las variables de la red en una serie de grupos de forma que todas las variables en un mismo grupo se simulan a la vez. Cuanto más grande sea cada grupo, menor será la dependencia entre las muestras, pero mayor será la complejidad de calcular la distribución conjunta que ha de usarse para simular las variables del grupo a la vez.

5 Muestreo Estratificado o Sistemático

La simulación estratificada es una técnica muy conocida en estadística [16] que conduce el proceso de simulación de forma que se eviten las muestras raras o desequilibradas. La idea básica consiste en dividir el espacio muestral en diversas regiones o estratos y elegir en cada uno de ellos un número óptimo de muestras. Esto produce una mejor representación del espacio muestral que la que resulta de las muestras aleatorias, y se pueden obtener mejores estimaciones para un tamaño determinado de la muestra o bien reducir el tamaño de la muestra para obtener la precisión requerida.

Los primeros algoritmos de propagación basados en muestreo estratificado fueron desarrollados por Bouckaert [1] y Bouckaert, Castillo y Gutiérrez [2]. La idea es considerar el espacio de todas las posibles configuraciones de las variables de la red, y asignar a cada una de ellas un subintervalo de $[0,1]$, de tal forma que las configuraciones más probables tengan asignado un subintervalo más amplio. Entonces, se selecciona un grupo de configuraciones muestreando sobre el intervalo $[0,1]$. El procedimiento es el siguiente:

Sea un conjunto de variables $X = \{X_1, \ldots, X_n\}$, donde cada variable X_i toma valores en $U_i = \{0, 1, \ldots, r_i - 1\}$. Sean f_i, $i = 1, \ldots, n$ las distribuciones condicionadas para cada variable dados sus padres en la red. En estas condiciones, podemos calcular todas las posibles configuraciones de las variables y su probabilidad de ocurrencia. El método de muestreo estratificado requiere que las configuraciones estén ordenadas, por ejemplo, según el siguiente criterio [2]:

Definición 8. Sean $x = (x_1, x_2, \ldots, x_n)$ e $y = (y_1, y_2, \ldots, y_n)$ dos configuraciones de la variable n-dimensional X. Se dice que x *precede* a y, y se denota $x < y$ si:

$$x < y \iff \exists k \ t.q. \ \forall j < k \quad x_j = y_j \text{ y } x_k < y_k. \tag{14}$$

□

En base al orden definido en (14), se construye una tabla que representa el espacio muestral. Esta tabla se usa para obtener las configuraciones en el proceso de muestreo. Por ejemplo, sea $X = \{X_1, X_2, X_3\}$ el conjunto de variables de la red de la figura 2, cuyas probabilidades *a priori* se encuentran en la tabla 1. En la tabla 4 pueden verse las configuraciones ordenadas y su probabilidad de ocurrencia, probabilidad acumulada e intervalo asociado. Cada configuración x^i

Configuración	Probabilidad	Prob. acumulada	Intervalo asociado
(0,0,0)	0.024	0.024	(0.000,0.024)
(0,0,1)	0.096	0.120	(0.024,0.120)
(0,1,0)	0.144	0.264	(0.120,0.264)
(0,1,1)	0.336	0.600	(0.264,0.600)
(1,0,0)	0.040	0.640	(0.600,0.640)
(1,0,1)	0.160	0.800	(0.640,0.800)
(1,1,0)	0.060	0.860	(0.800,0.860)
(1,1,1)	0.140	1.000	(0.860,1.000)

Tabla4. Probabilidades e intervalos para las configuraciones ordenadas.

tiene asociado un intervalo $I_i = [l(i), h(i)) \subseteq [0,1]$ cuyos límites se calculan a partir de las probabilidades acumuladas de acuerdo a las siguientes expresiones:

$$\begin{aligned} l(i) &= \sum_{j<i} \prod_{r=1}^{n} f_r^*(x^{j\downarrow r}), \\ h(i) &= l(i) + \prod_{r=1}^{n} f_r^*(x^{i\downarrow r}). \end{aligned} \tag{15}$$

donde x^j es la j-ésima configuración de la variable n-dimensional X y f_r^*, $r = 1, \ldots, n$, son las distribuciones de muestreo. La figura 3 muestra la división del intervalo $[0,1]$ para la red de la figura 2.

Para obtener una muestra de tamaño m, se generan m números en el intervalo $[0,1]$, y se comprueba qué configuración se corresponde con cada número generado, de acuerdo a la partición de la región (figura 3). A continuación, se pondera cada configuración de acuerdo con la distribución usada para calcular los intervalos (f_r^*) y la distribución original. Los m números no son aleatorios, sino que se calculan de forma determinista [2] de la siguiente manera,

$$k_i = \frac{i - 0.5}{m}, \quad i = 1, 2, \ldots, m.$$

El hecho de que los números "aleatorios" sean generados aquí de forma determinista, motiva el nombre de *muestreo sistemático* para este método.

El siguiente ejemplo explica cómo obtener una muestra a partir de una secuencia de números dada.

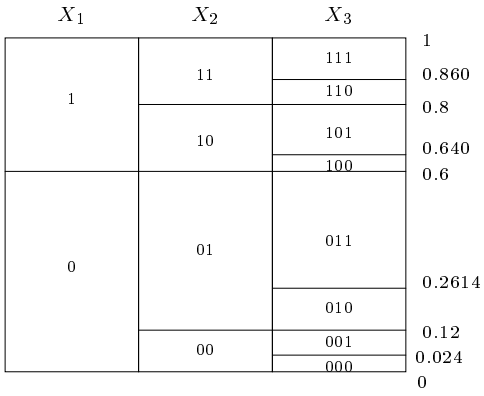

Figura3. Configuraciones y sus probabilidades acumuladas.

Ejemplo 9. Considérese la red mostrada en la figura 2. Generando cuatro números $k_i = (i - 0.5)/4$, $i = 1, \ldots, 4$, obtenemos la secuencia,

$$(0.125, 0.375, 0.625, 0.875).$$

Ahora, para cada número, buscamos en el diagrama representado en la figura 3 las configuraciones correspondientes. Éstas son:

Número	*Configuración* (x_1, x_2, x_3)
0.125	$(0, 1, 0)$
0.375	$(0, 1, 1)$
0.625	$(1, 0, 0)$
0.875	$(1, 1, 1)$

□

Se puede apreciar que cuando m aumenta, la frecuencia relativa de cada configuración converge a su valor de probabilidad. El hecho de que no se utilicen números aleatorios hace que este algoritmo tenga un carácter más numérico que de simulación. Nótese que las funciones de muestreo pueden ser cualesquiera, luego dependiendo de las que se usen, se obtendrán distintos resultados. Bouckaert, Castillo y Gutiérrez [2] usan las mismas funciones que en el algoritmo de ponderación por verosimilitud. Una descripción detallada del algoritmo correspondiente a este método puede encontrarse en [5].

6 Muestreo por Importancia basado en Precomputación Aproximada

La decisión más importante a la hora de diseñar un algoritmo de muestreo por importancia es la elección de la distribución de muestreo: ésta debería ser tan similar a la distribución original como sea posible. En el caso particular de una red causal, la distribución original viene dada como el producto de una serie de distribuciones condicionadas y un conjunto de observaciones. Los algoritmos conocidos de muestreo por importancia [4,9,18] usan las funciones originales (distribuciones condicionadas u observaciones) para aproximar la distribución producto. Es decir, estos métodos usan exclusivamente *información local* sobre cada variable a la hora de simularla.

En esta sección veremos un nuevo enfoque para obtener las distribuciones de muestreo. La idea es usar no sólo las condicionadas y las observaciones originales, sino toda la información concerniente a cada variable. Esto es, a la hora de simular valores para una variable, usar todas las funciones de las que disponemos. Éste es el caso ideal, pero si la red es suficientemente complicada, este proceso puede ser inviable; en concreto, la complejidad de este procedimiento sería la misma que la de la propagación exacta, y eso es precisamente lo que queremos evitar. En resumen, el problema es que el coste de la combinación de todas las funciones definidas para una variable puede ser demasiado alto.

El esquema que describimos en esta sección tiene dos fases principales: *precomputación aproximada* y *simulación*. La primera de ellas se basa en realizar una eliminación de variables para encontrar una aproximación de las funciones de muestreo. En la fase de simulación, se utilizan estas funciones obtenidas para generar configuraciones de las variables que serán ponderadas como en los métodos anteriores.

Por *eliminación de una variable* entendemos el proceso de combinación de todas las funciones definidas para dicha variable y la posterior marginalización de la función obtenida sobre el resto de variables. A saber, hay dos formas de realizar la eliminación de una variable X_i: exacta y aproximada.

Exacta

1. Combinar todas las funciones que están definidas para la variable X_i, obteniendo como resultado una función h_i.
2. Eliminar X_i de la combinación, h_i, marginalizando el resultado a $s(h_i) - \{i\}$.
3. Añadir el resultado de la marginalización a H.
4. Eliminar de H todas las funciones que se combinaron para obtener h_i.

Si es posible repetir este proceso para todas las variables, en cada paso se obtiene una distribución de muestreo proporcional a $p(x, e)$. En realidad, el proceso es como un algoritmo de propagación exacta [19], y se verifica el siguiente teorema:

Teorema 10. Supongamos que hemos realizado una eliminación exacta; entonces,

- Si h_n es la función obtenida al eliminar X_n entonces, para todo $x \in U_{\sigma(n)}$, $h_n(x)$ es proporcional a $p(x|e)$.
- Si h_i es la función obtenida al eliminar X_i $(i < n)$, $\Sigma(i) = \{i+1, \ldots, n\}$, y $x_0 \in U_{\Sigma(i) \cap s(h_i)}$, entonces, la restricción de h_i a x_0, h_i' es proporcional a la probabilidad $p(.|e, x_0)$.

□

Las dos propiedades del teorema anterior nos permiten simular un valor $x \in U_N$ con probabilidad igual a $p(x|e)$. Lo que tenemos que hacer es simular valores para las variables en el orden $X_n, \ldots, X_1$. Para obtener un valor para una variable X_i, muestreamos a partir de la función h_i, realizando primero la restricción de esta función a los valores x_0 obtenidos para las variables simuladas previamente $(X_{\Sigma(i)})$ y normalizando después.

En algunos casos, el tamaño[3] de h_i puede ser tan grande que su cálculo sea inviable. En este caso, la eliminación de las variables habrá de hacerse de forma aproximada. Pueden definirse numerosos criterios de aproximación, pero siempre dentro del siguiente esquema:

Aproximado

1. Sea $H(i) = \{h \in H \mid i \in s(h)\}$, el conjunto de funciones definidas para la variable X_i. Eliminar $H(i)$ de H.
2. Transformar $H(i)$ mediante combinación. Para ello, repetir el siguiente proceso:
   (a) Tomar $R \subseteq H(i)$.
   (b) Combinar todas las funciones contenidas en R, es decir, calcular $f = \prod_{h \in R} h$.
   (c) Añadir el resultado de la combinación, f, a $H(i)$.
   (d) Eliminar R de $H(i)$.

[3] Se define el *tamaño* de una función h como el producto del número de casos de todas las variables para las cuales h está definida.

3. Calcular $H^+(i)$ a partir de $H(i)$ eliminando X_i en todas las funciones pertenecientes a $H(i)$.
4. Añadir $H^+(i)$ a H.

Este procedimiento coincide con el exacto si en el segundo paso se combinan todas las funciones contenidas en $H(i)$. La idea del procedimiento aproximado es combinar funciones mientras no se sobrepase cierto umbral de tamaño. Es decir, la forma de elegir los $R \subseteq H(i)$ dependerá del tamaño del resultado de combinar las funciones que lo formen. Una propiedad importante de esta aproximación de cara a su validez para obtener funciones del muestreo por importancia es que no se añaden nuevos ceros. Esto queda garantizado por el siguiente lema:

Lema 11. Sean $H(i)$ y $H^+(i)$ como en el algoritmo aproximado. Sea $x \in U_N$. Se verifica que

$$h(x^{\downarrow s(h)}) > 0 \ \ \forall h \in H(i) \Rightarrow h(x^{\downarrow s(h)}) > 0 \ \ \forall h \in H^+(i).$$

□

Una vez realizado el proceso de eliminación, el siguiente paso es obtener configuraciones de las variables X_N. El proceso para simular un valor para una variable X_i según el algoritmo aproximado es el siguiente: si x_0 es la configuración obtenida para las variables $X_{\Sigma(i)}$, entonces

Simula(X_i,$H(i)$)

1. Sea $H(i)$ el conjunto calculado en el paso 2 del procedimiento de eliminación aproximada.
2. Restringir cada función en $H(i)$ a x_0. Combinar todas las funciones en $H(i)$, obteniendo una nueva función h_i' definida sobre $U_{\sigma(i)}$.
3. Si $N(h_i')$ es la normalización de h_i', obtener un valor x_i para X_i siguiendo la distribución de probabilidad $N(h_i')$.
4. Devolver el valor x_i.

Habiendo definido una forma de calcular las distribuciones de muestreo y de simular valores para las variables, se puede diseñar un algoritmo de propagación sin más que seguir el esquema general de la sección 3.1.

Referencias

1. Bouckaert, R.R., A stratified simulation scheme for inference in Bayesian belief networks. En: *Uncertainty in Artificial Intelligence, Proceedings of the Tenth Conference*, pp. 110-117, 1994.
2. Bouckaert, R.R., E. Castillo, J.M. Gutiérrez, A modified simulation scheme for inference in Bayesian networks. *International Journal of Approximate Reasoning*, 14, pp. 55-80, 1996.
3. Breiman, L., *Probability*. Addison Wesley. 1968.
4. Cano, J.E., L.D. Hernández, S. Moral, Importance sampling algorithms for the propagation of probabilities in belief networks. *International Journal of Approximate Reasoning*, 15, pp. 77-92, 1996.
5. Castillo, E., J.M. Gutiérrez, A.S. Hadi, *Sistemas expertos y modelos de redes probabilísticas*. Monografías de la Academia de Ingeniería. 1996.
6. Cooper, G.F., The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42, pp. 393-405, 1990.
7. Dagum, P., M. Luby, Approximating probabilistic inference in Bayesian networks is NP-hard. *Artificial Intelligence*, 60, pp. 141-153, 1993.
8. Feller, W., *Introducción a la teoría de probabilidades y sus aplicaciones*. Limusa. 1973.
9. Fung, R., K.C. Chang, Weighting and integrating evidence for stochastic simulation in Bayesian networks. En: *Uncertainty in Artificial Intelligence* 5. (M. Henrion, R.D. Shachter, L.N. Kanal, J.F. Lemmer, eds.) North-Holland (Amsterdam), pp. 209-220. 1990.
10. Henrion, M., Propagating uncertainty by logic sampling in Bayes networks. En: *Uncertainty in Artificial Intelligence*, 2 (J.F. Lemmer, L.N. Kanal, eds.) North-Holland (Amsterdam), pp. 317-324, 1988.
11. Hernández, L.D., S. Moral, A. Salmerón, Importance sampling algorithms for belief networks based on approximate computation. *Proceedings of the Sixth International Conference IPMU'96*. Vol. II, pp. 859-864, 1996.
12. Hernández, L.D., S. Moral, A. Salmerón, A Monte Carlo algorithm for probabilistic propagation based on importance sampling and stratified simulation techniques. *International Journal of Approximate Reasoning*. 1998. En prensa.
13. Jensen, C.S., A. Kong, U. Kjærulff, Blocking Gibbs sampling in very large probabilistic expert systems. *International Journal of Human-Computer Studies*, 42, pp. 647-666, 1995.
14. Jensen, F.V., *An introduction to Bayesian networks*. UCL Press. 1996.
15. Pearl, J., Evidential reasoning using stochastic simulation of causal models. *Artificial Intelligence*, 32, pp. 247-257, 1987.
16. Rubinstein, R.Y., *Simulation and the Monte Carlo Method*. Wiley (New York), 1981.
17. Salmerón, A., *Precomputación en grafos de dependencias mediante algoritmos aproximados*. Tesis Doctoral. Universidad de Granada. 1998.
18. Shachter, R.D., M.A. Peot, Simulation approaches to general probabilistic inference on belief networks. En: *Uncertainty in Artificial Intelligence* 5, (M. Henrion, R.D.

Shachter, L.N. Kanal, J.F. Lemmer, eds.) North Holland (Amsterdam), pp. 221-231. 1990.
19. Shafer, G., P.P. Shenoy, Probability propagation. *Annals of Mathematical and Artificial Intelligence*, 2, pp. 327-351. 1990.