INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY
CAMPUS MONTERREY
División de Graduados en Electrónica, Computación, Información y Comunicaciones
Dirección de Programas de Posgrado en Electrónica, Computación, Información y Comunicaciones

T E S I S

Maestría en Ciencias en Sistemas Inteligentes

Un ambiente de trabajo para sistemas de información basados en razonamiento Bayesiano

por

Armando Robles Pompa

Monterrey, N.L., 12 de Diciembre de 2003

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY
CAMPUS MONTERREY
División de Graduados en Electrónica, Computación, Información y Comunicaciones
Dirección de Programas de Posgrado en Electrónica, Computación, Información y Comunicaciones

Los miembros del comité de tesis recomendamos que la presente tesis de Armando Robles Pompa sea aceptada como requisito parcial para obtener el grado académico de Maestro en Ciencias en:

Sistemas Inteligentes

Comité de Tesis:

Dr. Francisco Javier Cantú Ortiz
Asesor de la tesis

Dr. Luis Eduardo Garza Castañón
Sinodal

Dr. Rubén Morales Menéndez
Sinodal

Dr. David Garza Salazar
Director del Programa de Graduados
en Electrónica, Computación,
Información y Comunicaciones

Diciembre de 2003

Un ambiente de trabajo para sistemas de información basados en razonamiento Bayesiano

por

Armando Robles Pompa

T E S I S

Presentada a la División de Electrónica, Computación, Información y Comunicaciones
Este trabajo es requisito parcial para obtener el grado académico de Maestro en ciencias en Sistemas Inteligentes.

Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Monterrey

Monterrey, N.L., 12 de Diciembre de 2003

A Cely, que vive en mi corazón.

Reconocimientos

Agradezco a mi asesor, el Dr. Francisco Cantú, por su guía y apoyo en la realización de esta tesis.
Agradezco a mi sinodal, el Dr. Luis Garza, quien me indujo al tema Bayesiano.
Agradezco en especial, a mi sinodal, el Dr. Rubén Morales, por su apoyo incondicional y sus críticas constructivas y enriquecedoras.

Armando Robles Pompa.

Instituto Tecnológico y de Estudios Superiores de Monterrey
Diciembre de 2003

Índice

1. Introducción **1**

I Marco Teórico **3**

2. Minería de Datos **3**
2.1. Minería de datos predictiva . . . 3
2.1.1. Exploración inicial . . . 4
2.1.2. Construcción del Modelo y Validación . . . 4
2.1.3. Aplicación del modelo . . . 4
2.2. Common Warehouse Metamodel (CWM) Specification[12] . . . 4

3. Aprendizaje Bayesiano **6**
3.1. Red Bayesiana . . . 6
3.2. Notación para variables, valores y distribuciones . . . 6
3.3. Enfoque Bayesiano . . . 7
3.3.1. Muestreo Binomial . . . 8
3.3.2. Utilidad de la distribución Beta . . . 8
3.4. Muestreo multinomial con distribución Dirichlet . . . 9
3.5. Aprendizaje de probabilidades en una red Bayesiana . . . 10
3.5.1. Aprendizaje con modelo de distribución multinomial . . . 11
3.5.2. Cálculo de la distribución posterior . . . 11
3.5.3. Actualización de parámetros en forma independiente . . . 12
3.5.4. Aprendizaje con datos incompletos . . . 13

4. Inferencia Bayesiana **14**
4.1. Conceptos básicos usados en el algoritmo de inferencia PPAC . . . 14
4.1.1. Potenciales . . . 14
4.1.2. Operaciones sobre potenciales . . . 14
4.1.3. Distribuciones de probabilidad . . . 15
4.1.4. Probabilidad condicional . . . 15
4.1.5. Representación de una red Bayesiana . . . 15
4.2. Estructura secundaria . . . 16
4.3. Construcción del árbol conjunto a partir de la red Bayesiana . . . 17
4.3.1. Construcción del grafo moral . . . 17
4.3.2. Triangulación del grafo moral . . . 19
4.3.3. Identificación de cliques . . . 19
4.3.4. Construcción del árbol conjunto óptimo . . . 20
4.4. Proceso de inferencia . . . 21
4.4.1. Inicialización . . . 22
4.4.2. Propagación global . . . 23
4.4.3. Marginalización . . . 24
4.5. Manejo de evidencia . . . 24
4.5.1. Observaciones y verosimilitud . . . 24
4.5.2. Inferencia con observaciones . . . 24
4.5.3. Inicialización con observaciones . . . 24

4.5.4. Entrada de observaciones 25
4.5.5. Normalización 25

II Planteamiento del problema y solución propuesta 27

5. Planteamiento del problema 27
5.1. Situación actual 27
5.2. Enfoque de esta tesis 28
5.3. Hipótesis de investigación 28
5.4. Prueba de hipótesis 28

6. Solución propuesta: BIframework 29
6.1. Capa 1. Sistema de información 29
6.2. Capa 2: BIserver (Middleware) 30
6.2.1. Componentes de BIserver 31
6.2.2. Flujo de datos 34
6.2.3. Mensajes SOAP del sistema de información a BIserver 34
6.2.4. Funciones de BIserver 34
6.3. Capa 3: Implementación de las técnicas de razonamiento Bayesiano 35
6.3.1. Estructuras para retorno a BIserver 35
6.3.2. Protocolo TagValor 35
6.4. Programación y prueba del algoritmo de Aprendizaje 36
6.5. Programación y prueba del algoritmo de Inferencia 36
6.6. Caso ejemplo de razonamiento Bayesiano 36
6.7. Composición del ambiente de trabajo 36
6.7.1. Integración del sistema de información en la arquitectura 36
6.7.2. Integración de los algoritmos Bayesianos en la arquitectura 37

III Caso de estudio 38

7. Caso de estudio 38
7.1. Análisis y diseño del sistema 38
7.2. Alcance del sistema de aprendizaje e inferencia construido 38
7.3. Red Bayesiana generada 41

8. Resultados obtenidos 43

IV Conclusiones 45

9. Conclusiones 45

10. Trabajo futuro 45

V Apéndices 46

A. Uso de funciones de BIserver **46**

B. Detalle de integración del sistema de información en BIframework **48**

C. Detalle de integración de razonamiento Bayesiano en BIframework **49**

D. Caso ejemplo de razonamiento Bayesiano con los algoritmos implementados en BIframework **53**

D.0.1. Estructura de la red Bayesiana 53
D.0.2. Generación de datos de entrada 53
D.0.3. Probabilidades aprendidas 54
D.0.4. Grafo moral 54
D.0.5. Grafo triangulado y cliques 55
D.0.6. Estructura de árbol conjunto 56
D.0.7. Cálculo de la inferencia 57
D.1. Iniciación de cliques y sepsets 64
D.2. Propagación global 67
D.2.1. Colecta evidencia 67
D.2.2. Distribuye evidencia 71

VI Referencias **77**

VII Curriculum Vitae **79**

Índice de figuras

1. Red Bayesiana mostrando probabilidades aprendidas 7
2. Ejemplo gráfico de la distribución multinomial 11
3. Red Bayesiana representada por un GAD 18
4. Grafo moral 18
5. Grafo triangulado y tabla que muestra el orden de eliminación de vértices 20
6. El árbol conjunto 21
7. Diagrama de bloques del PPAC sin evidencia 22
8. Diagrama de bloques del PPAC con observaciones 25
9. Arquitectura de BIframework 29
10. Diagrama de casos de uso. 39
11. Conocimiento del dominio: diagrama de clases (ontología). 39
12. Descomposición y diagrama de control de la tarea de asignación 40
13. Reglas de negocio para inferencias del método de asignación 40
14. Diagrama de actividad para método de asignación con algoritmo de aprendizaje. . . 42
15. Red Bayesiana para aprendizaje e inferencia 43
16. Red Bayesiana mostrando probabilidades aprendidas 54
17. Grafo triangulado 55
18. El árbol conjunto 57
19. Árbol conjunto inconsistente 59
20. Árbol conjunto consistente 62

Resumen

En esta tesis se implementan dos algoritmos de razonamiento Bayesiano, y se propone el ambiente de trabajo **BIframework** (Business Intelligence framework) para utilizarlos en sistemas de información que operan en línea. Se plantea el diseño conceptual de BIframework y se construye un prototipo mostrando su funcionalidad en un caso de estudio, donde se utiliza mediante BIframework el razonamiento Bayesiano para la asignación de asesores en la operación en línea de un sistema de servicio a clientes en una empresa de software.

1. Introducción

Hay una gran cantidad de sistemas de información aplicados a la administración y operación de diversos tipos de negocios, tales como hoteles, hospitales y comercio de todo tipo. Estos sistemas están enfocados a resolver el problema de operación del negocio. Con la evolución tecnológica en computación en la última década, se han ido construyendo sistemas de información que apuntan cada vez más hacia la integración de información y a su uso para la toma de decisiones. Por otro lado, al contar con equipos de cómputo cada vez más poderosos, las técnicas de inteligencia artificial cada vez son mas atractivas para apoyar la toma de decisiones utilizando el alto volumen de transacciones producidos en operación de dichos negocios. El área de inteligencia artificial que trata sobre diversas técnicas para descubrir patrones y secretos escondidos en los altos volúmenes de operación, es la Minería de Datos (MD).

Actualmente, la minería de datos se centra en lo que se puede llamar el método consultivo: una persona utiliza diversas técnicas de Inteligencia Artificial (IA) para analizar la información y descubrir los secretos contenidos en ella, produce un modelo y lo utiliza posteriormente en forma de un programa computacional, que implemente cierta lógica que represente algún comportamiento conveniente encontrado en el proceso de MD, para lograr apoyar y mejorar el proceso de decisiones en los negocios.

BIframework se propone como un ambiente de trabajo, que permita que un sistema de información que opera en línea, pueda utilizar aprendizaje Bayesiano para aprender relaciones que existan en los datos producidos por la operación del sistema, y que pueda utilizar dicho conocimiento para apoyar procesos de toma de decisiones en el sistema, utilizando inferencia Bayesiana.

BIframework, promoverá que los especialistas de Inteligencia Artificial se enfoquen a la elaboración de algoritmos que implementen diversas técnicas, para que los sistemas de información que operan en línea, puedan acceder a ellos sin su presencia. El sistema de información tendrá la iniciativa de uso, y por lo tanto, utilizará las técnicas de IA en forma autónoma, retroalimentando los procesos propios de la operación del negocio.

En los últimos años, la IA ha ganado terreno en aplicaciones reales. Esto se debe a que sus herramientas permiten crear sistemas que tomen decisiones parecidas a las que haría una persona ante una situación determinada. Una de estas herramientas es la del aprendizaje, que permite que un agente tenga la capacidad de aprender y adaptarse para poder tomar una decisión ante casos desconocidos que se le presenten. Con esto un agente puede funcionar de forma autónoma y con una mínima intervención humana.

En este proyecto de tesis se implementarán dos técnicas de IA, una de aprendizaje Bayesiano y una de inferencia Bayesiana (al conjunto de ambas técnicas se le llama razonamiento Bayesiano), y se utilizarán en el contexto de BIframework en un sistema de información que opera en línea.

En este documento se llama **modelo** a la representación del aprendizaje por parte de la técnica Bayesiana, de tal forma que el proceso de inferencia Bayesiana **usa el modelo** aprendido.

Se desea que el sistema de información utilice razonamiento Bayesiano en forma **autónoma**, es decir, que la aplicación del uso de la técnica se haga por iniciativa del sistema de información, decidiendo cuándo y sobre que datos se debe construir el modelo, **interactuando** con los algoritmos de razonamiento Bayesiano para aplicar el modelo a diversos casos que se presenten en la operación en línea del sistema.

Para efectos de esta tesis, la ejecución **autónoma** de un modelo, es la que se realiza sin intervención humana, y la operación de un modelo con **interacción** es la que permite la comunicación del modelo con el sistema de información de operación.

Este proyecto de tesis tiene dos contribuciones:

1. Un ambiente de trabajo (BIframework) que permita construir sistemas de información basados en razonamiento Bayesiano.
2. El diseño conceptual y metodología de integración de técnicas de IA en aplicaciones de sistemas de información.

Parte I
Marco Teórico

En esta sección se explican los términos, conceptos, técnicas y algoritmos que son esenciales para construir BIframework.

Los temas que se revisarán son los siguientes:

- Minería de Datos
- Aprendizaje Bayesiano
- Inferencia Bayesiana

2. Minería de Datos

La Minería de Datos (MD), o descubrimiento de conocimiento en bases de datos (KDD por sus siglas en inglés), es una de las áreas de aplicación en computación de más rápido crecimiento, ya que ofrece herramientas útiles para analizar grandes bases de datos que se utilizan en los negocios, ciencias, e industria[15]. La tecnología de MD analiza las bases de datos para extraer información e identificar patrones que se pueden trasladar en aplicaciones útiles.

Se puede ver a la MD, como un proceso analítico diseñado para explorar datos, usualmente grandes cantidades de datos de negocios o relacionados al mercado, en busca de relaciones sistemáticas entre variables, para construir un modelo que pueda ser aplicado a nuevos subconjuntos de datos[16].

Para establecer claramente el contexto en el que se ubica BIframework respecto a las aplicaciones de MD, se explican brevemente dos aspectos:

- Etapas de la Minería de Datos predictiva.
- Especificación del repositorio común de Metamodelos (Common Warehouse Metamodel Specification) de la OMG (Object Management Group).

2.1. Minería de datos predictiva

Se puede entender claramente el concepto de MD analizando las etapas de la MD predictiva, que es la que por ahora mayor uso tiene para aplicaciones de negocios.

El proceso de MD predictiva consiste en tres etapas,[13]:

1. Exploración inicial.
2. Construcción del modelo y validación.
3. Aplicación del modelo a nuevos datos para generar predicciones.

2.1.1. Exploración inicial

Esta etapa usualmente inicia con la preparación de datos, que puede incluir el limpiarlos, hacerles transformaciones, seleccionar subconjuntos de registros y - en caso de un conjunto de datos con demasiadas variables - hacer algunas operaciones preliminares de selección de características para conseguir un número de variables en un rango 'manejable' (dependiendo de los métodos estadísticos que sean considerados). Entonces, dependiendo de la naturaleza del problema analítico, esta etapa del proceso de MD puede incluir desde la simple elección de variables predictivas directas para un modelo de regresión, a un análisis de exploración muy elaborado, utilizando una amplia variedad de métodos apoyados en gráficas de representación y métodos estadísticos para identificar las variables más relevantes, y determinar la complejidad y naturaleza general de los modelos que se pueden considerar en la siguiente etapa.

2.1.2. Construcción del Modelo y Validación

En esta etapa se consideran varios modelos para encontrar el mejor, basado en su capacidad predictiva, es decir, explicando la variabilidad en cuestión y la producción de resultados estables en las muestras. Este camino suena como una operación simple, pero en realidad, algunas veces involucra un proceso muy elaborado. Hay una variedad de técnicas desarrolladas para alcanzar esta meta, algunas de las cuales están basadas en lo que se llama 'evaluación competitiva de modelos', esto es, la aplicación de varios modelos al mismo conjunto de datos, comparando su desempeño para seleccionar el mejor.

2.1.3. Aplicación del modelo

En esta etapa se utiliza el modelo que se seleccionó como el mejor en la etapa previa, para aplicarlo a los nuevos datos para generar las predicciones o estimaciones esperadas como salida.

El concepto de MD se está convirtiendo cada vez mas popular como la herramienta de exploración de información de negocios, y se espera que revele estructuras de conocimiento que puedan guiar decisiones en condiciones de incertidumbre. Recientemente, ha habido un interés creciente en el desarrollo de nuevas técnicas analíticas, específicamente diseñadas para cumplir con los aspectos más relevantes a la MD para negocios, por ejemplo, árboles de clasificación, algoritmos de aprendizaje e inferencia, redes neuronales, etc., pero, la minería de datos está aún basada en los principios conceptuales de la estadística, incluyendo el análisis de datos exploratorio tradicional y modelación, y comparte con ella algunos componentes, sus metodologías generales y técnicas específicas.

En la sección 3, se explica el enfoque Bayesiano a la probabilidad, que permite construir herramientas de Minería de datos para aprendizaje e inferencia, que es un tipo especial de predicción.

2.2. Common Warehouse Metamodel (CWM) Specification[12]

Con el propósito de facilitar la aplicación de técnicas de MD, y en general de Inteligencia de Negocios, la OMG (Object Management Group), ha desarrollado varios estándares para la

industria que establecen marcos de referencia sobre los cuales se pueden desarrollar y utilizar herramientas para análisis y elaboración de modelos de MD y en general, de aplicaciones de Inteligencia de Negocios.

El propósito principal de CWM[12] es habilitar el fácil intercambio de Metadatos acerca de información de Bases de Datos Corporativas y de Inteligencia de Negocios, entre herramientas de 'warehouse', plataformas de 'warehouse' y repositorios de metadatos de 'warehouses' en ambientes distribuidos y heterogéneos. CWM se basa en tres estándares de la OMG, claves de la industria:

1. UML - Unified Modelling Language.
2. MOF - Meta Object Facility, un estándar para repositorios de metamodelación y metadatos.
3. XMI-XML Metadata Interchange. Estándar para intercambio de metadatos.

Estos tres estándares, forman el centro de la arquitectura del repositorio de metadatos de la OMG.

El estándar UML define un lenguaje de modelación orientado a objetos, que es soportado por varias herramientas de diseño gráfico. El estándar MOF define una marco de trabajo extensible para definir modelos para metadatos, y proporciona herramientas con interfaces programáticas para almacenar y accesar metadatos en un repositorio. El estándar XMI permite que los metadatos sean intercambiados como 'cadenas' o archivos con un formato estándar basado en XML.

La arquitectura completa ofrece un amplio rango de alternativas de implementación para los desarrolladores de herramientas, repositorios y marcos de trabajo (frameworks) de objetos. En particular, XMI-XML, baja la barrera para entrar al estándar de metadatos de OMG.

Idealmente, las organizaciones deberían poder definir los metadatos empresariales en formatos estándar, tales como XML, y exportar esas definiciones a varias aplicaciones, incluyendo Data Warehouses, soluciones inter-empresariales (EAI) y nuevas aplicaciones. Lo único que habilitará a las múltiples herramientas a hablarse en el mismo 'idioma' es el estándar. Con el CWM la OMG ha creado un conjunto de interfases estándar que habilitarán a las diferentes soluciones de e-business a 'platicarse' entre ellas y prevenir quedar 'encerradas' en soluciones propietarias[14].

En BIframework se utiliza el enfoque de la **OMG** de la siguiente forma:

- Toda la comunicación entre el sistema de información y los algoritmos de razonamiento Bayesiano, se realiza en XML con su protocolo envolvente SOAP (ver la sección 6.2.1).
- Se propone UML como técnica de especificación de análisis y diseño utilizando la metodología de CommonKADS [9], para que el sistema de información identifique y ubique claramente las partes del sistema que utilizarán razonamiento Bayesiano.
- A futuro, BIFramework se ajustará al estándar de Meta Object Facility.

3. Aprendizaje Bayesiano

Debido a la gran importancia de los conceptos de aprendizaje e inferencia en el campo de la IA y específicamente en aplicaciones de MD, se explicarán en forma detallada, primero, los elementos que describen el enfoque Bayesiano a la probabilidad, posteriormente la justificación matemática para la construcción de un algoritmo que aprende probabilidades mediante el "conteo" de parámetros en una distribución Dirichlet, y finalmente, cómo se construye, a partir de una representación específica del conocimiento de un dominio utilizando una red Bayesiana, un algoritmo de Inferencia, que toma las relaciones y probabilidades aprendidas mediante el algoritmo de aprendizaje y puede "inferir" probabilidades, dadas ciertas observaciones "convenientes" que modelan las variables de predicción del problema tratado en el dominio.

En esta sección se explican los conceptos en los que se basa la implementación del algoritmo de aprendizaje Bayesiano [1] implementado como parte de BIframework.

Para examinar el enfoque Bayesiano, primero se define el concepto de red Bayesiana y se establecen los conceptos básicos requeridos para abordar el tema.

3.1. Red Bayesiana

Una red Bayesiana es un grafo acíclico dirigido (GAD) con un conjunto de probabilidades condicionales definidas sobre él. Los nodos de la red representan eventos (variables que toman ciertos valores), y las aristas representan dependencias condicionales de los eventos [3]. En la figura 1 se muestra una red Bayesiana.

Las redes Bayesianas se utilizan para capturar las dependencias que existen entre variables que representan situaciones para toma de decisiones. Cada variables se representa con un nodo en la red, y resolver una red es determinar la probabilidad condicional de un nodo o un conjunto de nodos, dada la instanciación de evidencia acerca de los valores de otros nodos [3].

3.2. Notación para variables, valores y distribuciones

Las **variables** se indican con letras itálicas en mayúscula (A,B,C), y los **valores** de las variables con letras itálicas en minúscula (a,b,c). Se **instancia** la variable A asignándole un valor a, (a indica una **instanciación** de A).

Los conjuntos de variables se indican por letras mayúsculas en negrilla ($\mathbf{V}, \mathbf{V_i}, \mathbf{X}, \mathbf{\Theta_i}$), y sus instanciaciones por letras minúsculas en negrilla ($\mathbf{v}, \mathbf{v_i}, \mathbf{x}, \theta_\mathbf{i}$). Se instancia un conjunto de variables $\mathbf{V}$ asignándole un valor a cada variable en $\mathbf{V}$; se indica esta asignación como $\mathbf{v}$, y se dice que $\mathbf{v}$ es una instanciación de $\mathbf{V}$.

Se dice que el conjunto de variables $\mathbf{V}$ está en la configuración $\mathbf{v}$. Se utiliza $p(V = v|\xi)$ (abreviando $p(v|\xi)$) para indicar la probabilidad de que $V = v$ para una persona con estado de información ξ. También se utiliza $p(v|\xi)$ para indicar la **distribución de probabilidad** de V (tanto funciones de masa como de densidad). Ya sea que $p(v|\xi)$ se refiera a una probabilidad, una densidad de probabilidad, o a una distribución de probabilidad, será claro en el contexto.

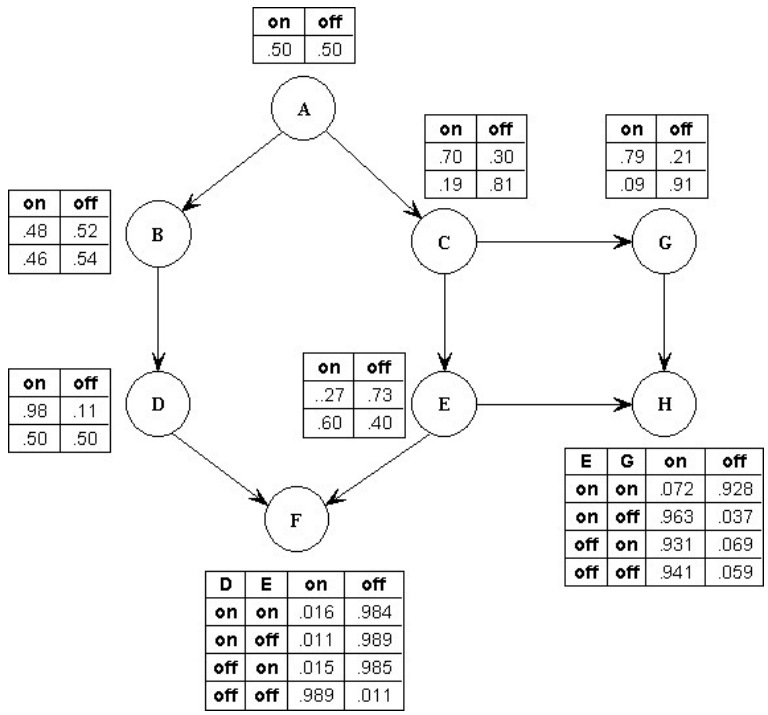


Figura 1: Red Bayesiana mostrando probabilidades aprendidas

3.3. Enfoque Bayesiano

Para ilustrar el enfoque Bayesiano, se usa una moneda común. Con 'cara' en un lado y 'sol' en el otro. Si se lanza la moneda hacia arriba en el aire, caerá al suelo en alguno de sus dos lados. Si se lanza la moneda N veces y las propiedades físicas de la moneda y las condiciones bajo las cuales se lanza permanecen estables en el tiempo, se desea determinar la probabilidad de 'cara' en un lanzamiento $N+1$.

En el análisis clásico del problema (frecuentista), se afirma que hay alguna probabilidad física de 'cara', que es desconocida, se estima esta probabilidad física de las N observaciones utilizando el criterio de bajo sesgo y baja varianza. Entonces se utiliza esta estimación como la probabilidad para 'cara' en el lanzamiento $N+1$.

En el enfoque Bayesiano, también se afirma que hay alguna probabilidad física de 'cara', pero se codifica la incertidumbre acerca de esta probabilidad física utilizando probabilidades (Bayesianas), y se utilizan las reglas de probabilidad para ***calcular*** la probabilidad de 'cara' en el lanzamiento $N+1$.

Para codificar los posibles valores verdaderos de una probabilidad física, se define la variable Θ. Se dice que θ es un *parámetro*. Se expresa la incertidumbre acerca de Θ usando la función de densidad de probabilidad $p(\theta|\xi)$. Adicionalmente, se usa V_l para indicar la variable que representa una salida en el lanzamiento l_{simo}, $l = 1, \ldots, N+1$ y $D = \{V_1 = v_1, \ldots, V_n = v_n\}$ para indicar el

conjunto de observaciones.

3.3.1. Muestreo Binomial

En términos Bayesianos, el problema de la moneda se reduce a calcular $p(v_{N+1}|D,\xi)$ a partir de $p(\theta|\xi)$.

Para hacerlo, primero se utiliza la regla de Bayes para obtener la distribución de probabilidad Θ dado D y el último conocimiento ξ:

$$p(\theta|D,\xi) = \frac{p(\theta|\xi)p(D|\theta,\xi)}{p(D|\xi)} \tag{1}$$

donde

$$p(\theta|\xi) = \int p(D|\theta,\xi)p(\theta|\xi)d\theta \tag{2}$$

Se expande el término $p(D|\theta,\xi)$. Tanto los estadistas Frecuentistas como Bayesianos están de acuerdo en este término: es la función de verosimilitud para muestreo binomial. En particular, dado el valor de Θ, las observaciones en D son mutuamente independientes, y la probabilidad de 'cara' (o 'sol') en cualquiera de las observaciones es $\theta(1-\theta)$. En consecuencia, la ecuación (1) se transforma a:

$$p(\theta|D,\xi) = \frac{p(\theta|\xi)\theta^c(1-\theta)^s}{p(D|\xi)} \tag{3}$$

donde c y s son el número de 'caras' y 'soles' observados en D respectivamente. Se llama a las distribuciones de probabilidad $p(\theta|\xi)$ y $p(\theta|D,\xi)$ como *prioris* y *posterioris* para Θ respectivamente. Se dice que las cantidades de c y s son estadísticos suficientes para muestreo binomial, porque proporcionan una sumarización para cada valor diferente que se presenta en los datos, y por tanto, se pueden usar para calcular la *posteriori* a partir de la *priori*.

Finalmente, se promedia sobre los posibles valores de Θ para determinar la probabilidad de que el lanzamiento $N+1$ de la moneda caerá 'cara':

$$p(V_{N+1} = cara|D,\xi) = \int p(V_{N+1} = cara|\theta,\xi)p(\theta|D,\xi)d\theta = \int \theta p(\theta|D,\xi)d\theta \equiv \mathbf{E}_{p(\theta|D,\xi)} \tag{4}$$

donde $\mathbf{E}_{p(\theta|D,\xi)}$ indica el valor esperado de θ con respecto a la distribución $p(\theta|D,\xi)$.

3.3.2. Utilidad de la distribución Beta

Para completar la historia Bayesiana para este ejemplo, se necesita un método para evaluar la distribución a *priori* para Θ. Un enfoque común, es asumir que esta distribución es una distribución *beta*:

$$p(\theta|\xi) = Beta(\theta|\alpha_c,\alpha_s) \equiv \frac{\Gamma(\alpha)}{\Gamma(\alpha_c)\Gamma(\alpha_s)}\theta^{\alpha_c-1}(1-\theta)^{\alpha_s-1} \tag{5}$$

donde $\alpha_c > 0$ y $\alpha_s > 0$ son los parámetros de la distribución beta, $\alpha = \alpha_c + \alpha_s$, y $\Gamma(.)$ es la función *Gamma* que satisface $\Gamma(x+1) = x\Gamma(x)$ y $\Gamma(1) = 1$. A las cantidades α_c y α_s se les llama

hiperparámetros para distinguirlos de los parámetros θ. Los hiperparámetros α_c y α_s deben ser mayores que cero para que la distribución pueda ser normalizada.

El **priori beta** es conveniente por varias razones, por la ecuación (3), la distribución *posteriori* también será una distribución beta:

$$p(\theta|D,\xi) = \frac{\Gamma(\alpha+N))}{\Gamma(\alpha_c+c)\Gamma(\alpha_s+s)}\theta^{\alpha_c+c-1}(1-\theta)^{\alpha_s+s-1} = \mathbf{Beta}(\theta|\alpha_c+c,\alpha_s+s) \tag{6}$$

Se dice que el conjunto de distribuciones beta son una familia conjugada de distribuciones para muestreo Binomial. También, el valor esperado de θ con respecto a esta distribución tiene una forma simple:

$$\int \theta \mathbf{Beta}(\theta|\alpha_c,\alpha_s)d\theta = \frac{\alpha_c}{\alpha} \tag{7}$$

Así, dado un priori beta, se cuenta con una expresión simple para la probabilidad de 'cara' en el lanzamiento $N+1$:

$$p(V_{N+1} = \mathit{cara}|D,\xi) = \frac{\alpha_c+c}{\alpha+N} \tag{8}$$

A pesar de la forma funcional, se puede aprender acerca de los parámetros dados los Datos usando el enfoque Bayesiano.

Como se hizo en el caso Binomial, se definen variables que correspondan a los parámetros desconocidos, se asignan *prioris* a esas variables, y se utiliza la regla de Bayes para actualizar la creencia acerca de esos parámetros dados los datos:

$$p(\boldsymbol{\theta}|D,\xi) = \frac{p(D|\boldsymbol{\theta},\xi)p(\boldsymbol{\theta}|\xi)}{p(D|\xi)} \tag{9}$$

Entonces se promedia sobre los posibles valores de Θ para hacer predicciones. Por ejemplo:

$$p(v_{N+1}|D,\xi) = \int p(v_{N+1}|\boldsymbol{\theta},\xi)p(\boldsymbol{\theta}|D,\xi)d\boldsymbol{\theta} \tag{10}$$

Para la clase de distribuciones conocidas como *familia exponencial*, estos cálculos pueden hacerse en forma eficiente y en forma cerrada. Algunos miembros de esta clase son las distribuciones binomial, multinomial, Gamma y Poisson.

3.4. Muestreo multinomial con distribución Dirichlet

En el muestreo multinomial, la variable observada V es discreta, con r posibles estados $v^1,\ldots,v^r$. La función de verosimilitud está dada por:

$$p(V = v^k|\boldsymbol{\theta},\xi) = \theta_k, \qquad k = 1,\ldots,r$$

donde $\boldsymbol{\theta} = \{\theta_2,\ldots,\theta_r\}$ son los parámetros. El parámetro θ_1 está dado por $1-\sum_{k=2}^{r}\theta_k$.

En este caso, como en el caso de muestreo binomial, los parámetros corresponden a probabilidades físicas. Los estadísticos suficientes para el conjunto de datos $D = \{V_1 = v_1,\ldots,V_n = v_n\}$ son

$\{N_1, \ldots, N_r\}$, donde N_i es el número de veces que $V = v^k$ en D. El prior conjugado usado con el muestreo multinomial es la distribución de Dirichlet:

$$p(\boldsymbol{\theta}|\xi) = \mathbf{Dir}(\boldsymbol{\theta}|\alpha_1, \ldots, \alpha_r) \equiv \frac{\Gamma(\alpha)}{\prod_{k=1}^{r} \Gamma(\alpha_k)} \prod_{k=1}^{r} \theta_k^{\alpha_k - 1} \tag{11}$$

donde $\alpha = \sum_{i=1}^{r} \alpha_k$, y $\alpha_k > 0, k = 1, \ldots, r$.

La distribución *posteriori* también es una distribución Dirichlet: $p(\boldsymbol{\theta}|D, \xi) = \mathbf{Dir}(\boldsymbol{\theta}|\alpha_1 + N_1, \ldots, \alpha_r + N_r)$.

Dado este *prior* conjugado y el conjunto de datos D, la distribución de probabilidad para la siguiente observación está dada por:

$$p(V_{N+1} = v^k|D, \xi) = \int \theta_k \mathbf{Dir}(\boldsymbol{\theta}|\alpha_1 + N_1, \ldots, \alpha_r + N_r) d\boldsymbol{\theta} = \frac{\alpha_k + N_k}{\alpha + N} \tag{12}$$

3.5. Aprendizaje de probabilidades en una red Bayesiana

Una vez que se estableció claramente el enfoque Bayesiano a la probabilidad, se observa que el Algoritmo de Aprendizaje Bayesiano representando probabilidades mediante muestreo multinomial, específicamente usando la distribución Dirichlet, consiste simplemente en el "conteo" de parámetros de dicha distribución. Este conteo se realiza tomando los datos y estructura de la red como entrada, y contando en los datos cada vez que cada variable (en cada uno de los nodos de la red) toma cada uno de los diferentes valores que puede tomar, dado el valor que a su vez puede tomar cada uno de sus padres (nodos que le preceden en la red), en cada una de las "instanciaciones" posibles dada la estructura de la red.

Se asume que la distribución de probabilidad conjunta V puede codificarse en alguna estructura de red S. Se escribe:

$$p(\mathbf{v}|\boldsymbol{\theta_s}, S^h) = \prod_{i=1}^{n} p(v_i|\mathbf{pa}_i, \boldsymbol{\theta_i}, S^h)$$

donde $\boldsymbol{\theta_i}$ es el vector de parámetros para la distribución $p(v_i|\mathbf{pa}_i, \boldsymbol{\theta_i}, S^h)$, $\boldsymbol{\theta_s}$ es el vector de parámetros $(\boldsymbol{\theta_1}, \boldsymbol{\theta_2}, \ldots, \boldsymbol{\theta_n})$, y S^h denota el evento de que la distribución física de probabilidad conjunta puede ser factorizada en base a S, es decir S es la ***estructura de la red Bayesiana***. Adicionalmente, se asume que se tiene la muestra aleatoria $\mathbf{D} = \{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_N}\}$ de la distribución física de probabilidad conjunta $\mathbf{V}$. Se dice que un elemento $\mathbf{v_i}$ de D es un *caso*. Se codifica la incertidumbre acerca de los parámetros $\boldsymbol{\theta_s}$ definiendo una variable (vector) $\boldsymbol{\Theta_s}$, y evaluando una función de densidad de probabilidad a priori $p(\boldsymbol{\theta_s}|S^h)$.

El problema de aprender probabilidades en una red Bayesiana puede establecerse simplemente:

Dada una muestra aleatoria D, se calcula la distribución posterior $p(\boldsymbol{\theta_s}|D, S^h)$.

La distribución $p(v_i|\mathbf{pa}_i, \boldsymbol{\theta_i}, S^h)$, vista como una función de $\boldsymbol{\theta_i}$, es una función de distribución local, es decir, una función de clasificación o regresión. Entonces, una red Bayesiana puede ser vista

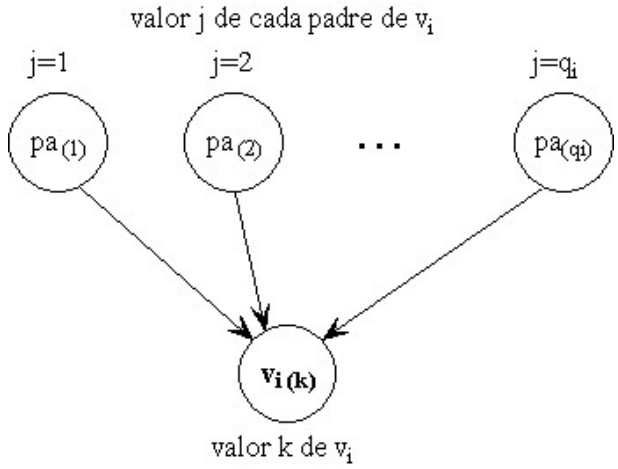


Figura 2: Ejemplo gráfico de la distribución multinomial

como una colección de modelos de clasificación/regresión organizados por relaciones de independencia condicional. Algunos ejemplos de modelos de clasificación/regresión que producen salidas probabilísticas son: regresión lineal, regresión lineal generalizada, redes neuronales probabilísticas y árboles de decisión probabilísticos. En principio, cualquiera de esas formas se pueden utilizar para aprender probabilidades en una red Bayesiana. Los modelos más estudiados son: distribución multinomial, regresión lineal con ruido Gaussiano y regresión lineal generalizada.

3.5.1. Aprendizaje con modelo de distribución multinomial

Para implementar el algoritmo de aprendizaje de este proyecto, se utilizará la distribución multinomial. En este caso, cada variable $\boldsymbol{V_i} \in \mathbf{V}$ es discreta, con r_i posibles valores $v_i^1, \ldots, v_i^{r_i}$, y cada función de distribución local es una colección de distribuciones multinomiales, una distribución para cada configuración $\mathbf{Pa}_i$. Expresamente, se asume:

$$p(v_i^k | \mathbf{pa}_i^j, \boldsymbol{\theta_i}, S^h) = \theta_{ijk} > 0$$

donde i es el nodo, j son los padres del nodo, k son los posibles valores del nodo y $pa_i^1, \ldots, pa_i^{q_i}$ $q_i = \prod_{V_i \in \mathbf{Pa}_i} r_i$ denotan las configuraciones de $\mathbf{Pa}_i$, y $\boldsymbol{\theta_i} = ((\theta_{ijk})_{k=2}^{r_i})_{j=1}^{q_i}$ son los parámetros. (El parámetro θ_{ij1} está dado por $1 - \sum_{k=2}^{r_i} \theta_{ijk}$. En la figura 2 se muestra una representación gráfica de la distribución multinomial. Por conveniencia, se definen los vectores de parámetros:

$$\boldsymbol{\theta}_{ij} = (\theta_{ij2}, \ldots, \theta_{ijr_i}) \qquad \forall \quad i, j$$

3.5.2. Cálculo de la distribución posterior

Dada esta clase de funciones de distribución local, se puede calcular la distribución posterior $p(\boldsymbol{\theta_s} | D, S^h)$ eficientemente y en forma cerrada bajo dos suposiciones.

1. La primera es que no hay datos incompletos en la muestra aleatoria D, es decir, la muestra aleatoria D es completa.

2. La segunda suposición es que los vectores de parámetros $\boldsymbol{\theta}_{ij}$ son mutuamente independientes, esto es:

$$p(\boldsymbol{\theta_s}|S^h) = \prod_{i=1}^{n}\prod_{j=1}^{q_i} p(\boldsymbol{\theta}_{ij}|S^h)$$

A esta suposición se le llama *independencia de parámetros.*

Bajo la suposición de datos completos e independencia de parámetros, los parámetros permanecen independientes dada una muestra aleatoria:

$$p(\boldsymbol{\theta_s}|D, S^h) = \prod_{i=1}^{n}\prod_{j=1}^{q_i} p(\boldsymbol{\theta}_{ij}|D, S^h)$$

3.5.3. Actualización de parámetros en forma independiente

Entonces, se puede actualizar cada vector de parámetros $\boldsymbol{\theta}_{ij}$ en forma independiente, tal como en el caso de una sola variable. Asumiendo que cada vector $\boldsymbol{\theta}_{ij}$ tiene su distribución a priori $\mathbf{Dir}(\boldsymbol{\theta}_{ij}|\alpha_{ij1}, \ldots, \alpha_{ijr_i})$, se obtiene la distribución posterior:

$$p(\boldsymbol{\theta}_{ij}|D, S^h) = \mathbf{Dir}(\boldsymbol{\theta}_{ij}|\alpha_{ij1} + N_{ij1}, \ldots, \alpha_{ijr_i} + N_{ijr_i})$$

donde N_{ijk} es el número de casos en D en que $V_i = v_i^k$ y $\mathbf{Pa}_i = \mathbf{pa}_i^j$.

Se puede promediar sobre las posibles configuraciones de $\boldsymbol{\theta_s}$ para obtener predicciones de interés. Por ejemplo, si se desea calcular $p(\mathbf{v}_{N+1}|D, S^h)$, donde $\mathbf{v}_{N+1}$ es el siguiente caso a ser visto después de D. Supóngase que, en el caso $\mathbf{v}_{N+1}$, $V_i = v_i^k$ y $\mathbf{Pa}_i = \mathbf{pa}_i^j$, donde k y j dependen de i. Entonces:

$$p(\mathbf{v}_{n+1}|D, S^h) = E_{p(\boldsymbol{\theta_s}|D,S^h)}\left(\prod_{i=1}^{n}\theta_{ijk}\right)$$

Para calcular este valor esperado, primero se utiliza el hecho de que los parámetros permanecen independientes dado D:

$$p(\mathbf{v}_{N+1}|D, S^h) = \int \prod_{i=1}^{n}\theta_{ijk} p(\boldsymbol{\theta_s}|D, S^h) d\boldsymbol{\theta_s} = \prod_{i=1}^{n}\int \theta_{ijk} p(\boldsymbol{\theta}_{ij}|D, S^h) d\boldsymbol{\theta}_{ij}$$

Entonces, se utiliza la ecuación (12) (sección 3.4) y se obtiene el modelo de aprendizaje Bayesiano basado en parámetros:

$$p(\mathbf{v}_{N+1}|D, S^h) = \prod_{i=1}^{n}\frac{\alpha_{ijk} + N_{ijk}}{\alpha_{ij} + N_{ij}}$$

donde $\alpha_{ij} = \sum_{k=1}^{r_i}\alpha_{ijk}$ y $N_{ij=}\sum_{k=1}^{r_i} N_{ijk}$.

Esta ecuación es muy relevante, indica que se puede determinar el valor de probabilidad de un dato no observado, simplemente contando la frecuencia con que se presenta en los datos cada valor posible, de cada una de las variables, para cada instanciación de sus padres. Este es el modelo que se implementa en el algoritmo de aprendizaje Bayesiano. En la ecuación se puede observar que la estructura de la red se conoce, por lo tanto, la implementación que se incluye en BIframework considera que se conoce la estructura de la red.

3.5.4. Aprendizaje con datos incompletos

Cuando el conjunto de datos D está incompleto, el cálculo exacto de la distribución *posterior* para $\boldsymbol{\theta_s}$ será intratable y se usa una aproximación basada en la configuración de 'Maximum a Posteriori' (MAP) para obtener el valor de cada parámetro en la muestra i, y se actualizan los parámetros como si se tuvieran datos Completos.

4. Inferencia Bayesiana

Las redes Bayesianas dependen de algoritmos de inferencia para calcular creencias de hipótesis alternativas en el contexto de evidencia observada. La tarea de implementar un algoritmo de inferencia no es trivial, se ha dedicado mucho esfuerzo en sintetizar métodos recolectados en la literatura para convertirlos a forma algorítmica. En esta sección se describe un algoritmo de inferencia probabilística en forma procedural, el método de Propagación de Probabilidades en árboles de Clusters (PPAC) [2], tal como se desarrolló por Lauritzen y Spiegelhalter [4]. El algoritmo PPAC es un método establecido para inferencia probabilística exacta, y se describe con suficiente nivel de detalle, para que el lector entienda la implementación ralizada como parte de BIframework.

El algoritmo PPAC trabaja en dos pasos:

1. Convierte la red Bayesiana en una estructura secundaria
2. Calcula las probabilidades de interés operando sobre la estructura secundaria.

En la sección 6.6 se presenta un ejemplo completo, paso a paso, de la mecánica de razonamiento Bayesiano utilizada en el algoritmo implementado como parte de BIframework. Para entender su funcionamiento, en esta sección se definen los conceptos básicos utilizados, y se explica a detalle la metodología y funcionamiento del algoritmo.

4.1. Conceptos básicos usados en el algoritmo de inferencia PPAC

4.1.1. Potenciales

Se define un **potencial** sobre un conjunto de variables $\mathbf{X}$ como una función que mapea cada instanciación de $\mathbf{x}$ en un número real no negativo; se indica este potencial como $\phi_{\mathbf{X}}$. Se usa la notación $\phi_{\mathbf{X}}$ para indicar el número al que $\phi_{\mathbf{X}}$ mapea a $\mathbf{x}$; A cada $\phi_{\mathbf{X}}(\mathbf{X})$ se le llama **elemento**. Los potenciales pueden verse como **matrices** y pueden ser implementados como **tablas**.

4.1.2. Operaciones sobre potenciales

Se definen dos operaciones básicas sobre potenciales: **marginalización** y **multiplicación**. Se supone que se tiene un conjunto de variables $\mathbf{Y}$, su potencial $\phi_{\mathbf{Y}}$, y un conjunto de variables $\mathbf{X}$ donde $\mathbf{X} \subseteq \mathbf{Y}$.

La **marginalización** de $\phi_{\mathbf{Y}}$ hacia $\mathbf{X}$ es un potencial $\phi_{\mathbf{X}}$, donde cada $\phi_{\mathbf{X}}(\mathbf{x})$ se calcula como sigue:

1. Identificar las instanciaciones $\mathbf{y_1}, \mathbf{y_2}, \ldots$ que son consistentes con $\mathbf{x}$.
2. Asignar a $\phi_{\mathbf{X}}(\mathbf{x})$ la suma $\phi_{\mathbf{Y}}(\mathbf{y_1}) + \phi_{\mathbf{Y}}(\mathbf{y_2}) + \ldots$
   Esta marginalización se indica como sigue:

$$\phi_{\mathbf{X}} = \sum_{\mathbf{Y} \setminus \mathbf{X}} \phi_{\mathbf{Y}}$$

Dados dos conjuntos de variables $\mathbf{X}$ y $\mathbf{Y}$ y sus potenciales $\phi_{\mathbf{X}}$ y $\phi_{\mathbf{Y}}$, la **multiplicación** de $\phi_{\mathbf{X}}$ y $\phi_{\mathbf{Y}}$ es un potencial $\phi_{\mathbf{Z}}$, donde $\mathbf{Z} = \mathbf{X} \cup \mathbf{Y}$, y cada $\phi_{\mathbf{Z}}$ se calcula como sigue:

1. Identificar la instanciación de $\mathbf{x}$ y la instanciación de $\mathbf{y}$ que son consistentes con $\mathbf{z}$.

2. Asignar a $\phi_{\mathbf{Z}}(\mathbf{z})$ el producto de $\phi_{\mathbf{X}}\phi_{\mathbf{Y}}$.
   Esta multiplicación de potenciales se indica como sigue:

$$\phi_{\mathbf{Z}} = \phi_{\mathbf{X}}\phi_{\mathbf{Y}}$$

4.1.3. Distribuciones de probabilidad

Una **distribución de probabilidad** es un caso especial de un potencial. Dado un conjunto de variables $\mathbf{X}$, se usa la notación $P(\mathbf{X})$ para indicar la **distribución de probabilidad de X**. $P(\mathbf{X})$ es un potencial sobre $\mathbf{X}$ cuyos elementos suman 1. Se indican los elementos de $P(\mathbf{X})$ como $P(\mathbf{x})$, y a cada elemento $P(\mathbf{x})$ se le llama la **probabilidad de x**. Con esta notación, se tiene:

$$\sum_{\mathbf{X}} P(\mathbf{x}) = 1.$$

4.1.4. Probabilidad condicional

Dados los conjuntos de variables $\mathbf{X}$ y $\mathbf{Y}$, se usa la notación $P(\mathbf{X}|\mathbf{Y})$ para indicar la **probabilidad condicional de X dado Y**, o simplemente la **probabilidad de X dado Y**. $P(\mathbf{X}|\mathbf{Y})$ es una colección de distribuciones de probabilidad indexadas por las instanciaciones de $\mathbf{Y}$; cada $P(\mathbf{X}|\mathbf{y})$ es una distribución de probabilidad sobre $\mathbf{X}$. Se indican los elementos de $P(\mathbf{X}|\mathbf{y})$ como $P(\mathbf{x}|\mathbf{y})$, y se dice que cada elemento $P(\mathbf{x}|\mathbf{y})$ es la **probabilidad de x dado y**. Con esta notación, se tiene para cada instanciación de $\mathbf{y}$:

$$\sum_{\mathbf{X}} P(\mathbf{x}|\mathbf{y}) = 1.$$

$P(\mathbf{X})$ es un caso especial de $P(\mathbf{X}|\mathbf{Y})$ donde $\mathbf{Y} = \emptyset$.

4.1.5. Representación de una red Bayesiana

Una red Bayesiana sobre un conjunto de variables $\mathbf{U} = \{V_1, \ldots, V_n\}$ se representa mediante dos componentes:

1. Un **Grafo Acíclico Dirigido** [8] (**GAD**) $\mathcal{G}$: Cada vértice en el grafo representa una variable V, que toma valores v_1, v_2, etc. Los **padres** de V en el grafo se indican como $\mathbf{\Pi_V}$, con instanciaciones π_V; la **familia** de V, indicada como $\mathbf{F_V}$, se define como $\{V\} \cup \mathbf{\Pi_V}$. La estructura del **GAD** codifica un conjunto de declaraciones de independencia que restringen la variedad de interacciones que pueden ocurrir entre las variables.

2. Una **cuantificación** de $\mathcal{G}$: Cada variable en $\mathcal{G}$ se cuantifica con una tabla de probabilidad condicional $P(V|\mathbf{\Pi_V})$. Mientras que $P(V|\mathbf{\Pi_V})$ es técnicamente una función de $\mathbf{F_V}$, es más útil pensar de la siguiente forma: para cada instanciación π_v, a cada valor de v se le asignan números reales $[0, 1]$ de tal forma que sumen 1. Cuando $\mathbf{\Pi_V} \neq \emptyset$, a $P(V|\mathbf{\Pi_V})$ se le llama la **probabilidad condicional de** V dados $\mathbf{\Pi_V}$; cuando $\mathbf{\Pi_V} = \emptyset$, a $P(V|\mathbf{\Pi_V})$ o simplemente $P(V)$ se le llama la **probabilidad a priori** de V.

Estos componentes inducen una **distribución de probabilidad conjunta** sobre $\mathbf{U}$, dada por:

$$P(\mathbf{U}) = \prod_{i=1}^{n} P(V_i|\mathbf{\Pi}_{V_i})$$

donde $V_1, V_2, \ldots, V_n$ son las variables en la red Bayesiana.

4.2. Estructura secundaria

Dada una red Bayesiana sobre un conjunto de variables $\mathbf{U} = \{V_1, \ldots, V_n\}$, se define la estructura secundaria que contiene un componente gráfico y un componente numérico.

El componente gráfico consiste de lo siguiente:

- Un **árbol no dirigido** [8] $\mathcal{T}$: Cada nodo en $\mathcal{T}$ es un **cluster** (conjunto no vacío) de variables. Los clusters satisfacen la propiedad de árbol conjunto: dados dos clusters $\mathbf{X}$ y $\mathbf{Y}$ en $\mathcal{T}$, todos los clusters en la trayectoria entre $\mathbf{X}$ y $\mathbf{Y}$ contienen $\mathbf{X} \cap \mathbf{Y}$. Para cada variable $V \in \mathbf{U}$, la familia de V, $\mathbf{F_V}$ se induce en al menos uno de los clusters.

- **sepsets**: Cada arista en $\mathcal{T}$ se etiqueta con la intersección de los clusters adyacentes; estas etiquetas son llamadas 'separator sets', o **sepsets**.

El componente numérico se describe utilizando la noción de potencial de creencia. Un potencial de creencia es una función que mapea cada instanciación de un conjunto de variables a un número real. Los potenciales de creencia se definen sobre el siguiente conjunto de variables:

- Clusters: Cada cluster $\mathbf{X}$ se asocia con un potencial de creencia $\phi_{\mathbf{X}}$ que mapea cada instanciación de $\mathbf{x}$ a un número real.

- Sepsets: Cada sepset $\mathbf{S}$ se asocia con un potencial de creencia $\phi_{\mathbf{S}}$ que mapea cada instanciación de $\mathbf{s}$ a un número real.

Los potenciales de creencia no se especifican en forma arbitraria; deben satisfacer las siguientes restricciones:

- Para cada cluster $\mathbf{X}$ y sus sepset vecino $\mathbf{S}$, debe cumplirse que:

$$\sum_{\mathbf{X} \setminus \mathbf{S}} \phi_{\mathbf{X}} = \phi_{\mathbf{S}} \tag{13}$$

  Cuando la ecuación (13) se satisface para un cluster $\mathbf{X}$ y su sepset vecino $\mathbf{S}$, se dice que $\phi_{\mathbf{S}}$ es consistente con $\phi_{\mathbf{X}}$. Cuando cada par de cluster-sepset es consistente, se dice que la estructura secundaria es **localmente consistente**.

- Los potenciales de creencia codifican una distribución conjunta $P(\mathbf{U})$ sobre la red Bayesiana de acuerdo a:

$$P(\mathbf{U}) = \frac{\prod_i \phi_{\mathbf{X}_i}}{\prod_j \phi_{\mathbf{S}_j}} \tag{14}$$

  donde $\phi_{\mathbf{X}_i}$ y $\phi_{\mathbf{S}_j}$ son los potenciales del cluster y sepset respectivamente.

Un paso clave en la PPAC es la construcción de una estructura secundaria que satisfaga las restricciones mencionadas. Tal estructura secundaria tiene la siguiente propiedad importante: para cada cluster (o sepset) $\mathbf{X}$, se cumple que $\phi_{\mathbf{X}} = P(\mathbf{X})$. Utilizando esta propiedad, se puede calcular la distribución de probabilidad de cualquier variable V, utilizando cualquier cluster (o sepset) de $\mathbf{X}$ que contenga a V, como sigue:

$$P(V) = \sum_{\mathbf{X} \setminus \{\mathbf{V}\}} \phi_{\mathbf{X}} \tag{15}$$

La estructura secundaria ha sido referida en la literatura como árbol conjunto (join tree), árbol de universos de creencia, árbol de clusters y árbol de cliques.

4.3. Construcción del árbol conjunto a partir de la red Bayesiana

La construcción del árbol conjunto a partir de una red Bayesiana como la que se muestra en la figura 3, involucra la construcción de varias estructuras intermedias que pueden resumirse como sigue:

1. A partir del **GAD**, construir un grafo no dirigido, llamado **grafo moral** (sección 4.3.1).
2. En forma selectiva, añadir arcos al grafo moral para formar un **grafo triangulado** (sección 4.3.2).
3. A partir del grafo triangulado, identificar y seleccionar subconjuntos de nodos, llamados **cliques** [8] (sección 4.3.3).
4. Construir el árbol conjunto, empezando con los cliques como clusters: conectar los clusters para formar un árbol no dirigido que satisfaga la propiedad de árbol conjunto, insertando los sepsets apropiados (sección 4.3.4).

Los pasos 2 a 4 son no determinísticos; por lo tanto, se pueden construir varios árboles conjuntos diferentes a partir del mismo **GAD**.

4.3.1. Construcción del grafo moral

Sea $\mathcal{G}$ el **GAD** de una red Bayesiana. El grafo moral $\mathcal{G}_M$ correspondiente a $\mathcal{G}$ se construye de la siguiente forma:

1. Crear el grafo no dirigido $\mathcal{G}_u$ copiando $\mathcal{G}$, pero eliminando las direcciones de los arcos.
2. Crear $\mathcal{G}_M$ a partir de $\mathcal{G}_u$ como sigue: Para cada nodo V, identificar a sus padres $\mathbf{\Pi_V}$ en $\mathcal{G}$. Conectar cada par de nodos en $\mathbf{\Pi_V}$ añadiendo arcos no dirigidos a $\mathcal{G}_u$.

La figura 4 ilustra esta transformación sobre el GAD de la figura 3. Los arcos no dirigidos añadidos a $\mathcal{G}_u$ son llamados **arcos morales** y se muestran como líneas punteadas en la figura.

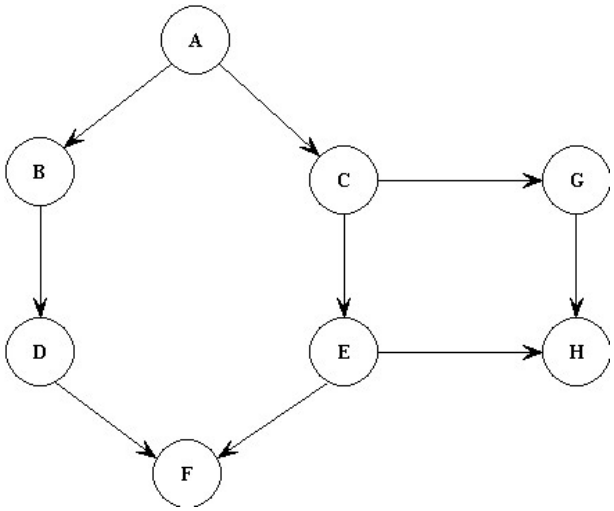


Figura 3: Red Bayesiana representada por un GAD

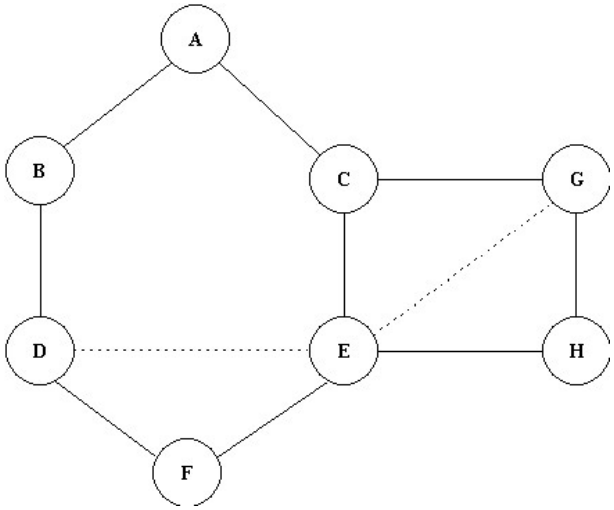


Figura 4: Grafo moral

4.3.2. Triangulación del grafo moral

Un grafo no dirigido está triangulado, si y solo sí, cada ciclo de longitud cuatro o mayor contiene una arista que conecta dos nodos no adyacentes en el ciclo. El procedimiento para triangular es el siguiente:

1. Hacer una copia de $\mathcal{G}_M$; llamarlo $\mathcal{G}'_M$.
2. Mientras aún queden nodos en $\mathcal{G}'_M$:
   a) Seleccionar un nodo V de $\mathcal{G}'_M$, de acuerdo al criterio descrito abajo.
   b) El nodo V y sus vecinos en $\mathcal{G}'_M$ forman un cluster. Conectar todos los nodos en el cluster. Para cada arista añadida a $\mathcal{G}'_M$, añadir la misma arista correspondiente a $\mathcal{G}_M$.
   c) Remover V de $\mathcal{G}'_M$.
3. $\mathcal{G}_M$, modificado por los arcos adicionales añadidos en los pasos previos, está ahora triangulado.

Para describir el criterio para seleccionar los nodos en el paso 2a, se utiliza el siguiente concepto de peso:

- El peso de un nodo V es el número de valores de V.
- El peso de un cluster es el producto de los pesos de los nodos que lo conforman.

El criterio para seleccionar que nodos remover se establece como sigue:
Elegir el nodo que cause el menor número de aristas añadidas en el paso 2b, eliminando los empates eligiendo al nodo que induce el cluster con el menor peso.

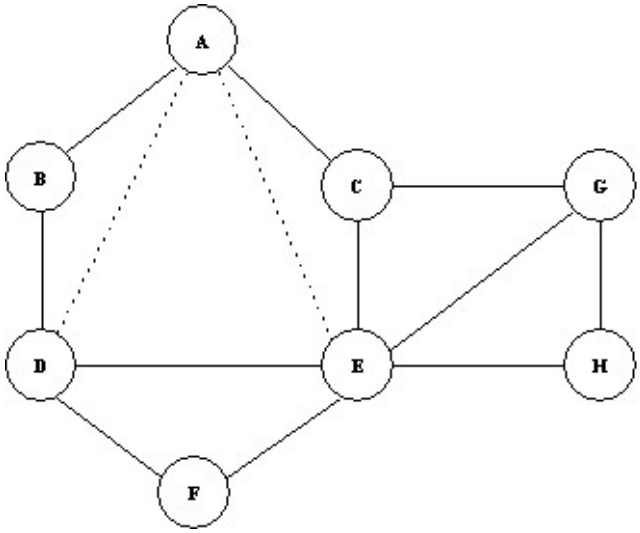
Si aún hay empate, elegir al nodo con **mayor** peso, para que el nodo con peso menor pueda seguir induciendo clusters con menor peso.

La figura 17 muestra el grafo triangulado obtenido a partir del grafo moral de la figura 4. Las líneas punteadas indican las aristas añadidas para triangular el grafo moral. También se muestra el orden de eliminación de los nodos por el algoritmo implementado en este proyecto.

4.3.3. Identificación de cliques

Un **clique** en un grafo no dirigido $\mathcal{G}$ es un subgrafo de $\mathcal{G}$ que es completo y máximo [8]. **Completo** significa que cada par de nodos distintos están conectados por una arista. **Máximo** significa que el clique no está contenido en un subgrafo completo más grande.

Debido a que cada clique en el grafo triangulado es un cluster inducido en el proceso de triangulación, y a que un cluster inducido no puede ser parte de un cluster inducido posteriormente, los cliques se pueden determinar en el proceso de triangulación guardando cada cluster inducido que no es un subconjunto de un cluster guardado previamente. Revisando la figura 17, se observa que los cliques del grafo triangulado son: EGH, CEG, DEF, ACE, ABD y ADE.



Vértice Eliminado	Cluster Inducido	Arista Añadida	Clique
H	HEG		HEG
G	GCE		GCE
F	FDE		FDE
B	BAD	AD	BAD
D	DAE	AE	DAE
C	CAE		CAE
E	AE		
A	A		

Figura 5: Grafo triangulado y tabla que muestra el orden de eliminación de vértices

4.3.4. Construcción del árbol conjunto óptimo

A partir de este punto, ya no se necesita el grafo no dirigido. Se busca construir un árbol conjunto óptimo conectando los cliques obtenidos. Para construir un árbol conjunto óptimo, se deben conectar los cliques de tal forma que el árbol de cliques resultante satisfaga la propiedad de árbol conjunto y un criterio de optimalidad que se detalla adelante. La propiedad de árbol conjunto es esencial para que el árbol sea útil para inferencia probabilística, y el criterio de optimalidad favorece aquellos árboles conjuntos que minimizan el tiempo computacional requerido para efectuar las inferencias.

Dado un conjunto de n cliques, se puede formar el árbol de cliques insertando en forma iterativa las aristas entre pares de cliques, hasta que los cliques estén conectados por $n-1$ aristas. También se puede ver esta tarea como insertar iterativamente sepsets entre pares de cliques, hasta que los cliques estén conectados por $n-1$ sepsets. Se tomará este enfoque para construir el árbol conjunto óptimo. El algoritmo se divide en dos partes:

1. Formar el árbol de cliques seleccionando e insertando iterativamente sepsets candidatos.
2. Cómo seleccionar los sepsets para que el árbol de cliques sea un árbol conjunto óptimo.

Para formar el árbol de cliques se ejecutan los siguientes pasos:

1. Iniciar con un conjunto de n árboles, cada uno con sólo un clique, y un conjunto vacío S.
2. Para cada par distinto de cliques $\mathbf{X}$ y $\mathbf{Y}$:
   a) Crear un sepset candidato, etiquetado $\mathbf{X} \cap \mathbf{Y}$, con apuntadores hacia atrás a los cliques $\mathbf{X}$ y $\mathbf{Y}$. Nombrar a este sepset $\mathbf{S_{XY}}$
   b) Insertar $\mathbf{S_{XY}}$ en S.
3. Repetir hasta que $n-1$ sepset hayan sido insertados en la foresta [8].

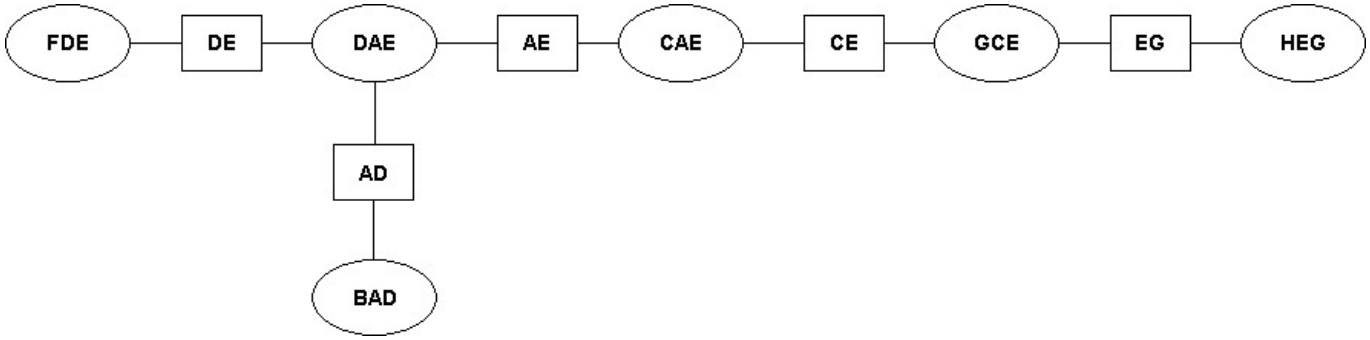


Figura 6: El árbol conjunto

a) Seleccionar un sepset $\mathbf{S_{XY}}$ de S, de acuerdo al criterio de selección especificado adelante. Borrar $\mathbf{S_{XY}}$ de S.

b) Insertar el sepset $\mathbf{S_{XY}}$ entre los cliques $\mathbf{X}$ y $\mathbf{Y}$ sí y solo sí, $\mathbf{X}$ y $\mathbf{Y}$ están en diferentes árboles en la foresta. (Esta inserción junta dos árboles en un árbol mayor).

Para describir como elegir al próximo sepset candidato, se definen las nociones de masa y costo, como sigue:

- La **masa** de un sepset $\mathbf{S_{XY}}$ es el número de variables que contiene, o el número de variables en $\mathbf{X} \cap \mathbf{Y}$.
- El **costo** de un sepset $\mathbf{S_{XY}}$ es el peso de $\mathbf{X}$ más el peso de $\mathbf{Y}$, donde el **peso** se define como sigue:
  - El peso de la variable V es el número de valores de V.
  - El peso de un conjunto de variables $\mathbf{V}$ es el producto del peso de las variables en $\mathbf{V}$.

Con estas nociones establecidas, la forma de elegir al próximo sepset candidato de $\mathbf{S}$ cuando se ejecuta el paso 3a del algoritmo es:

- Para que el árbol de cliques resultante satisfaga la propiedad de árbol conjunto, se debe elegir al sepset candidato con la masa mas grande.
- Cuando dos o mas sepset son elegibles y tienen la misma masa, se puede optimizar el tiempo de inferencia en el árbol conjunto resultante eliminando los empates como sigue: Elegir al sepset candidato con el menor costo.

La figura 6 muestra el árbol conjunto óptimo.

4.4. Proceso de inferencia

Habiendo construido la estructura del árbol conjunto, sólo faltan los procedimientos requeridos para calcular la componente numérica, de tal forma que se satisfagan las condiciones indicadas para la ejecución de la inferencia.

En la figura 7 se muestra el diagrama de bloques del PPAC sin evidencia. Los pasos para realizar inferencias son los siguientes:

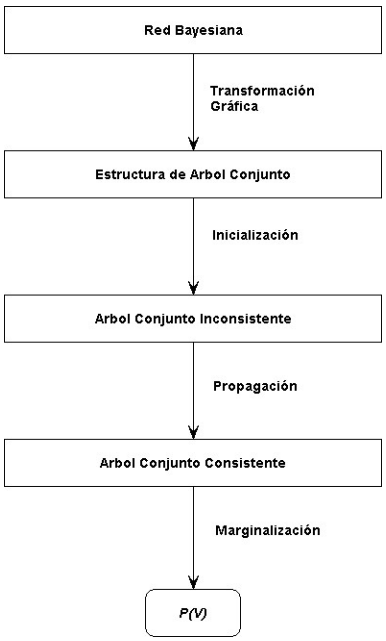


Figura 7: Diagrama de bloques del PPAC sin evidencia

Transformación Gráfica Transformar el GAD de la red Bayesiana en una estructura de árbol conjunto (sección 4.3).

Inicialización Cuantificar el árbol conjunto con los potenciales de creencia de tal forma que satisfagan la ecuación (14). El resultado es un árbol conjunto inconsistente debido a que esta asignación inicial de potenciales de creencia no cumple con los requerimientos de consistencia local indicados por la ecuación (13) (sección 4.4.1).

Propagación Global Ejecuta una serie de manipulaciones locales, llamadas paso de mensajes sobre los potenciales del árbol conjunto. Los pasos de mensajes reacomodan los potenciales del árbol conjunto de tal forma que se hacen consistentes localmente; así, el resultado de la propagación global es un árbol conjunto consistente, que satisface ambas ecuaciones (13) y (14) (sección 4.4.2).

Marginalización A partir del árbol conjunto consistente, calcular $P(C)$ para cada variables V de interés (sección 4.4.3).

4.4.1. Inicialización

El siguiente procedimiento asigna potenciales iniciales al árbol conjunto, usando las probabilidades condicionales de la red Bayesiana:

1. Para cada cluster y sepset $\mathbf{X}$, poner cada $\phi_{\mathbf{X}}(\mathbf{x})$ en 1:

$$\phi_{\mathbf{X}} = 1$$

2. Para cada variable V, ejecutar lo siguiente: Asignar a V un cluster $\mathbf{X}$ que contenga a $\mathbf{F_V}$; llamar a $\mathbf{X}$ el **cluster padre de** $\mathbf{F_V}$. Multiplicar $\phi_{\mathbf{X}}$ por $P(V|\mathbf{\Pi_V})$:

$$\phi_{\mathbf{X}} = \phi_{\mathbf{X}} P(V|\mathbf{\Pi_V})$$

Después de la inicialización, la distribución condicional $P(V|\mathbf{\Pi_V})$ para cada variable V ha sido multiplicada en algún potencial de cluster. El proceso de inicialización satisface la ecuación (14) como sigue:

$$\frac{\prod_{i=1}^{N} \phi_{\mathbf{X_i}}}{\prod_{j=1}^{N-1} \phi_{\mathbf{S_j}}} = \frac{\prod_{k=1}^{Q} P(V_k|\mathbf{\Pi}_{V_k})}{1} = P(\mathbf{U}),$$

donde N es el número de clusters, Q es el número de variables, y $\phi_{\mathbf{X_i}}$ y $\phi_{\mathbf{S_j}}$ son los potenciales de los clusters y sepsets respectivamente.

4.4.2. Propagación global

Habiendo inicializado los potenciales del árbol conjunto, ahora se realiza la propagación global para hacerlos localmente consistentes. La propagación Global consiste de una serie de manipulaciones locales, llamadas paso de mensajes que ocurren entre el cluster $\mathbf{X}$ y un cluster vecino $\mathbf{Y}$. Un paso de mensaje de $\mathbf{X}$ hacia $\mathbf{Y}$ forza que los potenciales de creencia del sepset que intervenga sean consistentes con $\phi_{\mathbf{X}}$ (ver la ecuación (13)), mientras que se preserva la invariancia de la ecuación (14). Cuando se termina la propagación Global, cada par cluster-sepset es consistente, y el árbol, conjunto es consistente.

Paso de mensaje simple.
Si se tienen dos clusters adyacentes $\mathbf{X}$ y $\mathbf{Y}$ con un sepset $\mathbf{R}$ y sus potenciales de creencia $\phi_{\mathbf{X}}$ $\phi_{\mathbf{Y}}$ y $\phi_{\mathbf{R}}$ respectivamente. Un paso de mensaje de $\mathbf{X}$ a $\mathbf{Y}$ ocurre en dos pasos:

1. **Proyección**. Asignar una nueva tabla $\mathbf{R}$, guardando la tabla vieja:

$$\phi_{\mathbf{R}}{}^{old} \longleftarrow \phi_{\mathbf{R}}$$

$$\phi_{\mathbf{R}} \longleftarrow \sum_{\mathbf{X}\backslash\mathbf{R}} \phi_{\mathbf{X}}$$

2. **Absorción**. Asignar una nueva tabla a $\mathbf{Y}$, usando los tablas nueva y vieja de $\mathbf{R}$:

$$\phi_{\mathbf{Y}} \longleftarrow \phi_{\mathbf{Y}} \frac{\phi_{\mathbf{R}}}{\phi_{\mathbf{R}}{}^{old}}$$

Coordinación de múltiples mensajes.

Dado un árbol conjunto de n clusters, el algoritmo de propagación de PPAC empieza por escoger un cluster arbitrario X, luego realiza $2(n-1)$ pases de mensajes, divididos en dos fases:

1. Durante la fase de **Recolección de Evidencia**, cada cluster pasa un mensaje a su vecino en la dirección de $\mathbf{X}$, iniciando con los clusters más alejados de $\mathbf{X}$.

2. Durante la fase de **Distribución de Evidencia**, cada cluster pasa mensajes a sus vecinos en la dirección contraria a $\mathbf{X}$, empezando por $\mathbf{X}$ mismo.

Cada fase pasa $n-1$ mensajes.

4.4.3. Marginalización

Cuando se tiene el árbol conjunto consistente, se puede calcular $P(V)$ para cada variable V, de la siguiente forma:

1. Identificar el cluster (o sepset) X que contiene V.
2. Calcular $P(V)$ por marginalización de $\phi_{\mathbf{X}}$ de acuerdo a la ecuación:

$$P(V) = \sum_{\mathbf{X}\setminus\{\mathbf{V}\}} \phi_{\mathbf{X}}$$

4.5. Manejo de evidencia

El procedimiento descrito arriba para el cálculo de $P(V)$ se modifica cuando se introduce evidencia, se desea calcular $P(V|\xi)$.

Para codificar las evidencias, se define el concepto de verosimilitud.

4.5.1. Observaciones y verosimilitud

Dada una variable V, la verosimilitud de V, denotada por $\mathbf{\Lambda_V}$, es un potencial sobre $\{\mathbf{V}\}$, es decir, $\mathbf{\Lambda_V}$ mapea a cada valor v de V a un número real. La codificación de un conjunto arbitrario de observaciones ξ utilizando $\mathbf{\Lambda_V}$ se hace de la siguiente forma:

- Si $V \in \mathbf{\Xi}$, es decir, si V es evidencia, se asigna cada $\Lambda_V(v)$ como sigue:

$$\Lambda_V(v) = \begin{cases} 1 & \text{cuando } v \text{ es el valor observado} \\ 0 & \text{de otra forma} \end{cases}$$

- Si $V \notin \mathbf{\Xi}$, es decir, si el valor de V se desconoce, entonces se asigna $\Lambda_V(v) = 1$ para cada valor de v.

4.5.2. Inferencia con observaciones

En la figura 8 se muestra el diagrama de bloques del PPAC con observaciones.

Inicialización con Observaciones El procedimiento para hacer inferencia se modifica para introducir un paso adicional: para cada variable V, se inicializa la verosimilitud Λ_V.

Entrada de Observaciones En este paso se codifican e incorporan las evidencias en un árbol conjunto; esto resulta en una modificación de los potenciales del árbol conjunto.

Normalización Para calcular $P(V|\xi)$ para cada variable de interés V, se efectúa marginalización y adicionalmente un paso de normalización.

4.5.3. Inicialización con observaciones

1. Para cada cluster y sepset $\mathbf{X}$, inicializar $\phi_{\mathbf{X}}(\mathbf{x})$a 1:

$$\phi_{\mathbf{X}} \longleftarrow 1$$

2. Para cada variable V:

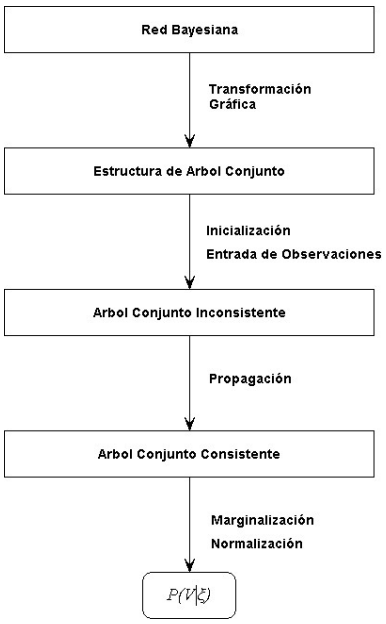


Figura 8: Diagrama de bloques del PPAC con observaciones

a) Asignar a V un cluster X que contenga $\mathbf{F_V}$; Multiplicar $\phi_\mathbf{X}$ por $P(V|\mathbf{\Pi_V})$

$$\phi_\mathbf{X} \longleftarrow \phi_\mathbf{X} P(V|\mathbf{\Pi_V})$$

b) Inicializar cada elemento de verosimilitud $\Lambda_V(v)$ a 1:

$$\mathbf{\Lambda_V}(v) \longleftarrow 1$$

4.5.4. Entrada de observaciones

1. Codificar la observación $V = v$ como una verosimilitud $\mathbf{\Lambda_V^{new}}$.
2. Identificar un cluster $\mathbf{X}$ que contenga a V.
3. Actualizar $\phi_\mathbf{X}$ y $\mathbf{\Lambda_V}$:

$$\phi_\mathbf{X} \longleftarrow \phi_\mathbf{X} \mathbf{\Lambda_V}^{new}$$
$$\mathbf{\Lambda_V} \longleftarrow \mathbf{\Lambda_V}^{new}$$

4.5.5. Normalización

Después de hacer consistente el árbol conjunto utilizando propagación global, se tiene que para cada cluster (o sepset) $\mathbf{X}$, $\phi_\mathbf{X} = P(\mathbf{X}, \xi)$, donde ξ denota las evidencias incorporadas en la sección anterior.
Cuando se marginaliza el potencial de un cluster $\phi_\mathbf{X}$ en una variable V, se obtiene la probabilidad de V y ξ:

$$P(V, \xi) = \sum_{\mathbf{X}\setminus\{\mathbf{V}\}} \phi_\mathbf{X}$$

La meta es calcular $P(V|\xi)$, la probabilidad de V dado ξ.
Se puede obtener a partir de $P(V,\xi)$ normalizando $P(V,\xi)$ de la siguiente forma:

$$P(V|\xi) = \frac{P(V,\xi)}{P(\xi)} = \frac{P(V,\xi)}{\sum_V P(V,\xi)}$$

Parte II
Planteamiento del problema y solución propuesta

5. Planteamiento del problema

5.1. Situación actual

En 1982, fundé la empresa qué aún dirijo: Tecnología Computacional Aplicada (TCA). Durante más de 20 años en TCA nos hemos dedicado a la construcción de sistemas de información integrales para diversas industrias. En la actualidad tenemos sistemas de información integrales que se utilizan con éxito en el mercado latinoamericano en las industrias de Turismo (hoteles), Medicina (hospitales) y Comercio (mayoreo y distribución). Con la apertura comercial que se ha presentado en la última década, la competencia en el mercado de sistemas de información es global, se compite contra empresas transnacionales que tienen propuestas tecnológicas sólidas y de vanguardia.

Los sistemas de información han evolucionado en cuanto a su enfoque, pasaron de ser sistemas modulares transaccionales, a sistemas integrales que abarcan la operación de toda una organización, como resultado de esta evolución, se produce un gran volumen de transacciones relacionadas, es decir, que al ser producidas por un sistema de información integral, incluyen una estrecha relación entre las áreas del negocio que las producen. Para mejorar el uso de las transacciones generadas por las organizaciones, evolucionó el concepto de repositorios de datos centrales (Data Warehouses) y también evolucionaron las técnicas de IA bajo el concepto de Inteligencia de Negocios (Business Intelligence) para utilizar esa información para fines analíticos y de toma de decisiones. Estas técnicas se conocen como técnicas de MD.

Las técnicas de MD se aplican principalmente mediante un enfoque consultivo, un profesional del tema es contratado por una organización para una aplicación específica de Inteligencia de Negocios. Este enfoque es útil, y produce muchos beneficios para las organizaciones, sin embargo, en el mercado de los sistemas de información, el incorporar "de entrada" técnicas de MD que puedan generar beneficios similares a los que se producen con la llamada inteligencia de negocios, sería una ventaja competitiva muy importante.

Las empresas productoras de sistemas de información integrales, deben aumentar sus ventajas competitivas. Ya no se trata solamente de resolver el problema de operación en los negocios, que incluyen organizaciones corporativas completas, sino de proporcionar elementos para análisis y toma de decisiones que hagan a la organización del cliente automáticamente más productiva.

Como un paso en esta dirección, se propone construir una herramienta que permita incorporar razonamiento Bayesiano en un sistema de información que opere en línea.

5.2. Enfoque de esta tesis

Esta tesis tiene un enfoque práctico: se desea que los sistemas de información que operan en línea, tengan acceso al uso de razonamiento Bayesiano en forma autónoma no supervisada. Para lograrlo, se propone construir un ambiente de trabajo que permita trasladar los beneficios del razonamiento Bayesiano al ámbito de los sistemas de información.

5.3. Hipótesis de investigación

La construcción del ambiente de trabajo (**BIframework**) descrito en la sección 6, permite que se puedan construir sistemas de información basados en razonamiento Bayesiano.

5.4. Prueba de hipótesis

Para probar la hipótesis, se implementarán dos técnicas de Inteligencia Artificial utilizadas en Minería de Datos, aprendizaje Bayesiano e inferencia Bayesiana, y se utilizarán mediante **BIframework** en un sistema de información para servicio a clientes que opera en línea.

La utilidad de **BIframework** se comprobará mediante la implementación en el sistema de información que opera en línea, del agente encargado de asignar asesores humanos para que atiendan los folios de servicio solicitado por los clientes, de tal forma que la asignación se realice para obtener la mejor evaluación en dos parámetros específicos: calificación del servicio proporcionado y tiempo requerido para proporcionar el servicio.

El Sistema de Información se representa mediante un programa en Java, que a través de BIframework utiliza un algoritmo de aprendizaje Bayesiano para aprender los criterios de asignación, y un algoritmo de inferencia Bayesiana para efectuar la asignación de asesores. Ambos algoritmos se construyen como parte de esta tesis conforme se explica en las secciones 3.5 y 4, están escritos en lenguaje 'c' y se pueden utilizar en el contexto de BIframework.

Desde el punto de vista del sistema de información, el agente aprende las probabilidades de una red bayesiana con estructura definida tomando como datos de entrada los registros históricos del servicio proporcionado por cada asesor a lo largo de 3 años de operación. Después de aprender, el agente hace la asignación mediante el proceso de inferencia, presentando como evidencia el nivel de evaluación y/o tiempo de servicio requerido.

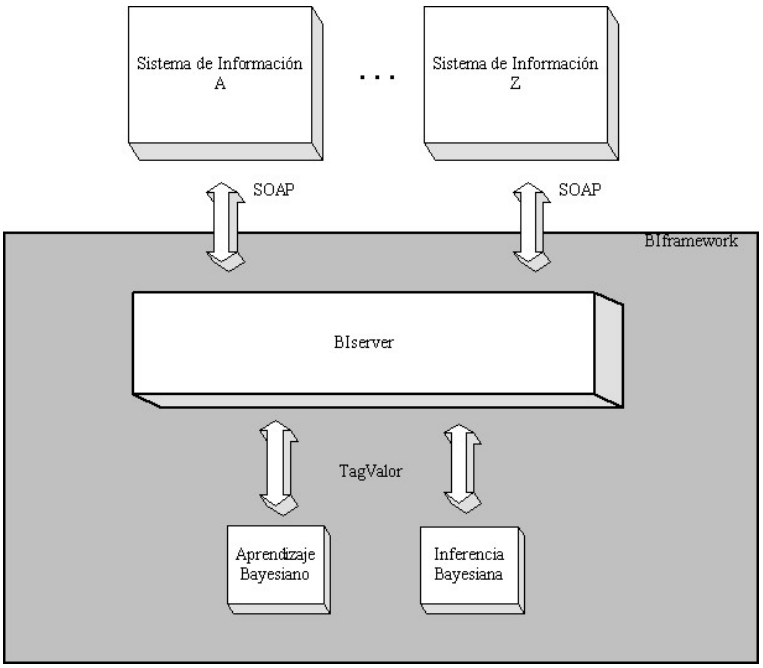


Figura 9: Arquitectura de BIframework

6. Solución propuesta: BIframework

BIframework se construye tomando como base el servidor central de aplicaciones de TCA (ver sección 5.1).

Se tomó el servidor de TCA como punto de partida y se adaptó para ser utilizado en este proyecto de tesis. Se aclara que este servidor tiene tres años de trabajo y evolución por parte de diversos ingenieros de TCA, razón por la cual es posible construir BIframework en el contexto de este proyecto de tesis.

BIframework, se construye utilizando una arquitectura de 3 capas:

1. Capa 1: Sistema de Información.
2. Capa 2: BIserver (Middleware).
3. Capa 3: Implementación de las técnicas de razonamiento Bayesiano.

En la figura 9 se muestra la arquitectura de BIframework.

6.1. Capa 1. Sistema de información

En esta capa es donde está implementado el Sistema de Información que opera en linea y se considera como cliente en el contexto de BIframework. Se implementan mecanismos de comunicación con el Middleware utilizando SOAP/XML.

Las aplicaciones de sistemas de información que pueden acceder a BIframework, pueden estar en cualquier plataforma y pueden estar implementadas como aplicaciones independientes, o como aplicaciones que requieren de un browser para funcionar. El único requisito es que tengan acceso al protocolo SOAP/XML.

Componentes de Comunicación

- Sockets Clientes
- Sockets Servidores
- CORBA
- SOAP/XML

Los 'clientes' se conectan directamente al Middleware o servidor de aplicaciones mediante sockets, que permiten hacer el enlace en cada uno de los puntos requeridos. Existen dos tipos de sockets: socket **cliente** y socket **servidor**. Los sockets cliente son están en esta parte, permitiendo establecer la conexión hacia el socket servidor en el Middleware. También se utiliza un socket servidor para que se esté ejecutando en espera de alguna petición del Middleware, permitiendo así una comunicación bidireccional en línea.

Para lograr estandarizar el envío de información sin importar qué sistema operativo está siendo utilizado en la capa del cliente, o en la capa del servidor, se utiliza el protocolo **SOAP/XML**. Su función es proporcionar un estándar de comunicación que sea independiente de la plataforma tecnológica utilizada.

Flujo de Datos
Los Datos son enviados mediante funciones de la arquitectura **CORBA**, que simplifica la programación escondiendo las funciones de comunicación, y optimiza transmisión y recepción de datos.

Los datos que van a través de la red hacia el Middleware tienen un formato estandarizado llamado SOAP/XML, que implica que los datos son empaquetados mediante el protocolo SOAP, estructurando la información en XML (Extensible Markup Language) que utiliza las ventajas del protocolo HTTP para la transmisión de los datos.

El Middleware recibe los datos en formato XML, realiza el 'parse' o traducción de XML a datos interpretables de más bajo nivel, los procesa y devuelve al cliente la información en el mismo formato SOAP/XML. De este modo, se logran Capas de 'Clientes' ejecutándose en cualquier plataforma compatible con SOAP/XML.

6.2. Capa 2: BIserver (Middleware)

El BIserver es la parte central de BIframework, es el "Middleware" del sistema completo, es la capa que sirve de enlace entre el sistema de información y los algoritmos de razonamiento Bayesiano.

Las funciones principales de BIserver son las siguientes:

- Administrar sesiones de usuarios que residen en la capa cliente (Sistema de Información).
- Administrar y controlar la ejecución de los algoritmos de razonamiento Bayesiano.
- Mantener sesiones vivas, hasta la desconexión del usuario en la capa cliente.
- Convertir una conexión inestable de HTTP, en una conexión permanente.

De esta forma se mantienen vigentes los "tubos de comunicación" necesarios para que un sistema de información acceda a las Técnicas de IA e información requeridas.

Se puede decir que la tarea principal de BIserver es actuar como "facilitador de técnicas de razonamiento Bayesiano" para los diversos usuarios firmados en las distintas capas de clientes.

El BIserver está preparado para atender a varios usuarios a la vez ya que cuenta con proceso de tareas en multi hilos (multi-thread), esto indica que puede procesar varios usuarios a la vez, sin necesidad de interrumpir o detener la tarea de cada uno para la capa cliente, el procesamiento de la información y el acceso a las técnicas de IA es totalmente transparente entre ellos.

6.2.1. Componentes de BIserver

Con el propósito de proporcionar un esquema escalable y multiplataforma, BIserver utiliza los siguientes componentes:

- Sockets
- CORBA/ORB
- SOAP/XML
- Componentes ASTA

Las funciones de cada uno de estos componentes es diversa pero en conjunto conforman un Middleware capaz de atender a usuarios en diferentes plataformas y en diferentes situaciones.

BIserver es capaz de atender a varios usuarios a la vez utilizando el mismo puerto de comunicación.

Sockets

Son los puntos extremos de una comunicación de dos vías (envío y recepción). Es el enlace entre dos aplicaciones ejecutándose en una red, de tal manera que sea posible la transferencia de datos entre ellas.

Hay dos tipos de sockets:

- Sockets Clientes
- Sockets Servidores

Estas tipos son utilizados para representar la conexión entre el Sistema de Información (Capa 1) y el BIserver o servidor de técnicas de IA (Capa 2).

CORBA/ORB

La capas 1 y 2 se comunican entre sí mediante componentes que implementan la funcionalidad de CORBA y SOAP/XML.

La arquitectura multi-capas permite la comunicación en línea entre diferentes entidades. Esto es posible con la implementación DRM (Distributed Resource Manager), agentes de red y el protocolo base CORBA. Se utiliza la tecnología OMC (Object Memory Caching) para mejorar el proceso de altos volúmenes de transacción, obtener un mejor desempeño y respuesta en línea. Se provee conectividad por medio de SOAP/XML.

CORBA: Common Object Request Broker Architecture, es una infraestructura emergente de interfases orientadas a objetos que permiten una comunicación a través de cualquier plataforma de una manera distribuida. Estandarizado por la OMG (Object Management Group) [11], CORBA automatiza muchas de las tareas comunes de la red como:

- registro de objetos.
- localización y activación.
- peticiones de multiplexado.
- envío de tramas y manejo de errores.
- empleo de marshalling y demarshalling (es el acto de organizar los datos de una computadora en una forma estándar, de tal forma que pueda ser leída por diferentes aplicaciones).
- operaciones de envío y respuesta eficientes.

los componentes primarios de la arquitectura CORBA son los siguientes:

Object: Es una entidad de programación CORBA, que consiste de una identidad, una interfase, y una implementación, que es conocida como Servant.

Servant - Es un lenguaje de entidad de programación implementado que define las operaciones que soportan una interfase IDL (Interface Definition Language) de CORBA. Los Servants pueden estar escritos en una variedad de lenguajes como: C, C++, Java, Smalltalk, Ada, entre otros.

Client - Invoca una operación a un objeto ORB. El acceso de servicios remotos debe ser transparente para el que está haciendo la petición (Cliente). Idealmente, debe ser una simple llamada a un método en un objeto.

Object Request Broker (ORB) - Provee un mecanismo para comunicar a los clientes hacia los servidores o Middleware de una forma transparente. El ORB simplifica la programación distribuida separando al cliente de los detalles de la invocación de métodos. Esto hace que los clientes aparenten que están haciendo llamadas a procedimientos locales. Cuando el cliente invoca una operación, el ORB es responsable de encontrar el objeto de implementación o bien, el servidor

que atenderá la petición del cliente, entregando ésta hacia el objeto remoto, y devolviendo la respuesta al cliente.

ORB Interface - Un ORB es una entidad lógica que debe ser implementada en varias formas (como uno o varios procesos o conjunto de librerías). Para separar las aplicaciones de los detalles de la implementación, la especificación de CORBA define una interfase abstracta para un ORB. Esta interfase provee varias funciones de ayuda, como convertir referencias de objetos a strings y viceversa, y creando listas de argumentos para peticiones hechas a través de invocaciones dinámicas de interfases.

CORBA IDL stubs and skeletons - Actúan como un enlace entre el cliente, el servidor de aplicaciones y el ORB. El uso de un compilador reduce el potencial para las inconsistencias entre clientes stubs y servidores skeletons e incrementa oportunidades para la optimización automatizada de los compiladores.

Dynamic Invocation Interface (DII) - Esta interfase permite al cliente accesar directamente los mecanismos fundamentales de la petición realizados por un ORB. Las aplicaciones utilizan el DII para emitir peticiones dinámicamente hacia los objetos sin necesidad de requerir una interfase IDL. Los DII también permiten a los clientes hacer operaciones asíncronas y envíos en una dirección (de envío solamente).

Dynamic Skeleton interface (DSI) - Es el lado del servidor, análogo al DII del lado del cliente. El DSI permite a un ORB entregar peticiones a un servidor de aplicaciones que no tiene conocimiento del tipo de objeto que está implementando la petición. El cliente que hace la petición no tiene idea si la implementación está utilizando un tipo específico de IDL skeletons o si está usando skeletons dinámicos.

Object Adapter - Este asiste al ORB con las entregas de las peticiones hacia los objetos. Más importante aún, un Object Adapter asocia implementaciones de objetos con el ORB. Los Object adapters pueden ser especializados para proveer soporte para ciertos estilos de implementación de objetos.

SOAP/XML

SOAP se basa en el protocolo http como un mecanismo de transporte para enviar mensajes en formato XML. Los mensajes son empacados en lo que se llama estructura SOAP y enviados hacia un servidor para que los procese en una forma de Petición/Respuesta. SOAP, distinto a los protocolos DCOM o RMI, no requiere una fuerte conexión ente el cliente y el servidor.

La ventaja de SOAP se basa en la comunicación entre diferentes plataformas, esto permite que los extremos que se estén comunicando no necesariamente tengan que estar programados en los mismos lenguajes, o bien, que sistemas incompatibles puedan interoperar con éxito.

Debido a que SOAP trabaja en http, obtiene todos los beneficios de seguridad que están disponibles para el protocolo http, los mensajes SOAP pueden pasar a través de un firewall de un servidor de web por medio del puerto 80.

ASTA

Cuando un cliente en Browser se conecta al BIserver lo hace por medio del Servidor SOAP ASTA y el HTTP Listener de ASTA, éstos a su vez ejecutan los Métodos requeridos por el usuario mediante el objeto Métodos SOAP de ASTA.

6.2.2. Flujo de datos

Cuando llega una petición a BIserver por medio del protocolo SOAP, los objetos ORB (Object Request Broker) se encargan de administrar cada una de las peticiones entrantes y se cercioran de que BIserver haya recibido esa petición. Cuando BIserver desee mandar la respuesta al cliente, también se hace por medio de los agentes ORB, permitiendo así las comunicación desde el Cliente al BIserver y viceversa.

Cada componente en particular realiza una tarea específica, y estas tareas terminan cuando el usuario termina su conexión con el BIserver. Mientras que el usuario esté accesando las técnicas de razonamiento Bayesiano a través de BIserver, éste mantiene su sesión activa en memoria, realizando ejecuciones y consultas a los algoritmos de razonamiento Bayesiano implementados, conforme el cliente lo vaya requiriendo.

6.2.3. Mensajes SOAP del sistema de información a BIserver

Para darle "significado" a los mensajes SOAP que fluyen entre el Sistema de Información y BIserver, se utiliza un protocolo para etiquetado de valores y funciones que se llama TagValor.

En el protocolo TagValor los mensajes se componen de 4 partes:

```
<ns1></ns1>                 MÉTODO REMOTO
<UserParams></UserParams>   PARÁMETROS
<MethodName></MethodName>   NOMBRE DEL MÉTODO A EJECUTAR
<Parametro1></Parametro1>   PARÁMETROS
...
<ParametroN></ParametroN
```

6.2.4. Funciones de BIserver

BIserver se compone de 4 funciones principales que son las que se invocan remotamente por los Clientes (Sistemas de Información 'usuarios') para ejecutar las técnicas Bayesianas. Las funciones son:

- Ejecuta-Dll()
- Libera-Dll()
- AlmacenaDato()
- EjecutaProceso()

Estas funciones son las que interactúan directamente con el protocolo TagValor, ya que de esta manera leen las peticiones efectuadas por las Aplicaciones y devuelven la respuesta en el mismo formato.

En el anexo A se explica el uso de cada función de BIserver.

6.3. Capa 3: Implementación de las técnicas de razonamiento Bayesiano

6.3.1. Estructuras para retorno a BIserver

Cuando las Técnicas de razonamiento Bayesiano ejecutan las funciones solicitados desde la Capa 1, devuelven un valor de una serie de posibles valores de retorno:

```
<@Result_EjecutaDll></@ Result_EjecutaDll>          Retorno de EJECUTA-DLL
<@Result_LiberaDll></@ Result_LiberaDll>            Retorno de LIBERA-DLL
<@Result_AlmacenaDato></@ Result_AlmacenaDato >     Retorno de ALMACENAR
<@Result_EjecutaProceso></@Result_EjecutaProceso >  Retorno de EJECUTAR
```

Estas cuatro directivas contienen una de los siguientes 6 formatos de resultados:

```
<@dat></@dat>       Regresa información en formato de Tag-Valor.
<@err></@err>       Cuando ocurre un error regresa el número del
                    mismo y su descripción.
<@dsz></@dsz>       Formato utilizado para
                    desplegar información en modo catálogo o reporte.
<@imp></@imp>       Impresión de archivo.
<@eje></@eje>       Manda a ejecutar otro proceso al servidor.
<@dsd></@dsd>       Devuelve @dsz y @dat.
```

6.3.2. Protocolo TagValor

En TCA, se creó un protocolo de comunicación con el objetivo de que las diferentes plataformas en la Capa Cliente (Thin Clients), se puedan comunicar con las Reglas de Negocio contenidas en su repositorio central de aplicaciones de software. Este protocolo se ha adecuado para ser utilizado por el BIserver para permitir la comunicación entre la capa del cliente y la capa de las Técnicas de IA, que conforman el núcleo del sistema de razonamiento Bayesiano.

El protocolo TagValor es utilizado para que BIserver entienda qué es lo que necesita hacer la Capa Cliente, y para que el Cliente entienda qué requiere la Técnica de IA para que los procesos o funciones sean ejecutadas acorde al flujo de información requerido.

El protocolo de comunicación entre las Técnicas de IA y la Capa Cliente de la Tecnología Multi-Capas funciona con directivas de Tag-Valor, donde:

Tag es un número que identifica a un componente en específico, este puede ser un campo de edición, un contenedor, un botón o cualquier otro componente que se encuentre en las formas de la Capa Cliente.

Valor es la instrucción que le indicará a la Capa Cliente qué es lo que debe hacer con el componente enviado en la directiva Tag.

6.4. Programación y prueba del algoritmo de Aprendizaje

En el sitio de internet http://www.tca.com.mx/∼arobles/tesis se puede obtener una copia del algoritmo de aprendizaje implementado según se explica en la sección 3.5.1.

6.5. Programación y prueba del algoritmo de Inferencia

En el sitio de internet http://www.tca.com.mx/∼arobles/tesis se puede obtener una copia del algoritmo de aprendizaje implementado según se explica detalladamente en la sección 4.

6.6. Caso ejemplo de razonamiento Bayesiano

Para probar el funcionamiento de los algoritmos de Razonamiento Bayesiano implementados en este proyecto, se diseñó y se procesó un caso de Ejemplo. La metodología es la siguiente:

- Considerando las dependencias entre variables, se generaron datos de entrada en forma aleatoria (ver anexo D.0.2).
- Tomando como entrada la estructura de la red ejemplo (ver anexo D.0.1) y los registros generados (ver anexo D.0.2), se ejecuta el algoritmo de Aprendizaje Bayesiano para generar las tablas de probabilidad aprendidas que se muestran en la anexo D.0.3.
- Se toma como entrada la estructura de la red Bayesiana y las tablas de probabilidades aprendidas, para ejecutar el Algoritmo de Inferencia, se da entrada a las observaciones y se muestran los pasos intermedios y final del proceso de razonamiento. Las etapas del proceso más relevantes son las siguientes:
  - Grafo triangulado y cliques (anexo D.0.5)
  - Estructura de árbol conjunto (anexo D.0.6)
  - Proceso de inferencia (anexo D.0.7)

6.7. Composición del ambiente de trabajo

6.7.1. Integración del sistema de información en la arquitectura

En el Anexo B se muestra el detalle del código fuente que integra el Agente de Asignación del Sistema de Información a BIframework. Se observa que la comunicación con BIserver se realiza utilizando SOAP como se explicó en la sección 6.1.

6.7.2. Integración de los algoritmos Bayesianos en la arquitectura

En el Anexo C se muestra el detalle del código fuente que integra los algoritmos de razonamiento Bayesiano a BIframework. Se observa que la comunicación se realiza utilizando el protocolo TagValor explicado en la sección 6.3.2 y regresa las estructuras para datos explicadas en la sección 6.3.

Parte III
Caso de estudio

7. Caso de estudio

Para la prueba de hipótesis, se implementó un Sistema que represente un agente autónomo ejecutando el método de asignación utilizando razonamiento Bayesiano. La implementación se realizó en base a las siguiente metodología:

- Se efectuó el análisis y diseño del Sistema según se explica en la sección 7.1.
- Se programaron los algoritmos de razonamiento Bayesiano.
- Se implementó el método de asignación de asesores utilizando las técnicas de razonamiento Bayesiano mediante BIframework. Para lograr mejorar la calidad del servicio, se asignan los asesores que según los requerimientos de asignación (mejor evaluación y mejor tiempo de servicio), mostrados como evidencia, sean los que resulten con mayor probabilidad de éxito en el proceso de inferencia Bayesiana. Por tal motivo se puede afirmar que dada la historia de servicio, *se logra la asignación que asegura la mejor evaluación del cliente.*
- La interacción entre el agente 'usuario' y las técnicas Bayesianas se realiza exclusivamente mediante **BIframework**.

7.1. Análisis y diseño del sistema

El proceso de análisis se hizo siguiendo la metodología de CommonKADS [9], y para la especificación de los agentes implicados se utilizó MAS-CommonKADS [10].

Para efecto de delimitar el alcance de la funcionalidad del sistema, y establecer claramente el rol de la implementación del método de Asignación con aprendizaje y razonamiento Bayesiano, se incluyen las siguiente figuras:

- Diagrama de casos de uso. Figura 10. En este diagrama se muestra en gris la inferencia de Asignación implementada utilizando el algoritmo de razonamiento Bayesiano.
- Conocimiento del dominio: Diagrama de clases (ontología). Figura 11.
- Modelo de tareas. Figura 12.
- Reglas de negocio. Figura 13.

7.2. Alcance del sistema de aprendizaje e inferencia construido

Para el método de asignación, se implementó un algoritmo de aprendizaje Bayesiano que es utilizado para aprender los criterios de asignación de asesores a folios de servicio, de tal forma que aprenda que asesor asignar a cada folio dependiendo de las características del servicio solicitado.

El método para que el agente controle las inferencias se muestra en el diagrama de actividad para el método de asignación en la figura 14. Como puede observarse en la figura, con el uso

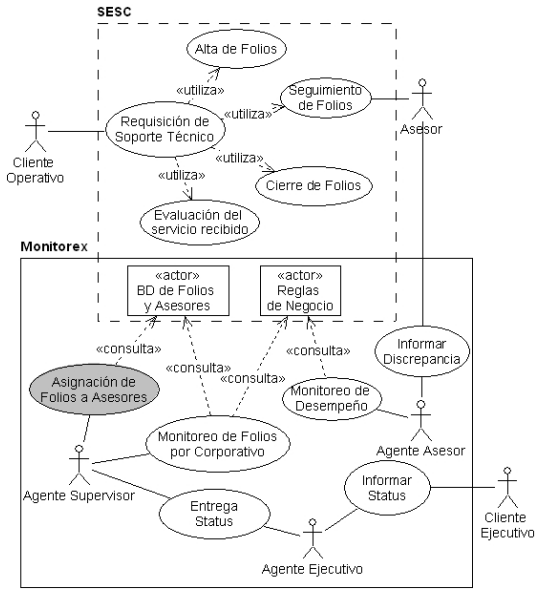


Figura 10: Diagrama de casos de uso.

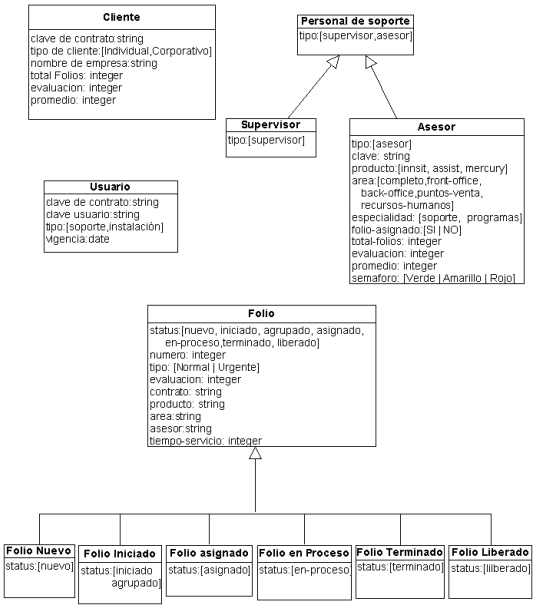


Figura 11: Conocimiento del dominio: diagrama de clases (ontología).

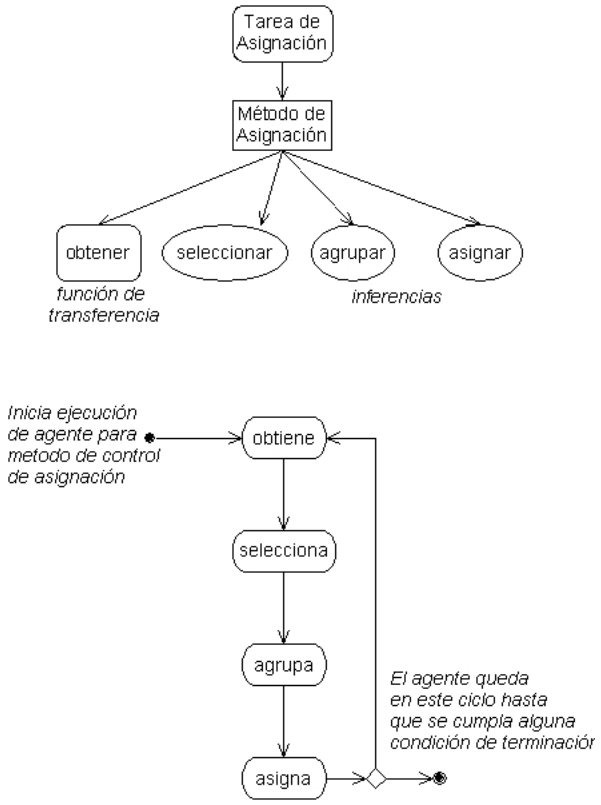


Figura 12: Descomposición y diagrama de control de la tarea de asignación

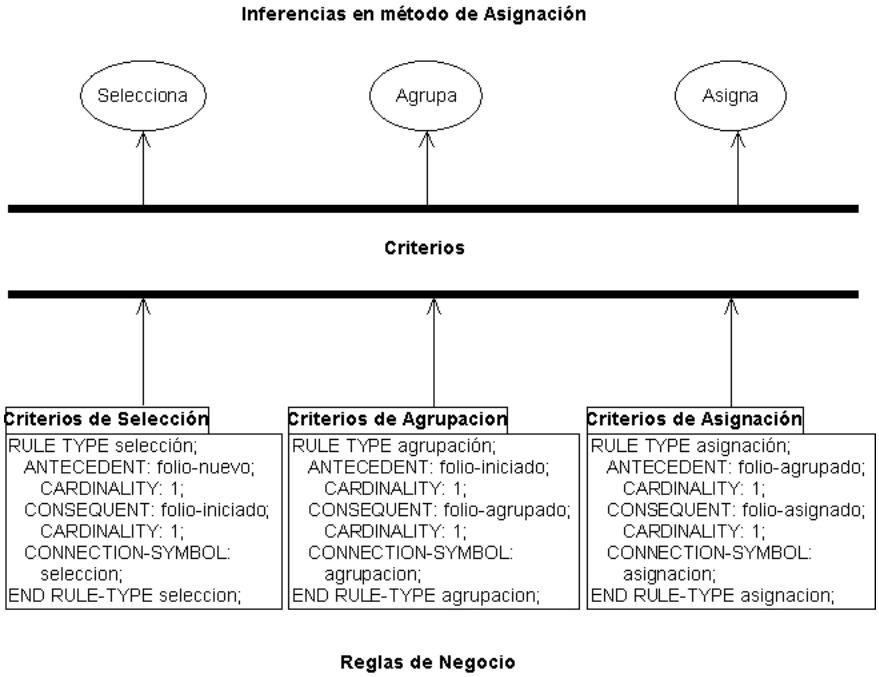


Figura 13: Reglas de negocio para inferencias del método de asignación

de **BIframework** se integra el razonamiento Bayesiano para que sea utilizado por el sistema de información como si fuese parte integral del mismo. Este ambiente de trabajo, permite que el sistema de información se construya basado en técnicas de razonamiento Bayesiano, sin que el programador del sistema de información tenga que preocuparse de los detalles de implementación de los algoritmos Bayesianos.

La interacción se realiza en línea tomando todo tipo de consideraciones en el Sistema, mezclando criterios de Sistemas de Información con las recomendaciones específicas de las técnicas de razonamiento Bayesiano, en este caso, con las recomendaciones de asignación.

La posibilidad de tener acceso a las técnicas de razonamiento Bayesiano desde el agente 'usuario' es muy relevante, ya que permite que el sistema aspire a maximizar la evaluación otorgada por el cliente que solicita el servicio. Sin el elemento de aprendizaje ésto no sería posible ya que no se tiene una situación estática sobre la cual evaluar una función objetivo, es decir, las condiciones del criterio de evaluación son dinámicas ya que se modifican con cada folio evaluado por el cliente.

El beneficio de incorporar aprendizaje y razonamiento Bayesiano no se limita a maximizar la evaluación, sino que también se eliminan funciones de supervisores humanos debido a que el agente 'usuario', es capaz hacer asignaciones equivalentes a la heurística del humano.

7.3. Red Bayesiana generada

La red utilizada en este proyecto se observa en la figura 15. Las tablas de probabilidad conjunta se obtuvieron en base a los datos históricos de tres años de operación del sistema. Se tomaron 9,000 registros referentes a folios de servicio correspondientes al área de productos de software para turismo de TCA.

La estructura de la red Bayesiana se proporciona como entrada al algoritmo de aprendizaje, está compuesta por 23 nodos, de los cuales 11 corresponden a los asesores disponibles para servicio en la división de turismo, es decir objetivos de 'búsqueda' para la red. En la red se tienen sólo variables discretas.

Debido a las relaciones existentes entre los datos de la red, la tabla más grande de probabilidad conjunta generada fue de 64 entradas. Se tienen dos nodos de rango 9 [8], es decir, con 9 aristas incidiendo en la misma dirección (salida).

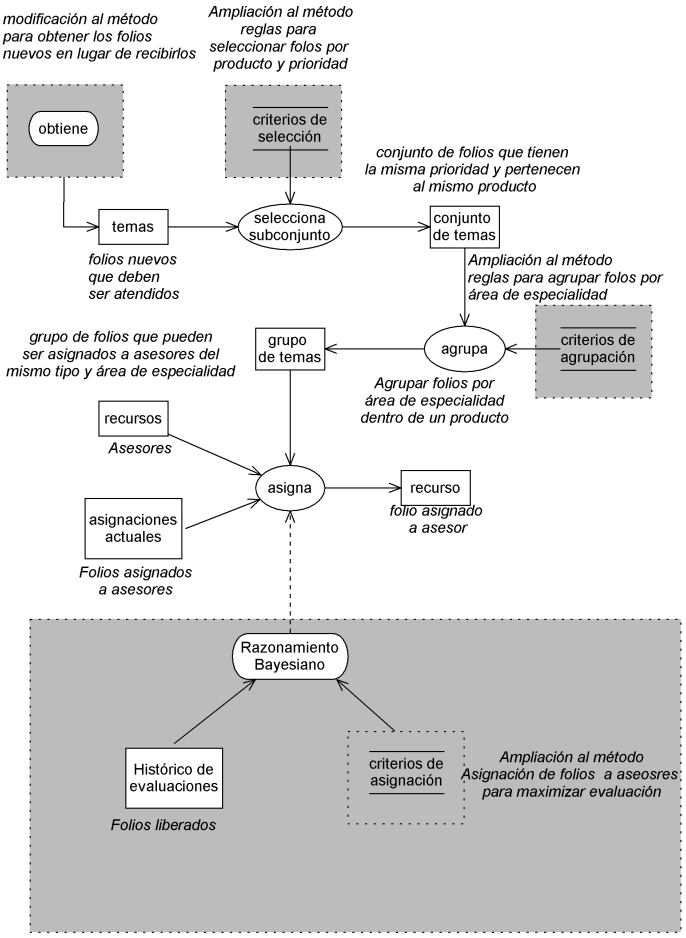


Figura 14: Diagrama de actividad para método de asignación con algoritmo de aprendizaje.

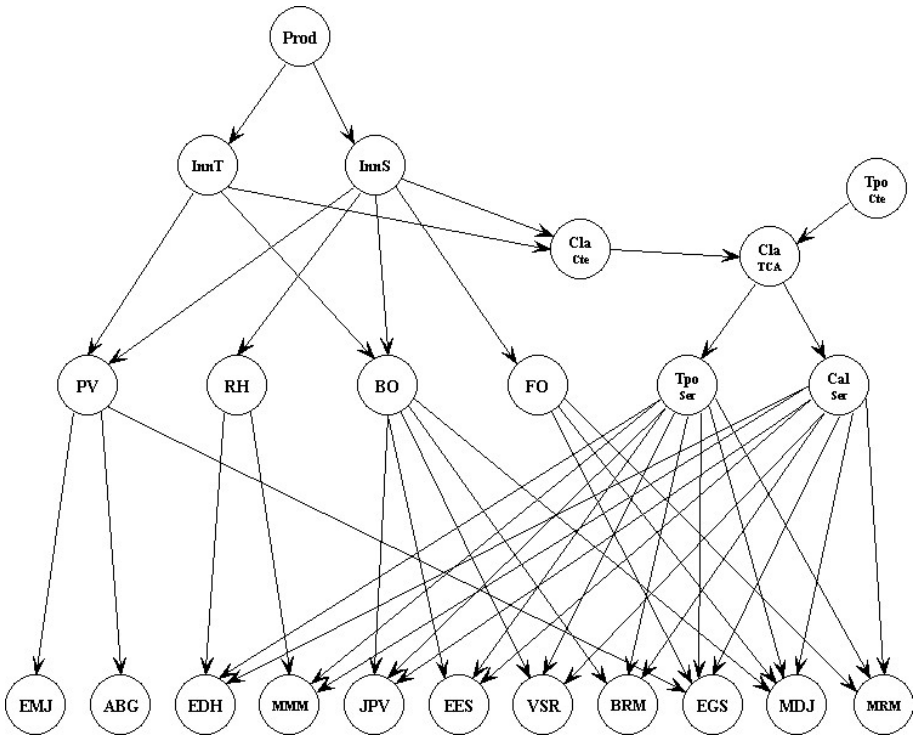


Figura 15: Red Bayesiana para aprendizaje e inferencia

8. Resultados obtenidos

Con la implementación del método de asignación utilizando técnicas de razonamiento Bayesiano mediante BIframework, se logró dar inteligencia "empaquetada" a un Sistema Información que opera en línea. Este resultado era el esperado, ya que indica que se puede incorporar razonamiento Bayesiano mediante BIframework a los sistemas de información de uso industrial.

Sin el ambiente de trabajo que proporciona BIframework, para que el sistema de información pueda utilizar las técnicas de razonamiento Bayesiano, se deben realizar al menos las siguientes actividades:

1. Elegir o programar los algoritmos de razonamiento Bayesiano.
2. Programar una interfase en el sistema de información para incorporar el uso de los algoritmos.
3. Elegir o programar algún método de coordinación entre las técnicas de razonamiento Bayesiano y el sistema de información.

Si se observa detenidamente la cantidad de trabajo que implica cada actividad y se considera solamente una implementación de BIFramework, en el mejor de los casos sin utilizar BIframework, el trabajo requerido es equivalente, pero se deben cuidar los detalles de sincronización y control, para que el sistema de información y la técnica Bayesiana interactúen sin problemas. Cuando se desee utilizar razonamiento Bayesiano desde otro sistema de información - que pudiera estar programado en otro lenguaje -, se debe repetir gran parte del paso 2 y todo el paso 3. Con estas

observaciones, se puede concluir, que BIframework es un ambiente útil que facilita el desarrollo de sistemas de información basados en razonamiento Bayesiano.

En el contexto de la problemática expuesta en la sección 5.1, se puede concluir que BIframework es una excelente opción para evolucionar los sistemas de información de uso industrial, ya que se **automatiza** la toma de decisiones en forma **autónoma**, lo que tiene un valor agregado en sí mismo, pero adicionalmente se proporciona **consistencia** en la toma de decisiones, se promueve el uso de mas información en forma sistemática a la que un ser humano normal puede manejar, y se aumenta la **efectividad** en la toma de decisiones. Se debe señalar también, que aún hay deficiencias que quizá resulte imposible abordar, por ejemplo, por ahora resulta imposible reemplazar el factor de sentido común, que solamente los seres humanos podemos explotar.

Respecto a las Técnicas Bayesianas programadas, ambas cumplieron con su misión, aprender y hacer inferencias correctas. De los 9,000 registros mencionados en la sección 7.3, 7,000 se utilizaron para aprender las dependencias involucradas en el proceso de asignación de asesores, 2,000 se utilizaron para "probar" las asignaciones generadas por el algoritmo de inferencia. Se encontró que el 100 % de los casos de prueba fueron asignados correctamente. Estos resultados permiten asegurar que el sistema completo efectivamente ayudará a obtener las mejores evaluaciones por el servicio proporcionado a los clientes, ya que el sistema de información basado en razonamiento Bayesiano, asigna al mejor asesor considerando la evidencia manejada como objetivo de búsqueda: mejor evaluación y menor tiempo de servicio.

Parte IV
Conclusiones

9. Conclusiones

BIframework resultó ser un ambiente de trabajo útil para la integración de información y automatización de la toma de decisiones, ya que se implementó con éxito en el sistema de información del caso de estudio, la inferencia de asignación utilizando razonamiento Bayesiano.

Con el propósito de delimitar el alcance de BIframework, este proyecto de tesis se desarrolló considerando únicamente razonamiento Bayesiano. Los resultados obtenidos indican que la idea de construir sistemas de información basados en razonamiento Bayesiano, es correcta. Los conceptos y metodología desarrollados en este proyecto, se pueden extender a otras técnicas de IA, formando un "repositorio" central de técnicas implementadas, para que puedan ser utilizadas en el desarrollo de sistemas de información en le contexto propuesto por BIframework.

Se observó que el problema del caso de estudio, aunque desde el punto de vista de sistemas de información es amplio, resultó ser un problema fácil para los algoritmos Bayesianos programados, lo que hace suponer, que el rango de aplicaciones que se pueden implementar en sistemas de información mediante BIframework es amplio.

10. Trabajo futuro

Este proyecto de tesis está planteado como la primera parte de un proyecto mayor que será mi proyecto de tesis doctoral. Los resultados obtenidos fueron satisfactorios, sin embargo, queda mucho por hacer. Algunos de los temas que se incorporarán a BIframework a futuro son:

- Manejo de Ontologías, para promover la especialización y evitar las definiciones redundantes.
- Definir a BIframework como un agente, pudiendo interactuar varios en un ambiente multiagentes, quizá para la solución colaborativa de un problema.
- Evaluación competitiva de modelos, para que sea BIframework quien decida que técnica utilizar ante un problema dado.
- Implementar un repositorio central de técnicas de IA según se sugiere en la sección 2.2.
- Control de ejecución de reglas de negocio utilizando sistemas expertos, para supervisar sistemáticamente el cumplimiento de las normas establecidas.
- Monitoreo de desempeño de funciones de negocio utilizando lógica difusa, para codificar adecuadamente las métricas de desempeño utilizadas por el ser humano.

Parte V
Apéndices

A. Uso de funciones de BIserver

Ejecuta-Dll()

Su función principal es ejecutar un programa dll (Dynamic Link Library) y cargarlo en la memoria de BIserver con el fin de mantener disponibles las direcciones de memoria de los procedimientos de enlace que se incluyen en la implementación de cada técnica de IA. Los procedimientos de enlace son:

- Entrada: Define las variables de inicialización y prepara al dll para ejecutar la Técnica de IA Implementada.
- AlmacenaDato.
- EjecutaProceso.

Una vez conocidas las direcciones de estos procedimientos, se almacenan en una tabla interna de BIserver, donde se hace un mapeo de que direcciones de memoria corresponden a cada Técnica de IA, y a qué usuario están direccionadas.

Cuando se ejecuta el procedimiento Ejecuta-Dll, se crean nuevos registros en las tablas de memoria de BIserver:

- Usuarios
- Programas
- DireccionesProgramas

Estos registros sirven para llevar un directorio de Usuarios vs Técnicas de IA, para poder administrar sus sesiones y tener información relevante para cuando BIserver necesite comunicarse con él.

Libera-Dll()

Este procedimiento libera de la memoria de BIserver alguna Técnica de IA que se haya cargado previamente y ya no se requiera.

Cuando se ejecuta este método, se actualizan las tablas de Programas y DireccionesProgramas para determinar que la Técnica de IA que está liberando, quede efectivamente eliminada de estas tablas y llevar así un registro de lo ocurrido.

Cuando el usuario libera todos sus programas dll de BIserver, se invoca inmediatamente la desconexión del mismo, debido a que ya no tiene nada más con qué interactuar.

AlmacenaDato()

Se ejecuta este procedimiento para almacenar en memoria algún dato relevante para el Sistema de Información en la Capa Cliente, como por ejemplo, los campos contenidos en un archivo de datos producidos por la operación de algún sistema, y que se desee pasar al algoritmo de

aprendizaje para que los utilice para 'aprender' probabilidades. Cuando estos datos se 'mandan almacenar' son grabados en memoria para uso de la Técnica de IA requerida.

EjecutaProceso()

Este procedimiento sirve para ejecutar un proceso dentro de la Técnica de IA.

Estos procesos son aquellos que tienen la función de realizar alguna consulta hacia una base de datos, o bien, guardar los datos que ya habían sido almacenados en memoria por el procedimiento AlmacenaDato(), o bien, algún otro proceso que se requiera para facilitar el uso de alguna técnica de IA.

Cuando finaliza la ejecución de alguna de las rutinas descritas anteriormente, pueden devolver valores generados por las Técnicas de IA, que tendrán un formato acorde al protocolo TagValor.

B. Detalle de integración del sistema de información en BIframework

```
public void sendData(){
  String prog = "Infer";                              // nombre del programa
  String proceso = "5";                               // Numero de proceso
  TCASOAPMessage soapMessage=new TCASOAPMessage();
  soapMessage.methodName = 5;                         //ejecuta
  soapMessage.params[0] = prog;
  soapMessage.params[1] = "";
  soapMessage.params[2] = proceso;
  soapMessage.params[3] = (String)jProduct.getSelectedItem();
  soapMessage.params[4] = (String)tContrato.getText();
  soapMessage.params[5] = (String)tCuenta.getText();
  soapMessage.params[6] = (String)tPassword.getText();
  soapMessage.params[7] = (String)tTitulo.getText();
  soapMessage.params[8] = (String)tDescripcion.getText();
  soapMessage.params[9] = (String)jArea.getSelectedItem();
  System.out.println(
      "PARAMETRO PARA EJECUTAR EL PROCESO -"+soapMessage.params[3]+"_");
  soapMessage.numParams = 4;
  soapMessage.windowParent = null;
  java.util.List resp = tcaService.callService(soapMessage);
  if (resp==null || resp.size() == 0) {
      System.out.println(
          "El vector de respuesta para ejecutar el catalogo es nulo ");
      return;
  }
  String codRet = tcaService.getCodRet();
  String msg = null;
  if (resp! = null && resp.size() >= 1) { msg = (String)resp.get(0);}
  if (codRet == null || codRet.compareTo("@err") == 0){
  if (msg != null){
          JOptionPane.showMessageDialog(this , msg );
  }
     resp = null;
     return ;
  }
}
```

C. Detalle de integración de razonamiento Bayesiano en BIframework

```
/**************************************************/
/*            INTERFASE CON BIserver              */
/**************************************************/
__declspec (dllexport)
char * AlmacenaDato(TagDato, param)
int    TagDato;
char * param;
{
 campos = 0;
 val_err = 0;
 libera_dat = 'N';
 switch(TagDato)
 {
   case 1:
     break;
   case 2:
           val_err = RutinaPrueba(TagDato, param);
     break;
   default:
     val_err=901;
     break;
 }

 switch(val_err)
 {
   case NO_ERROR:
     sprintf(regresa,"OK");
     break;
   case IMPRIME:
     sprintf(regresa, "@imp001|%5.5d|%s|%s|%c",TagDato,token,nom_arch_imp,libera_dat);
     break;
   case WINDESPZ:
     sprintf(regresa, "@dsz001|%5.5d|%d|%c",TagDato,&ValoresWINdespz,libera_dat);
     break;
   case DESPDATOS:
     sprintf(regresa, "@dat001|%3.3d|%3.3d|%d",TagDato,campos,&ValorCampos);
     break;
   default:
     sprintf(regresa, "@err001|%3.3d|%3.3d|%s",TagDato,val_err,TextoError);
     break;
 }
 campos = 0;
 return(regresa);
```

```
}

/*****************************************************/
/*              Inicio de EjecutaProceso             */
/*****************************************************/

__declspec (dllexport)
char *EjecutaProceso(IdProceso, NumParam, ListParam)
int   IdProceso;
int   NumParam;
char *ListParam[];
{
 campos = 0;
 val_err = 0;
 libera_dat = 'N';

 switch (IdProceso)
 {
   case CIERRA_CONEXIONES :
       val_err = CierraConexiones();
       break;
   case 10 :
       val_err = Infer(NumParam, ListParam);
       break;
   default:
       val_err=901;
       break;
 }

 switch (val_err)
 {
   case NO_ERROR:
     sprintf(regresa, "OK");
     break;
   case IMPRIME:
     sprintf(regresa, "@imp001|%5.5d|%s|%s|%c",IdProceso,token,nom_arch_imp,libera_dat);
     break;
   case WINDESPZ:
     sprintf(regresa, "@dsz001|%5.5d|%d|%c",IdProceso,&ValoresWINdespz,libera_dat);
     break;
   case DESPDATOS:
     sprintf(regresa, "@dat001|%3.3d|%3.3d|%d",IdProceso,campos,&ValorCampos);
     break;
   default:
     sprintf(regresa, "@err001|%3.3d|%3.3d|%s",IdProceso,val_err,TextoError);
     break;
 }
```

```
 campos = 0;
 return(regresa);
}

__declspec (dllexport)
int  LiberaMemoria(Memoria)
char *Memoria;
{
 if((char *) Memoria == (char *) NULL)
 {
   return(1);  //No se recibió dirección
 }
 else
 {
   free(Memoria);
   return(0);
 }
}

/*****************************************************/
/*       EXPORTACION DE FUNCIONES DE INTERFASE       */
/*****************************************************/

__declspec (dllexport)
char *Entrada(int argc, char *argv[])
{
 campos = 0;
 val_err=TCAmain(argc, argv);
 if (val_err == 0)
   val_err = CargaDefaults();

 switch (val_err)
 {
   case NO_ERROR:
     sprintf(regresa, "OK");
     break;
   case DESPDATOS:
     sprintf(regresa, "@dat001|%3.3d|%3.3d|%d",0,campos,&ValorCampos);
     break;
   default:
     sprintf(regresa, "@err001|%3.3d|%3.3d|%s",0,val_err,TextoError);
     break;
 }
 campos = 0;
   return(regresa);
}
```

```
/*****************************************************/
/* INICIA IMPLEMENTACION DE LA TECNICA DE IA         */
/* Autor.  Armando Robles                            */
/* Algoritmo de Inferencia Bayesiana.                */
/*****************************************************/

int Infer(NumParam, ListParam)
int NumParam;
char * ListParam[];
{
   Mensaje("Inicio Infer");
   strcpy(red, ListParam[0]);
   strcpy(query_entrada, ListParam[1]);

   for (i=0; i<strlen(query_entrada); i++)
       if (query_entrada[i]=='+') query_entrada[i]='|';
 .
 .
 .
```

D. Caso ejemplo de razonamiento Bayesiano con los algoritmos implementados en BIframework

D.0.1. Estructura de la red Bayesiana

La estructura de la Red Bayesiana se da como entrada al algoritmo por medio de un archivo tipo texto. En el caso ejemplo se llama: 'demred.txt'. Su contenido es el siguiente:

```
#1,%1,n=A,h2,h3,v=on,v=off
#2,%2,n=B,p1,h4,v=on,v=off
#3,%3,n=C,p1,h5,h7,v=on,v=off
#4,%4,n=D,p2,h6,v=on,v=off
#5,%5,n=E,p3,h6,h8,v=on,v=off
#6,%6,n=F,p4,p5,v=on,v=off
#7,%7,n=G,p3,h8,v=on,v=off
#8,%8,n=H,p5,p7,v=on,v=off
```

La primera columna indica el número de nodo, la segunda indica el número de columna en el conjunto de datos de entrada de donde se tomará el valor para el nodo. $n = B$ indica que el nombre del nodo es B, $p1$ indica que este nodo tiene comp padre al nodo número 1 (en esta caso el nodo A), $h4$, indica que este nodo tiene como hijo al nodo 4. Los valores que puede tomar este nodo se indican como $v = on$ y $v = off$.

D.0.2. Generación de datos de entrada

Para generar los datos de entrada del caso ejemplo, se elaboró un programa en 'Matlab' [17] que mediante el uso de la función 'random' generó las probabilidades para cada nodo en cada registro. Las dependencias entre las variables se simularon mediante estatutos condicionales. El archivo de datos generado contiene 1000 registros y tiene la siguiente forma:

```
on ,off,on ,on ,on ,off,on ,off
off,on ,off,on ,on ,off,off,on
on ,off,on ,off,off,on ,on ,on
off,off,off,off,on ,off,off,on
off,off,off,off,off,on ,off,on
              .
              .
              .
on ,off,on ,on ,off,off,off,on
off,on ,off,on ,off,off,off,on
on ,off,off,off,on ,off,off,on
off,off,off,off,on ,off,off,on
```

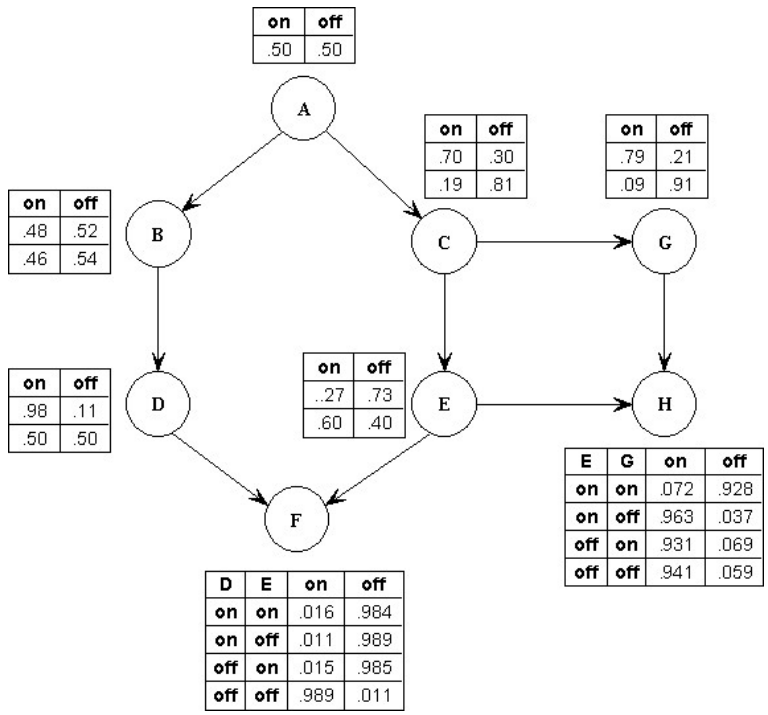


Figura 16: Red Bayesiana mostrando probabilidades aprendidas

D.0.3. Probabilidades aprendidas

En la figura 16 se muestra la representación gráfica de la Red Bayesiana ejemplo. Se muestran también las probabilidades aprendidas al ejecutar el algoritmo de aprendizaje.

D.0.4. Grafo moral

El programa lee el archivo con la estructura de la red Bayesiana, para efectuar la representación interna mediante una matriz de adyacencias. Las n variables de entrada se representan con n nodos numerados de 0 a $n-1$. Los valores $G_{ij} \neq 0$ indican conexión entre los nodos i, j del Grafo. Si $G_{ij} > 1$ indica que la dirección de la arista es de i hacia j, y si $G_{ij} = -1$ indica que la dirección es de j hacia i. El grafo generado en el ejemplo es el siguiente:

```
********* Grafo Original ***********************
    0  1  2  3  4  5  6  7
0   0  1  1  0  0  0  0  0
1  -1  0  0  1  0  0  0  0
2  -1  0  0  0  1  0  1  0
3   0 -1  0  0  0  1  0  0
4   0  0 -1  0  0  1  0  1
5   0  0  0 -1 -1  0  0  0
6   0  0 -1  0  0  0  0  1
7   0  0  0  0 -1  0 -1  0
```

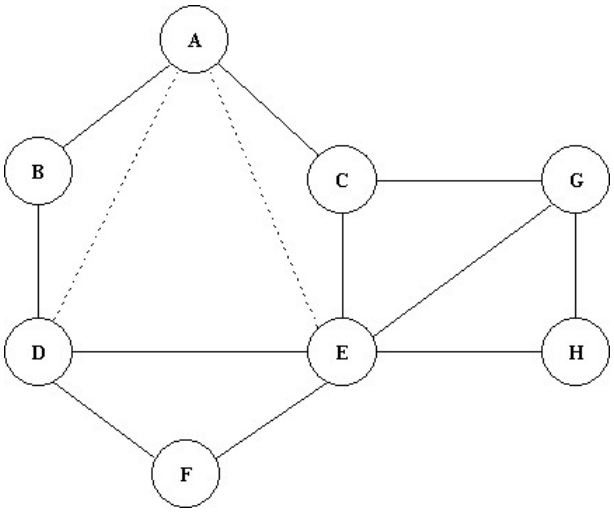


Figura 17: Grafo triangulado

Partiendo del Grafo Original, se efectúan las transformaciones necesarias para producir el Grafo Moral. Como las aristas no tienen dirección, el Grafo Moral se representa por medio de una matriz de adyacencias triangular, donde el valor G_{ij} indica la distancia entre el nodo i y el j.

```
********** Grafo Moral **********************
    0  1  2  3  4  5  6  7
0   0
1   1  0
2   1  0  0
3   0  1  0  0
4   0  0  1  2  0
5   0  0  0  1  1  0
6   0  0  1  0  2  0  0
7   0  0  0  0  1  0  1  0
```

D.0.5. Grafo triangulado y cliques

Se utiliza el grafo moral para efectuar el proceso de triangulación, se forman los clusters y se elimina en cada iteración un vértice y se inducen aristas cuando es necesario. En el proceso de triangulación, se identifican también los cliques. El grafo triangulado obtenido se muestra en la figura 17.

El programa despliega los cliques formados y las aristas inducidas:

```
====================================
```

```
 *** CLIQUES ***
clique No. 1: ==> 7 4 6  ---> H E G
clique No. 2: ==> 6 2 4  ---> G C E
clique No. 3: ==> 5 3 4  ---> F D E
clique No. 4: ==> 1 0 3  ---> B A D
clique No. 5: ==> 3 0 4  ---> D A E
clique No. 6: ==> 2 0 4  ---> C A E
---------------------------
 *** ARISTAS INDUCIDAS ***
Arista inducida 1 ==> 3 0--->  D - A
Arista inducida 2 ==> 4 0--->  E - A
```

D.0.6. Estructura de árbol conjunto

A partir de los cliques obtenidos en el paso anterior, el programa forma los sepsets necesarios para formar el árbol conjunto mostrado en la figura 18. En el proceso es posible desplegar los sepsets identificados (solo se muestran los sepsets elegidos) y el árbol conjunto (join tree) producido, mismo que se representa como una matriz de adyacencias triangular.

```
----------------
*** SEPSETS ***
sepset[1]= 0 3 --> A D FROM cliques: (5,4)  masa=2 ..costo=64
sepset[2]= 0 4 --> A E FROM cliques: (6,5)  masa=2 ..costo=64
sepset[3]= 2 4 --> C E FROM cliques: (6,2)  masa=2 ..costo=96
sepset[4]= 3 4 --> D E FROM cliques: (5,3)  masa=2 ..costo=160
sepset[5]= 4 6 --> E G FROM cliques: (2,1)  masa=2 ..costo=192

jt[5,4]=1
jt[6,5]=2
jt[6,2]=3
jt[5,3]=4
jt[2,1]=5


------------------
*** JOIN TREE ***
   1  2  3  4  5  6
1  0
2  5  0
3  0  0  0
4  0  0  0  0
5  0  0  4  1  0
6  0  3  0  0  2  0
```

En la figura 18 se muestra la estructura del Árbol Conjunto generado.

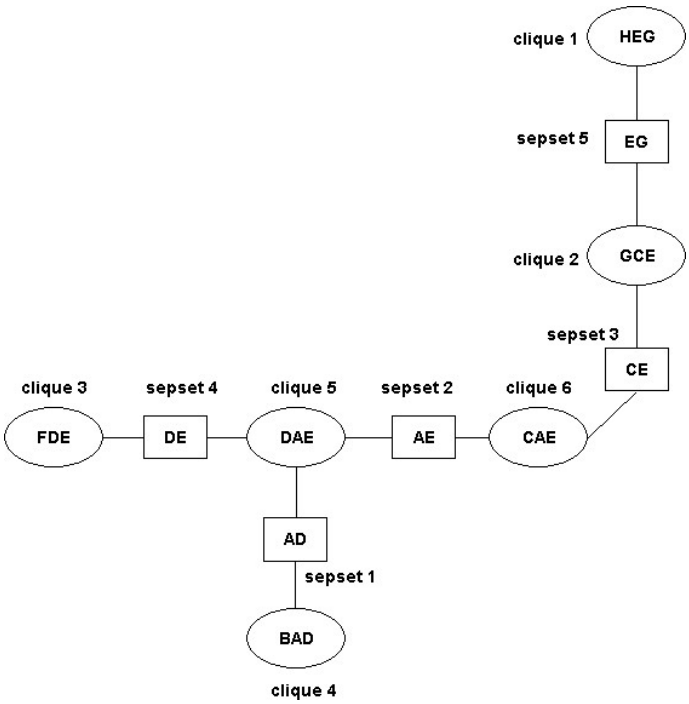


Figura 18: El árbol conjunto

D.0.7. Cálculo de la inferencia

Primero se inicializa cada clique a 1 (representado por una X en el programa) como se indica en el proceso. Posteriormente se multiplica cada variable de la red por algún clique que contenga su familia. En el anexo D.1 se muestra el proceso completo de iniciación de cliques y sepsets.

En los desplegados de los programas que implementan el razonamento Bayesiano se utiliza la siguiente nomenclatura:

- Los nodos se numeran de 0 a 7: nodo 0 = "A",... nodo 7 = "H".
- Los cliques y sepsets se numeran según se indica en la figura 18.
- X[4].probababilidad[1][0][1][0][0][0]=0.4980 indica:
    - Se trata del clique 4, que coniene a los nodos 1, 0 y 3 que son las variables B, A y D respectivamente.
    - Las columnas de izquierda a derecha se indexan con los posibles valores que puede tomar la variable representada en esa columna, es decir, los valores de B, A y D respectivamente.
    - El ejemplo mostrado, se refiere a que B tome el valor cuyo índice es 1 (off), A el valor cuyo índice es 0 (on), y D el valor cuyo índice es 1 (off).
    - El valor de probabilidad del clique 4, para B=off, A=on y D=off es 0.4980.

En el siguiente segmento del listado de ejecución se muestra el proceso de iniciación para el clique 4:

```
--> Se multiplico Nodo *0* con X[4] que consta de nodos *1*0*3*
*B*A*D*
======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.4980
X[4].probababilidad[0][0][1][0][0][0]=0.4980
X[4].probababilidad[0][1][0][0][0][0]=0.5020
X[4].probababilidad[0][1][1][0][0][0]=0.5020
X[4].probababilidad[1][0][0][0][0][0]=0.4980
X[4].probababilidad[1][0][1][0][0][0]=0.4980
X[4].probababilidad[1][1][0][0][0][0]=0.5020
X[4].probababilidad[1][1][1][0][0][0]=0.5020
Suma:                                =4.0000

--> Se multiplico Nodo *1* con X[4] que consta de nodos *1*0*3*
*B*A*D*
======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.2380
X[4].probababilidad[0][0][1][0][0][0]=0.2380
X[4].probababilidad[0][1][0][0][0][0]=0.2290
X[4].probababilidad[0][1][1][0][0][0]=0.2290
X[4].probababilidad[1][0][0][0][0][0]=0.2600
X[4].probababilidad[1][0][1][0][0][0]=0.2600
X[4].probababilidad[1][1][0][0][0][0]=0.2730
X[4].probababilidad[1][1][1][0][0][0]=0.2730
Suma:                                =2.0000

--> Se multiplico Nodo *3* con X[4] que consta de nodos *1*0*3*
*B*A*D*
======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.2120
X[4].probababilidad[0][0][1][0][0][0]=0.0260
X[4].probababilidad[0][1][0][0][0][0]=0.2040
X[4].probababilidad[0][1][1][0][0][0]=0.0250
X[4].probababilidad[1][0][0][0][0][0]=0.1312
X[4].probababilidad[1][0][1][0][0][0]=0.1288
X[4].probababilidad[1][1][0][0][0][0]=0.1378
X[4].probababilidad[1][1][1][0][0][0]=0.1352
Suma:                                =1.0000
```

Una vez iniciados los valores de los cliques y sepsets, se obtiene el árbol conjunto inconsistente.

Árbol conjunto inconsistente

En la figura 19 se muestra el árbol conjunto inconsistente que se produjo en el proceso de iniciación

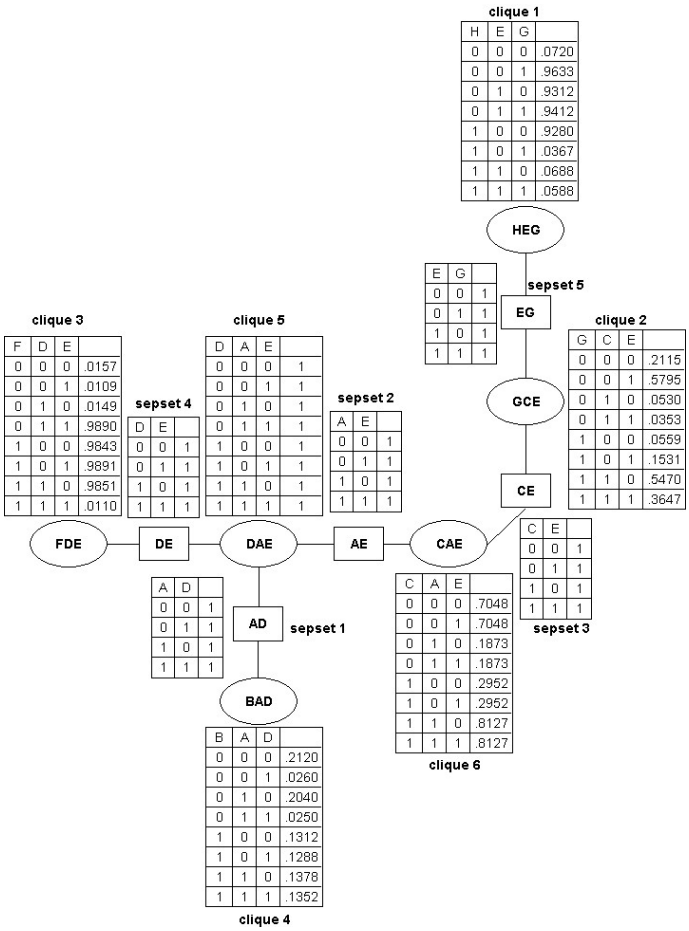


Figura 19: Árbol conjunto inconsistente

y multiplicación de nodos. Enseguida de cada clique y sepset se muestran los potenciales.

Entrada de Query

Además de las tablas de probabilidades y la representación abstracta de la red Bayesiana, la única entrada que requiere el algoritmo, es el 'Query', es decir, la pregunta por la probabilidad de una variable en particular como resultado del proceso de inferencia.

```
Query= P(D=on)
variable_query=*D*
valor_query   =*on*
```

Propagación Global

En el anexo D.2 se muestra el desplegado completo que se obtienen en el proceso de propagación global. Enseguida se muestra el segmento del proceso donde interviene el clique 4 directamente:

```
****** Paso de mensajes en Colecta_Evidencia(6) ******
pasar_mensaje[1]=4      recibir_mensaje[1]=5    conexion[1]=1
pasar_mensaje[2]=3      recibir_mensaje[2]=5    conexion[2]=4
pasar_mensaje[3]=5      recibir_mensaje[3]=6    conexion[3]=2
pasar_mensaje[4]=1      recibir_mensaje[4]=2    conexion[4]=5
pasar_mensaje[5]=2      recibir_mensaje[5]=6    conexion[5]=3

==============> pasar_mensaje[1]=4      recibir_mensaje[1]=5
***** Proyecta:        Cluster 4 --> Sepset 1

======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.2120
X[4].probababilidad[0][0][1][0][0][0]=0.0260
X[4].probababilidad[0][1][0][0][0][0]=0.2040
X[4].probababilidad[0][1][1][0][0][0]=0.0250
X[4].probababilidad[1][0][0][0][0][0]=0.1312
X[4].probababilidad[1][0][1][0][0][0]=0.1288
X[4].probababilidad[1][1][0][0][0][0]=0.1378
X[4].probababilidad[1][1][1][0][0][0]=0.1352
Suma:                                 =1.0000

**** Resultado de proyectar Cluster 4 en Sepset 1 ***
---------------OLD-------                        NEW
S_old[1].[0][0][0][0][0][0]=  1.00      S[1].[0][0][0][0][0][0]=  0.34
S_old[1].[0][1][0][0][0][0]=  1.00      S[1].[0][1][0][0][0][0]=  0.15
S_old[1].[1][0][0][0][0][0]=  1.00      S[1].[1][0][0][0][0][0]=  0.34
S_old[1].[1][1][0][0][0][0]=  1.00      S[1].[1][1][0][0][0][0]=  0.16


****** Paso de mensajes en Distribuye_Evidencia(6) ******
pasar_mensaje[1]=6      recibir_mensaje[1]=2    conexion[1]=3
pasar_mensaje[2]=2      recibir_mensaje[2]=1    conexion[2]=5
pasar_mensaje[3]=6      recibir_mensaje[3]=5    conexion[3]=2
pasar_mensaje[4]=5      recibir_mensaje[4]=3    conexion[4]=4
pasar_mensaje[5]=5      recibir_mensaje[5]=4    conexion[5]=1

=============> pasar_mensaje[5]=5       recibir_mensaje[5]=4
***** Proyecta:    Cluster 5 --> Sepset 1

**** Resultado de proyectar Cluster 5 en Sepset 1 ***
---------------OLD-------                        NEW
S_old[1].[0][0][0][0][0][0]=  0.34      S[1].[0][0][0][0][0][0]=  0.34
S_old[1].[0][1][0][0][0][0]=  0.15      S[1].[0][1][0][0][0][0]=  0.15
S_old[1].[1][0][0][0][0][0]=  0.34      S[1].[1][0][0][0][0][0]=  0.34
S_old[1].[1][1][0][0][0][0]=  0.16      S[1].[1][1][0][0][0][0]=  0.16
```

```
**** Resultado de Absorber Sepset 1 en Cluster 4 ***

======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.2120
X[4].probababilidad[0][0][1][0][0][0]=0.0260
X[4].probababilidad[0][1][0][0][0][0]=0.2040
X[4].probababilidad[0][1][1][0][0][0]=0.0250
X[4].probababilidad[1][0][0][0][0][0]=0.1312
X[4].probababilidad[1][0][1][0][0][0]=0.1288
X[4].probababilidad[1][1][0][0][0][0]=0.1378
X[4].probababilidad[1][1][1][0][0][0]=0.1352
Suma:                                 =1.0000
```

Árbol conjunto consistente

En la figura 20 se muestra el árbol conjunto consistente que se produjo en el proceso de propagación global. Enseguida de cada clique y sepset se muestra su tabla de probabilidad conjunta.

Marginalización y Normalización Para obtener el valor de la probabilidad solicitada en el query, el algoritmo elige un cluster (o sepset) que contenga la variable de interés, y simplemente la marginaliza y normaliza para obtener el resultado.

En el caso ejemplo, se elige el clique 4 para marginalizar el valor de D=on, y así obtener la probabilidad $P(D = on)$. El desplegado del programa es el siguiente:

```
P(BAD)
======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.2120
X[4].probababilidad[0][0][1][0][0][0]=0.0260
X[4].probababilidad[0][1][0][0][0][0]=0.2040
X[4].probababilidad[0][1][1][0][0][0]=0.0250
X[4].probababilidad[1][0][0][0][0][0]=0.1312
X[4].probababilidad[1][0][1][0][0][0]=0.1288
X[4].probababilidad[1][1][0][0][0][0]=0.1378
X[4].probababilidad[1][1][1][0][0][0]=0.1352
Suma:                                 =1.0000

P(D)
======== V ==========
V.probababilidad[0][0][0][0][0][0]=0.6850
V.probababilidad[1][0][0][0][0][0]=0.3150

=====>>> P(D=on) = 0.6850

==== Valor Normalizado ====
V.probababilidad[0][0][0][0][0][0]=0.6850
V.probababilidad[1][0][0][0][0][0]=0.3150
```

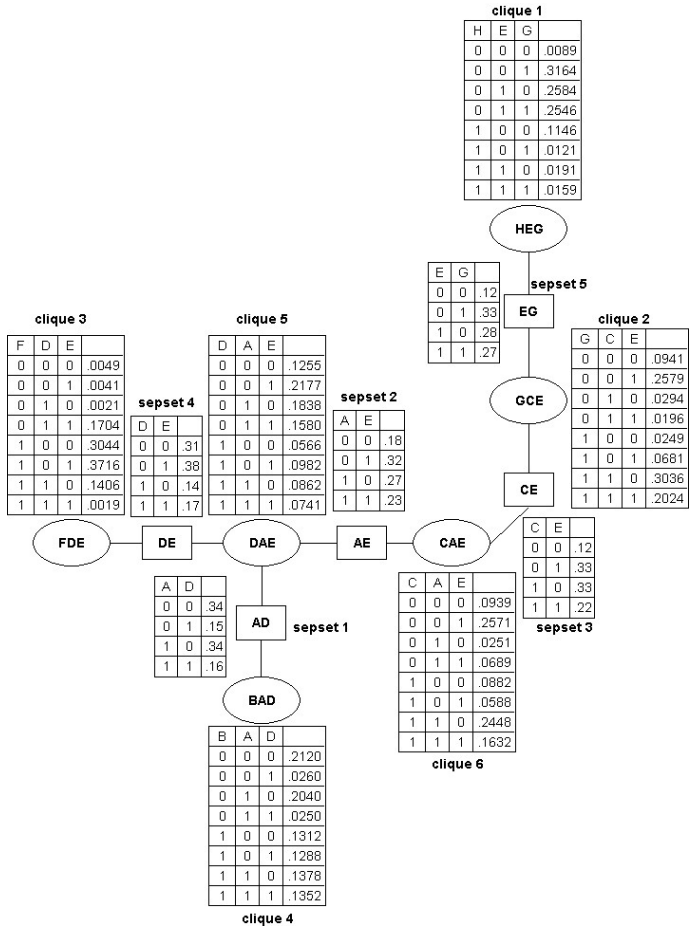


Figura 20: Árbol conjunto consistente

```
=====>>> P(D=on) = 0.6850
```

Inferencia con observaciones Las observaciones se ingresan al momento de ingresar el Query:

```
Query= P(D=on|B=off)

varobs[0] B
valobs[0] off
tot_obs=1

variable_query=*D*
valor_query   =*on*

variable_observada[0]=*B*
valor_observado[0]   =*off*
```

En la presencia de observaciones, el algoritmo multiplica el arreglo Lambda de observaciones en los nodos pertinentes, en este caso, codifica la observación $B = off$ en el nodo 1 que corresponde a la variable B. Se multiplica el nodo 1 modificado por el clique 4, resultando en la siguiente tabla de probabilidades:

```
----- inicia APLICA OBSERVACIONES -----------------

--> Se multiplico Lambda Nodo *1* con X[4]
que consta de nodos *1*0*3* *B*A*D*
======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.0000
X[4].probababilidad[0][0][1][0][0][0]=0.0000
X[4].probababilidad[0][1][0][0][0][0]=0.0000
X[4].probababilidad[0][1][1][0][0][0]=0.0000
X[4].probababilidad[1][0][0][0][0][0]=0.1312
X[4].probababilidad[1][0][1][0][0][0]=0.1288
X[4].probababilidad[1][1][0][0][0][0]=0.1378
X[4].probababilidad[1][1][1][0][0][0]=0.1352
Suma:                                 =0.5330

----- finaliza APLICA OBSERVACIONES ---
```

Una vez efectuada la aplicación de observaciones, el proceso de propagación global y formación del árbol conjunto consistente se hace igual que cuando no hay observaciones, el resultado obtenido para el clique del ejemplo es siguiente:

```
P(BAD)
======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.0000
X[4].probababilidad[0][0][1][0][0][0]=0.0000
X[4].probababilidad[0][1][0][0][0][0]=0.0000
X[4].probababilidad[0][1][1][0][0][0]=0.0000
X[4].probababilidad[1][0][0][0][0][0]=0.1312
X[4].probababilidad[1][0][1][0][0][0]=0.1288
X[4].probababilidad[1][1][0][0][0][0]=0.1378
X[4].probababilidad[1][1][1][0][0][0]=0.1352
Suma:                                 =0.5330

P(D)
======== V ==========
V.probababilidad[0][0][0][0][0][0]=0.2690
V.probababilidad[1][0][0][0][0][0]=0.2640

=====>>> P(D=on|B=off) = 0.2690
```

Para obtener la probabilidad exacta, dada las observaciones, se marginaliza para obtener:

```
==== Valor Normalizado ====
=====>>> P(D=on|B=off) = 0.5047
```

D.1. Iniciación de cliques y sepsets

Como primer paso, el valor de cada combinación de variables en las tablas de cliques y sepsets, se inicializa a 1. Enseguida se efectúa la multiplicación de las probabilidades de cada nodo por algún clique o sepset que contenga a su familia. El desplegado del proceso completo de multiplicación es el siguiente:

```
--> Se multiplico Nodo *0* con X[4] que consta de nodos *1*0*3*
*B*A*D*
======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.4980
X[4].probababilidad[0][0][1][0][0][0]=0.4980
X[4].probababilidad[0][1][0][0][0][0]=0.5020
X[4].probababilidad[0][1][1][0][0][0]=0.5020
X[4].probababilidad[1][0][0][0][0][0]=0.4980
X[4].probababilidad[1][0][1][0][0][0]=0.4980
X[4].probababilidad[1][1][0][0][0][0]=0.5020
X[4].probababilidad[1][1][1][0][0][0]=0.5020
```

```
Suma:                                =4.0000

--> Se multiplico Nodo *1* con X[4] que consta de nodos *1*0*3*
*B*A*D*
======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.2380
X[4].probababilidad[0][0][1][0][0][0]=0.2380
X[4].probababilidad[0][1][0][0][0][0]=0.2290
X[4].probababilidad[0][1][1][0][0][0]=0.2290
X[4].probababilidad[1][0][0][0][0][0]=0.2600
X[4].probababilidad[1][0][1][0][0][0]=0.2600
X[4].probababilidad[1][1][0][0][0][0]=0.2730
X[4].probababilidad[1][1][1][0][0][0]=0.2730
Suma:                                =2.0000



--> Se multiplico Nodo *2* con X[6] que consta de nodos *2*0*4*
*C*A*E*
======== X[6] ==========
X[6].probababilidad[0][0][0][0][0][0]=0.7048
X[6].probababilidad[0][0][1][0][0][0]=0.7048
X[6].probababilidad[0][1][0][0][0][0]=0.1873
X[6].probababilidad[0][1][1][0][0][0]=0.1873
X[6].probababilidad[1][0][0][0][0][0]=0.2952
X[6].probababilidad[1][0][1][0][0][0]=0.2952
X[6].probababilidad[1][1][0][0][0][0]=0.8127
X[6].probababilidad[1][1][1][0][0][0]=0.8127
Suma:                                =4.0000

--> Se multiplico Nodo *3* con X[4] que consta de nodos *1*0*3*
*B*A*D*
======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.2120
X[4].probababilidad[0][0][1][0][0][0]=0.0260
X[4].probababilidad[0][1][0][0][0][0]=0.2040
X[4].probababilidad[0][1][1][0][0][0]=0.0250
X[4].probababilidad[1][0][0][0][0][0]=0.1312
X[4].probababilidad[1][0][1][0][0][0]=0.1288
X[4].probababilidad[1][1][0][0][0][0]=0.1378
X[4].probababilidad[1][1][1][0][0][0]=0.1352
Suma:                                =1.0000

--> Se multiplico Nodo *4* con X[2] que consta de nodos *6*2*4*
*G*C*E*
======== X[2] ==========
X[2].probababilidad[0][0][0][0][0][0]=0.2674
X[2].probababilidad[0][0][1][0][0][0]=0.7326
```

```
X[2].probababilidad[0][1][0][0][0][0]=0.6000
X[2].probababilidad[0][1][1][0][0][0]=0.4000
X[2].probababilidad[1][0][0][0][0][0]=0.2674
X[2].probababilidad[1][0][1][0][0][0]=0.7326
X[2].probababilidad[1][1][0][0][0][0]=0.6000
X[2].probababilidad[1][1][1][0][0][0]=0.4000
Suma:                                =4.0000


--> Se multiplico Nodo *5* con X[3] que consta de nodos *5*3*4*
*F*D*E*
======== X[3] ==========
X[3].probababilidad[0][0][0][0][0][0]=0.0157
X[3].probababilidad[0][0][1][0][0][0]=0.0109
X[3].probababilidad[0][1][0][0][0][0]=0.0149
X[3].probababilidad[0][1][1][0][0][0]=0.9890
X[3].probababilidad[1][0][0][0][0][0]=0.9843
X[3].probababilidad[1][0][1][0][0][0]=0.9891
X[3].probababilidad[1][1][0][0][0][0]=0.9851
X[3].probababilidad[1][1][1][0][0][0]=0.0110
Suma:                                =4.0000

--> Se multiplico Nodo *6* con X[2] que consta de nodos *6*2*4*
*G*C*E*
======== X[2] ==========
X[2].probababilidad[0][0][0][0][0][0]=0.2115
X[2].probababilidad[0][0][1][0][0][0]=0.5795
X[2].probababilidad[0][1][0][0][0][0]=0.0530
X[2].probababilidad[0][1][1][0][0][0]=0.0353
X[2].probababilidad[1][0][0][0][0][0]=0.0559
X[2].probababilidad[1][0][1][0][0][0]=0.1531
X[2].probababilidad[1][1][0][0][0][0]=0.5470
X[2].probababilidad[1][1][1][0][0][0]=0.3647
Suma:                                =2.0000

--> Se multiplico Nodo *7* con X[1] que consta de nodos *7*4*6*
*H*E*G*
======== X[1] ==========
X[1].probababilidad[0][0][0][0][0][0]=0.0720
X[1].probababilidad[0][0][1][0][0][0]=0.9633
X[1].probababilidad[0][1][0][0][0][0]=0.9312
X[1].probababilidad[0][1][1][0][0][0]=0.9412
X[1].probababilidad[1][0][0][0][0][0]=0.9280
X[1].probababilidad[1][0][1][0][0][0]=0.0367
X[1].probababilidad[1][1][0][0][0][0]=0.0688
X[1].probababilidad[1][1][1][0][0][0]=0.0588
Suma:                                =4.0000
```

D.2. Propagación global

D.2.1. Colecta evidencia

Paso de mensajes en Colecta Evidencia.

```
clique 2 conectado por sepset 5 a clique 1
clique 6 conectado por sepset 3 a clique 2
clique 5 conectado por sepset 4 a clique 3
clique 5 conectado por sepset 1 a clique 4
clique 6 conectado por sepset 2 a clique 5

****** Paso de mensajes en Colecta_Evidencia(6) ******
pasar_mensaje[1]=4      recibir_mensaje[1]=5    conexion[1]=1
pasar_mensaje[2]=3      recibir_mensaje[2]=5    conexion[2]=4
pasar_mensaje[3]=5      recibir_mensaje[3]=6    conexion[3]=2
pasar_mensaje[4]=1      recibir_mensaje[4]=2    conexion[4]=5
pasar_mensaje[5]=2      recibir_mensaje[5]=6    conexion[5]=3

==============> pasar_mensaje[1]=4      recibir_mensaje[1]=5

***** Proyecta: Cluster 4 --> Sepset 1

======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.2120
X[4].probababilidad[0][0][1][0][0][0]=0.0260
X[4].probababilidad[0][1][0][0][0][0]=0.2040
X[4].probababilidad[0][1][1][0][0][0]=0.0250
X[4].probababilidad[1][0][0][0][0][0]=0.1312
X[4].probababilidad[1][0][1][0][0][0]=0.1288
X[4].probababilidad[1][1][0][0][0][0]=0.1378
X[4].probababilidad[1][1][1][0][0][0]=0.1352
Suma:                                =1.0000

**** Resultado de proyectar Cluster 4 en Sepset 1 ***
---------------OLD-------                        NEW
S_old[1].[0][0][0][0][0][0]=  1.00      S[1].[0][0][0][0][0][0]=  0.34
S_old[1].[0][1][0][0][0][0]=  1.00      S[1].[0][1][0][0][0][0]=  0.15
S_old[1].[1][0][0][0][0][0]=  1.00      S[1].[1][0][0][0][0][0]=  0.34
S_old[1].[1][1][0][0][0][0]=  1.00      S[1].[1][1][0][0][0][0]=  0.16


***** Absorbe: Cluster 5 --> Sepset 1
**** Resultado de Absorber Sepset 1 en Cluster 5 ***

======== X[5] ==========
```

```
X[5].probababilidad[0][0][0][0][0][0]=0.3432
X[5].probababilidad[0][0][1][0][0][0]=0.3432
X[5].probababilidad[0][1][0][0][0][0]=0.3418
X[5].probababilidad[0][1][1][0][0][0]=0.3418
X[5].probababilidad[1][0][0][0][0][0]=0.1548
X[5].probababilidad[1][0][1][0][0][0]=0.1548
X[5].probababilidad[1][1][0][0][0][0]=0.1602
X[5].probababilidad[1][1][1][0][0][0]=0.1602
Suma:                                =2.0000

==============> pasar_mensaje[2]=3      recibir_mensaje[2]=5

***** Proyecta: Cluster 3 --> Sepset 4

======== X[3] ==========
X[3].probababilidad[0][0][0][0][0][0]=0.0157
X[3].probababilidad[0][0][1][0][0][0]=0.0109
X[3].probababilidad[0][1][0][0][0][0]=0.0149
X[3].probababilidad[0][1][1][0][0][0]=0.9890
X[3].probababilidad[1][0][0][0][0][0]=0.9843
X[3].probababilidad[1][0][1][0][0][0]=0.9891
X[3].probababilidad[1][1][0][0][0][0]=0.9851
X[3].probababilidad[1][1][1][0][0][0]=0.0110
Suma:                                =4.0000

**** Resultado de proyectar Cluster 3 en Sepset 4 ***
---------------OLD-------                        NEW
S_old[4].[0][0][0][0][0][0]=  1.00      S[4].[0][0][0][0][0][0]=  1.00
S_old[4].[0][1][0][0][0][0]=  1.00      S[4].[0][1][0][0][0][0]=  1.00
S_old[4].[1][0][0][0][0][0]=  1.00      S[4].[1][0][0][0][0][0]=  1.00
S_old[4].[1][1][0][0][0][0]=  1.00      S[4].[1][1][0][0][0][0]=  1.00


***** Absorbe: Cluster 5 --> Sepset 4
**** Resultado de Absorber Sepset 4 en Cluster 5 ***

======== X[5] ==========
X[5].probababilidad[0][0][0][0][0][0]=0.3432
X[5].probababilidad[0][0][1][0][0][0]=0.3432
X[5].probababilidad[0][1][0][0][0][0]=0.3418
X[5].probababilidad[0][1][1][0][0][0]=0.3418
X[5].probababilidad[1][0][0][0][0][0]=0.1548
X[5].probababilidad[1][0][1][0][0][0]=0.1548
X[5].probababilidad[1][1][0][0][0][0]=0.1602
X[5].probababilidad[1][1][1][0][0][0]=0.1602
Suma:                                =2.0000
```

```
==============> pasar_mensaje[3]=5      recibir_mensaje[3]=6
***** Proyecta: Cluster 5 --> Sepset 2
======== X[5] ==========
X[5].probababilidad[0][0][0][0][0][0]=0.3432
X[5].probababilidad[0][0][1][0][0][0]=0.3432
X[5].probababilidad[0][1][0][0][0][0]=0.3418
X[5].probababilidad[0][1][1][0][0][0]=0.3418
X[5].probababilidad[1][0][0][0][0][0]=0.1548
X[5].probababilidad[1][0][1][0][0][0]=0.1548
X[5].probababilidad[1][1][0][0][0][0]=0.1602
X[5].probababilidad[1][1][1][0][0][0]=0.1602
Suma:                                 =2.0000

**** Resultado de proyectar Cluster 5 en Sepset 2 ***
---------------OLD-------                        NEW
S_old[2].[0][0][0][0][0][0]=  1.00      S[2].[0][0][0][0][0][0]=  0.50
S_old[2].[0][1][0][0][0][0]=  1.00      S[2].[0][1][0][0][0][0]=  0.50
S_old[2].[1][0][0][0][0][0]=  1.00      S[2].[1][0][0][0][0][0]=  0.50
S_old[2].[1][1][0][0][0][0]=  1.00      S[2].[1][1][0][0][0][0]=  0.50



***** Absorbe: Cluster 6 --> Sepset 2
**** Resultado de Absorber Sepset 2 en Cluster 6 ***

======== X[6] ==========
X[6].probababilidad[0][0][0][0][0][0]=0.3510
X[6].probababilidad[0][0][1][0][0][0]=0.3510
X[6].probababilidad[0][1][0][0][0][0]=0.0940
X[6].probababilidad[0][1][1][0][0][0]=0.0940
X[6].probababilidad[1][0][0][0][0][0]=0.1470
X[6].probababilidad[1][0][1][0][0][0]=0.1470
X[6].probababilidad[1][1][0][0][0][0]=0.4080
X[6].probababilidad[1][1][1][0][0][0]=0.4080
Suma:                                 =2.0000

==============> pasar_mensaje[4]=1      recibir_mensaje[4]=2
***** Proyecta: Cluster 1 --> Sepset 5
======== X[1] ==========
X[1].probababilidad[0][0][0][0][0][0]=0.0720
X[1].probababilidad[0][0][1][0][0][0]=0.9633
X[1].probababilidad[0][1][0][0][0][0]=0.9312
X[1].probababilidad[0][1][1][0][0][0]=0.9412
X[1].probababilidad[1][0][0][0][0][0]=0.9280
X[1].probababilidad[1][0][1][0][0][0]=0.0367
X[1].probababilidad[1][1][0][0][0][0]=0.0688
X[1].probababilidad[1][1][1][0][0][0]=0.0588 Suma: =4.0000
```

```
**** Resultado de proyectar Cluster 1 en Sepset 5 ***
---------------OLD-------                        NEW
S_old[5].[0][0][0][0][0][0]=  1.00      S[5].[0][0][0][0][0][0]= 1.00
S_old[5].[0][1][0][0][0][0]=  1.00      S[5].[0][1][0][0][0][0]= 1.00
S_old[5].[1][0][0][0][0][0]=  1.00      S[5].[1][0][0][0][0][0]= 1.00
S_old[5].[1][1][0][0][0][0]=  1.00      S[5].[1][1][0][0][0][0]= 1.00


***** Absorbe: Cluster 2 --> Sepset 5
**** Resultado de Absorber Sepset 5 en Cluster 2 ***

======== X[2] ==========
X[2].probababilidad[0][0][0][0][0][0]=0.2115
X[2].probababilidad[0][0][1][0][0][0]=0.5795
X[2].probababilidad[0][1][0][0][0][0]=0.0530
X[2].probababilidad[0][1][1][0][0][0]=0.0353
X[2].probababilidad[1][0][0][0][0][0]=0.0559
X[2].probababilidad[1][0][1][0][0][0]=0.1531
X[2].probababilidad[1][1][0][0][0][0]=0.5470
X[2].probababilidad[1][1][1][0][0][0]=0.3647
Suma:                                 =2.0000

==============> pasar_mensaje[5]=2      recibir_mensaje[5]=6
***** Proyecta: Cluster 2 --> Sepset 3
======== X[2] ==========
X[2].probababilidad[0][0][0][0][0][0]=0.2115
X[2].probababilidad[0][0][1][0][0][0]=0.5795
X[2].probababilidad[0][1][0][0][0][0]=0.0530
X[2].probababilidad[0][1][1][0][0][0]=0.0353
X[2].probababilidad[1][0][0][0][0][0]=0.0559
X[2].probababilidad[1][0][1][0][0][0]=0.1531
X[2].probababilidad[1][1][0][0][0][0]=0.5470
X[2].probababilidad[1][1][1][0][0][0]=0.3647
Suma:                                 =2.0000

**** Resultado de proyectar Cluster 2 en Sepset 3 ***
---------------OLD-------                        NEW
S_old[3].[0][0][0][0][0][0]=  1.00      S[3].[0][0][0][0][0][0]=  0.27
S_old[3].[0][1][0][0][0][0]=  1.00      S[3].[0][1][0][0][0][0]=  0.73
S_old[3].[1][0][0][0][0][0]=  1.00      S[3].[1][0][0][0][0][0]=  0.60
S_old[3].[1][1][0][0][0][0]=  1.00      S[3].[1][1][0][0][0][0]=  0.40


***** Absorbe: Cluster 6 --> Sepset 3

======== X[6] ==========
X[6].probababilidad[0][0][0][0][0][0]=0.3510
X[6].probababilidad[0][0][1][0][0][0]=0.3510
```

```
X[6].probababilidad[0][1][0][0][0][0]=0.0940
X[6].probababilidad[0][1][1][0][0][0]=0.0940
X[6].probababilidad[1][0][0][0][0][0]=0.1470
X[6].probababilidad[1][0][1][0][0][0]=0.1470
X[6].probababilidad[1][1][0][0][0][0]=0.4080
X[6].probababilidad[1][1][1][0][0][0]=0.4080
Suma:                                =2.0000

**** Resultado de Absorber Sepset 3 en Cluster 6 ***

======== X[6] ==========
X[6].probababilidad[0][0][0][0][0][0]=0.0939
X[6].probababilidad[0][0][1][0][0][0]=0.2571
X[6].probababilidad[0][1][0][0][0][0]=0.0251
X[6].probababilidad[0][1][1][0][0][0]=0.0689
X[6].probababilidad[1][0][0][0][0][0]=0.0882
X[6].probababilidad[1][0][1][0][0][0]=0.0588
X[6].probababilidad[1][1][0][0][0][0]=0.2448
X[6].probababilidad[1][1][1][0][0][0]=0.1632
Suma:                                =1.0000

------------------
*** JOIN TREE despues de Colecta_Evidencia***
   1  2  3  4  5  6
1  0
2 -5  0
3  0  0  0
4  0  0  0  0
5  0  0 -4 -1  0
6  0 -3  0  0 -2  0
```

D.2.2. Distribuye evidencia

Paso de mensajes en Distribuye Evidencia.

```
****** Paso de mensajes en Distribuye_Evidencia(6) ******
pasar_mensaje[1]=6      recibir_mensaje[1]=2    conexion[1]=3
pasar_mensaje[2]=2      recibir_mensaje[2]=1    conexion[2]=5
pasar_mensaje[3]=6      recibir_mensaje[3]=5    conexion[3]=2
pasar_mensaje[4]=5      recibir_mensaje[4]=3    conexion[4]=4
pasar_mensaje[5]=5      recibir_mensaje[5]=4    conexion[5]=1

=============> pasar_mensaje[1]=6       recibir_mensaje[1]=2

***** Proyecta: Cluster 6 --> Sepset 3
======== X[6] ==========
```

```
X[6].probababilidad[0][0][0][0][0][0]=0.0939
X[6].probababilidad[0][0][1][0][0][0]=0.2571
X[6].probababilidad[0][1][0][0][0][0]=0.0251
X[6].probababilidad[0][1][1][0][0][0]=0.0689
X[6].probababilidad[1][0][0][0][0][0]=0.0882
X[6].probababilidad[1][0][1][0][0][0]=0.0588
X[6].probababilidad[1][1][0][0][0][0]=0.2448
X[6].probababilidad[1][1][1][0][0][0]=0.1632
Suma:                                =1.0000

**** Resultado de proyectar Cluster 6 en Sepset 3 ***
---------------OLD-------                        NEW
S_old[3].[0][0][0][0][0][0]=  0.27      S[3].[0][0][0][0][0][0]=  0.12
S_old[3].[0][1][0][0][0][0]=  0.73      S[3].[0][1][0][0][0][0]=  0.33
S_old[3].[1][0][0][0][0][0]=  0.60      S[3].[1][0][0][0][0][0]=  0.33
S_old[3].[1][1][0][0][0][0]=  0.40      S[3].[1][1][0][0][0][0]=  0.22

***** Absorbe: Cluster 2 --> Sepset 3
======== X[2] ==========
X[2].probababilidad[0][0][0][0][0][0]=0.2115
X[2].probababilidad[0][0][1][0][0][0]=0.5795
X[2].probababilidad[0][1][0][0][0][0]=0.0530
X[2].probababilidad[0][1][1][0][0][0]=0.0353
X[2].probababilidad[1][0][0][0][0][0]=0.0559
X[2].probababilidad[1][0][1][0][0][0]=0.1531
X[2].probababilidad[1][1][0][0][0][0]=0.5470
X[2].probababilidad[1][1][1][0][0][0]=0.3647
Suma:                                =2.0000

**** Resultado de Absorber Sepset 3 en Cluster 2 ***
======== X[2] ==========
X[2].probababilidad[0][0][0][0][0][0]=0.0941
X[2].probababilidad[0][0][1][0][0][0]=0.2579
X[2].probababilidad[0][1][0][0][0][0]=0.0294
X[2].probababilidad[0][1][1][0][0][0]=0.0196
X[2].probababilidad[1][0][0][0][0][0]=0.0249
X[2].probababilidad[1][0][1][0][0][0]=0.0681
X[2].probababilidad[1][1][0][0][0][0]=0.3036
X[2].probababilidad[1][1][1][0][0][0]=0.2024
Suma:                                =1.0000

=============> pasar_mensaje[2]=2       recibir_mensaje[2]=1

***** Proyecta: Cluster 2 --> Sepset 5
======== X[2] ==========
X[2].probababilidad[0][0][0][0][0][0]=0.0941
X[2].probababilidad[0][0][1][0][0][0]=0.2579
```

```
 X[2].probababilidad[0][1][0][0][0][0]=0.0294
 X[2].probababilidad[0][1][1][0][0][0]=0.0196
 X[2].probababilidad[1][0][0][0][0][0]=0.0249
 X[2].probababilidad[1][0][1][0][0][0]=0.0681
 X[2].probababilidad[1][1][0][0][0][0]=0.3036
 X[2].probababilidad[1][1][1][0][0][0]=0.2024
 Suma:                                =1.0000

 **** Resultado de proyectar Cluster 2 en Sepset 5 ***
 ---------------OLD-------                        NEW
 S_old[5].[0][0][0][0][0][0]=  1.00      S[5].[0][0][0][0][0][0]=  0.12
 S_old[5].[0][1][0][0][0][0]=  1.00      S[5].[0][1][0][0][0][0]=  0.33
 S_old[5].[1][0][0][0][0][0]=  1.00      S[5].[1][0][0][0][0][0]=  0.28
 S_old[5].[1][1][0][0][0][0]=  1.00      S[5].[1][1][0][0][0][0]=  0.27

 ***** Absorbe: Cluster 1 --> Sepset 5
 ======== X[1] ==========
 X[1].probababilidad[0][0][0][0][0][0]=0.0720
 X[1].probababilidad[0][0][1][0][0][0]=0.9633
 X[1].probababilidad[0][1][0][0][0][0]=0.9312
 X[1].probababilidad[0][1][1][0][0][0]=0.9412
 X[1].probababilidad[1][0][0][0][0][0]=0.9280
 X[1].probababilidad[1][0][1][0][0][0]=0.0367
 X[1].probababilidad[1][1][0][0][0][0]=0.0688
 X[1].probababilidad[1][1][1][0][0][0]=0.0588
 Suma:                                =4.0000

**** Resultado de Absorber Sepset 5 en Cluster 1 ***

 ======== X[1] ==========
 X[1].probababilidad[0][0][0][0][0][0]=0.0089
 X[1].probababilidad[0][0][1][0][0][0]=0.3164
 X[1].probababilidad[0][1][0][0][0][0]=0.2584
 X[1].probababilidad[0][1][1][0][0][0]=0.2546
 X[1].probababilidad[1][0][0][0][0][0]=0.1146
 X[1].probababilidad[1][0][1][0][0][0]=0.0121
 X[1].probababilidad[1][1][0][0][0][0]=0.0191
 X[1].probababilidad[1][1][1][0][0][0]=0.0159
 Suma:                                =1.0000

 =============> pasar_mensaje[3]=6       recibir_mensaje[3]=5

 ***** Proyecta: Cluster 6 --> Sepset 2
 ======== X[6] ==========
 X[6].probababilidad[0][0][0][0][0][0]=0.0939
 X[6].probababilidad[0][0][1][0][0][0]=0.2571
 X[6].probababilidad[0][1][0][0][0][0]=0.0251
```

```
X[6].probababilidad[0][1][1][0][0][0]=0.0689
X[6].probababilidad[1][0][0][0][0][0]=0.0882
X[6].probababilidad[1][0][1][0][0][0]=0.0588
X[6].probababilidad[1][1][0][0][0][0]=0.2448
X[6].probababilidad[1][1][1][0][0][0]=0.1632
Suma:                                 =1.0000

**** Resultado de proyectar Cluster 6 en Sepset 2 ***
---------------OLD-------                        NEW
S_old[2].[0][0][0][0][0][0]=  0.50      S[2].[0][0][0][0][0][0]=  0.18
S_old[2].[0][1][0][0][0][0]=  0.50      S[2].[0][1][0][0][0][0]=  0.32
S_old[2].[1][0][0][0][0][0]=  0.50      S[2].[1][0][0][0][0][0]=  0.27
S_old[2].[1][1][0][0][0][0]=  0.50      S[2].[1][1][0][0][0][0]=  0.23


***** Absorbe: Cluster 5 --> Sepset 2
======== X[5] ==========
X[5].probababilidad[0][0][0][0][0][0]=0.3432
X[5].probababilidad[0][0][1][0][0][0]=0.3432
X[5].probababilidad[0][1][0][0][0][0]=0.3418
X[5].probababilidad[0][1][1][0][0][0]=0.3418
X[5].probababilidad[1][0][0][0][0][0]=0.1548
X[5].probababilidad[1][0][1][0][0][0]=0.1548
X[5].probababilidad[1][1][0][0][0][0]=0.1602
X[5].probababilidad[1][1][1][0][0][0]=0.1602
Suma:                                 =2.0000

**** Resultado de Absorber Sepset 2 en Cluster 5 ***
======== X[5] ==========
X[5].probababilidad[0][0][0][0][0][0]=0.1255
X[5].probababilidad[0][0][1][0][0][0]=0.2177
X[5].probababilidad[0][1][0][0][0][0]=0.1838
X[5].probababilidad[0][1][1][0][0][0]=0.1580
X[5].probababilidad[1][0][0][0][0][0]=0.0566
X[5].probababilidad[1][0][1][0][0][0]=0.0982
X[5].probababilidad[1][1][0][0][0][0]=0.0862
X[5].probababilidad[1][1][1][0][0][0]=0.0741
Suma:                                 =1.0000

==============> pasar_mensaje[4]=5       recibir_mensaje[4]=3

***** Proyecta: Cluster 5 --> Sepset 4
======== X[5] ==========
X[5].probababilidad[0][0][0][0][0][0]=0.1255
X[5].probababilidad[0][0][1][0][0][0]=0.2177
X[5].probababilidad[0][1][0][0][0][0]=0.1838
X[5].probababilidad[0][1][1][0][0][0]=0.1580
X[5].probababilidad[1][0][0][0][0][0]=0.0566
```

```
X[5].probababilidad[1][0][1][0][0][0]=0.0982
X[5].probababilidad[1][1][0][0][0][0]=0.0862
X[5].probababilidad[1][1][1][0][0][0]=0.0741
Suma:                                 =1.0000

**** Resultado de proyectar Cluster 5 en Sepset 4 ***
---------------OLD-------                        NEW
S_old[4].[0][0][0][0][0][0]=  1.00      S[4].[0][0][0][0][0][0]=  0.31
S_old[4].[0][1][0][0][0][0]=  1.00      S[4].[0][1][0][0][0][0]=  0.38
S_old[4].[1][0][0][0][0][0]=  1.00      S[4].[1][0][0][0][0][0]=  0.14
S_old[4].[1][1][0][0][0][0]=  1.00      S[4].[1][1][0][0][0][0]=  0.17

***** Absorbe: Cluster 3 --> Sepset 4
======== X[3] ==========
X[3].probababilidad[0][0][0][0][0][0]=0.0157
X[3].probababilidad[0][0][1][0][0][0]=0.0109
X[3].probababilidad[0][1][0][0][0][0]=0.0149
X[3].probababilidad[0][1][1][0][0][0]=0.9890
X[3].probababilidad[1][0][0][0][0][0]=0.9843
X[3].probababilidad[1][0][1][0][0][0]=0.9891
X[3].probababilidad[1][1][0][0][0][0]=0.9851
X[3].probababilidad[1][1][1][0][0][0]=0.0110
Suma:                                 =4.0000

**** Resultado de Absorber Sepset 4 en Cluster 3 ***
======== X[3] ==========
X[3].probababilidad[0][0][0][0][0][0]=0.0049
X[3].probababilidad[0][0][1][0][0][0]=0.0041
X[3].probababilidad[0][1][0][0][0][0]=0.0021
X[3].probababilidad[0][1][1][0][0][0]=0.1704
X[3].probababilidad[1][0][0][0][0][0]=0.3044
X[3].probababilidad[1][0][1][0][0][0]=0.3716
X[3].probababilidad[1][1][0][0][0][0]=0.1406
X[3].probababilidad[1][1][1][0][0][0]=0.0019
Suma:                                 =1.0000

=============> pasar_mensaje[5]=5       recibir_mensaje[5]=4
***** Proyecta: Cluster 5 --> Sepset 1
======== X[5] ==========
X[5].probababilidad[0][0][0][0][0][0]=0.1255
X[5].probababilidad[0][0][1][0][0][0]=0.2177
X[5].probababilidad[0][1][0][0][0][0]=0.1838
X[5].probababilidad[0][1][1][0][0][0]=0.1580
X[5].probababilidad[1][0][0][0][0][0]=0.0566
X[5].probababilidad[1][0][1][0][0][0]=0.0982
X[5].probababilidad[1][1][0][0][0][0]=0.0862
X[5].probababilidad[1][1][1][0][0][0]=0.0741
```

```
Suma:                                =1.0000

**** Resultado de proyectar Cluster 5 en Sepset 1 ***
---------------OLD-------                        NEW
S_old[1].[0][0][0][0][0][0]=  0.34      S[1].[0][0][0][0][0][0]=  0.34
S_old[1].[0][1][0][0][0][0]=  0.15      S[1].[0][1][0][0][0][0]=  0.15
S_old[1].[1][0][0][0][0][0]=  0.34      S[1].[1][0][0][0][0][0]=  0.34
S_old[1].[1][1][0][0][0][0]=  0.16      S[1].[1][1][0][0][0][0]=  0.16

***** Absorbe: Cluster 4 --> Sepset 1
======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.2120
X[4].probababilidad[0][0][1][0][0][0]=0.0260
X[4].probababilidad[0][1][0][0][0][0]=0.2040
X[4].probababilidad[0][1][1][0][0][0]=0.0250
X[4].probababilidad[1][0][0][0][0][0]=0.1312
X[4].probababilidad[1][0][1][0][0][0]=0.1288
X[4].probababilidad[1][1][0][0][0][0]=0.1378
X[4].probababilidad[1][1][1][0][0][0]=0.1352
Suma:                                =1.0000

**** Resultado de Absorber Sepset 1 en Cluster 4 ***
======== X[4] ==========
X[4].probababilidad[0][0][0][0][0][0]=0.2120
X[4].probababilidad[0][0][1][0][0][0]=0.0260
X[4].probababilidad[0][1][0][0][0][0]=0.2040
X[4].probababilidad[0][1][1][0][0][0]=0.0250
X[4].probababilidad[1][0][0][0][0][0]=0.1312
X[4].probababilidad[1][0][1][0][0][0]=0.1288
X[4].probababilidad[1][1][0][0][0][0]=0.1378
X[4].probababilidad[1][1][1][0][0][0]=0.1352
Suma:                                =1.0000

------------------
*** JOIN TREE despues de Distribuye_Evidencia***
   1  2  3  4  5  6
1  0
2 -5  0
3  0  0  0
4  0  0  0  0
5  0  0 -4 -1  0
6  0 -3  0  0 -2  0
```

Parte VI
Referencias

Referencias

[1] David Heckerman. *A Tutorial on Learning with Bayesian Networks*, 1995 rev 1996. Technical Report MSR-TR-95-06. Microsoft Research, Advanced Technololgy Division. Microsoft Corporation. USA.

[2] Cecil Huang, Adnan Darwiche. *Inference in Belief Networks: A Procedural Guide*, International Journal of Approximate Reasoning. 1994 11:1-158. © 1994 Elsevier Science Inc. USA.

[3] Kozlov, Alexander., Pal Singh Jaswinder. *A Paralell lauritzen-Spielgelhalter Algorithm for Probabilistic Inference*, IEEE 1994. 1063-9535/94.

[4] Lauritzen, S.L., and Spiegelhalter, D.J. *Local computations with probabilities on graphical structures and their application to expert systems*, Journal of the Royal Statistical Soc. B50-253-258, 1988.

[5] Jensen, F. V., Lauritzen, S.L., and Olesen, K. G. *Bayesian updating in causal probabilistic networks by local computations*, Comp. Stat. Quart., 4, 269-282, 1990.

[6] Shenoy, P., and Shafer, G. *Axioms for probability and belief-function propagation*, Uncertainty and Artificial Intelligence, Vol. 4. (R.D. Shachter, T. Levitt, J.F. Lemmer and L.N. Kanal, Eds.), Elsevier, North-Holland, Amsterdam, 169-198, 1990.

[7] Mitchell, Tom. *Machine Learning*. McGraw-Hill. Boston, Massachusetts, 1997.

[8] Sara Baase, Allen Van Gelder. *Computer Algorithms: Introduction to Design and Analysis* 3rd Edition. Chapter 7. Addison-Wesley, Inc. 2000.

[9] Schreiber August Th,... [et al.]. *Knowledge Engineering and Management*, MIT Press, 2002.

[10] Iglesias, Garijo, González and Velasco. *A Methodological Proposal for Multiagent Systems Development extending CommonKADS*, MIX Consortium: Modular Integration of Connectionist and Symbolic Processing in Knowledge Based Systems, ESPIRIT-9119. http://citeseer.nj.nec.com/iglesias96methodological.html

[11] Object Manegement Group (OMG). *CORBA* http://www.omg.org/.

[12] Object Management Group (OMG). *Common Warehouse Metamodel*, http://www.omg.org/cwm/, Version 1.0, 2 February 2001.

[13] Homepage, *Data Mining Techniques.*, StatSoft, Inc., 1984-2003 http://www.statsoftinc.com/textbook/stdatmin.html

[14] Home Page, *ebizQ* IT Quadrant, Inc. http://www.ebizq.net/quicktakes/062600.html

[15] Ian H. Witten and Eibe Frank, *Data Mining, Practical Machine learning tools and techniques with Java implementations* , Morgan Kaufman, 2000. USA.

[16] Michael J. A. Berry and Gordon S. Linoff, *Data Mining Techniques for Marketing, Salesa and Customer Support*, Wiley, 1997, USA.

[17] The MathWorks. *Matlab version 6* www.mathworks.com.