Detección de Fallas Distribuida usando Redes Bayesianas de Múltiples Secciones

por

Ana Li Oña García

Tesis sometida como requerimiento parcial para obtener el grado de

Maestra en Ciencias Computacionales

por el

Instituto Nacional de Astrofísica, Óptica y Electrónica

Enero, 2019

Tonantzintla, Puebla

Supervisor:

Dr. Luis Enrique Sucar Succar

Dr. Eduardo Morales Manzanares

Coordinación de Ciencias Computacionales

INAOE

Índice general

Índice de figuras

Índice de tablas

Lista de acrónimos

AG Algoritmos Genéticos

CPT Tabla de Probabilidad Condicional

DAG Grafo Dirigido Acíclico

DLC Caso de Carga de Diseño

EF Discretización no supervisada de igual frecuencia

EMB Cobija de Markov Extendida

EW Discretización no supervisada de ancho equitativo

IA Inteligencia Artificial

JT Algoritmo de Árbol de Uniones

MAP *Mean Average Precision*

MB Cobija de Markov

MEM Máquina Eólica Mexicana

MSBN Redes Bayesianas de Múltiples Secciones

MSDAG DAG con múltiples secciones

P@i Precisión en las i primeras variables

RB Redes Bayesianas

RNA Redes Neuronales Artificiales

SVM Máquinas de Soporte Vectorial

Resumen

La detección de fallas en sistemas complejos ha adquirido una gran importancia, debido al posible impacto que tiene en la reducción de los costos de reparación o para evitar pérdidas de producción en los sistemas industriales. En la literatura científica se han propuesto diferentes enfoques para la detección de fallas, algunos de los cuales se basan en las Redes Bayesianas. Una ventaja de estos enfoques es que no requieren de datos de fallas, solo del sistema en condición de operación normal. Las Redes Bayesianas constituyen un formalismo adecuado para representar y razonar en condiciones de incertidumbre; sin embargo, a medida que aumenta la complejidad del dominio del problema inherente a la representación de sistemas complejos, los mecanismos de inferencia de este tipo de redes no son eficientes. Para superar esta limitación, los investigadores han propuesto las Redes Bayesianas de Múltiples Secciones. Estas son una extensión de las Redes Bayesianas para la representación de dominios grandes, al tiempo que se garantiza la inferencia de la red de manera eficiente. En este trabajo, proponemos un método distribuido para la detección de fallas en sistemas complejos que utiliza las Redes Bayesianas de Múltiples Secciones. El método fue probado en la detección de fallas en circuitos lógicos combinacionales y en simulaciones de una turbina eólica. Los resultados obtenidos muestran un desempeño comparable con el método centralizado basado en Redes Bayesianas en términos de precisión, pero con una reducción significativa del tiempo de ejecución.

Abstract

Faults detection in complex systems has acquired great importance, due to the possible impact it has on the reduction of repair costs or to avoid production losses in industrial systems. In the scientific literature, different approaches to faults detection have been proposed, some of which are based on Bayesian Networks. An advantage of these approaches is that they do not require fault data, only the system in normal operating condition. The Bayesian Networks constitute an adequate formalism to represent and reason in conditions of uncertainty; however, as the complexity of the domain of the problem inherent in the representation of complex systems increases, the inference mechanisms of this type of networks are not efficient. To overcome this limitation, researchers have proposed Multiply Sectioned Bayesian Networks. These are an extension of the Bayesian Networks for the representation of large domains while ensuring the inference of the network efficiently. In this work, we propose a distributed method for faults detection in complex systems that uses Multiply Sectioned Bayesian Networks. The method was tested in faults detection in combinational logic circuits and simulations of a wind turbine. The results obtained show a comparable performance with the centralized method based on Bayesian Networks regarding of accuracy, but with a significant reduction in execution time.

Capítulo 1

Introducción

1.1. Antecedentes

La detección de fallas es una parte importante de los sistemas de ingeniería y, a menudo, es un requisito previo para la puesta en servicio de un sistema, por lo tanto una característica fundamental que debe cumplir este tipo de sistemas lo constituye su robustez ante las posibles fallas. Algunos de los requisitos esenciales impuestos a los sistemas de ingeniería son niveles altos de confiabilidad, facilidad de mantenimiento y disponibilidad [Jovanovic, 2014]. En otras palabras, los sistemas de ingeniería deben ser sostenibles y esta es la razón principal por la que la detección de fallas es un aspecto importante a tener en consideración.

Un sistema complejo es un sistema formado por muchos componentes cuyo comportamiento es emergente, es decir, el comportamiento del sistema no puede deducirse simplemente del comportamiento de sus componentes. Una medida para evaluar la complejidad de un sistema es la cantidad de información necesaria para describir su comportamiento [Bar-Yam, 1997]. Ejemplos de sistemas complejos incluyen los modelos de economía actuales, el clima, el sistema nervioso, las infraes-

tructuras modernas de energía, entre otros.

Las investigaciones relacionadas con la detección de fallas aplicadas a los sistemas de ingeniería se han centrado en la utilización de diferentes técnicas estadísticas o de Inteligencia Artificial (IA), enfocando esfuerzos para detectar oportunamente comportamientos anormales a partir del análisis de los datos de las variables detectadas. En tal caso, se encuentran en el estado del arte, métodos simples basados en pruebas o en modelos físicos o matemáticos [Bertrand-Krajewski et al., 2007] [Branisavljević et al., 2011], donde a partir del análisis de los datos se clasifican los mismos en válidos, inválidos o faltantes; sin embargo, en la mayoría de los casos requieren de un modelo del sistema, lo que en algunos contextos resulta difícil de obtener.

La mayoría de las investigaciones estudiadas han utilizado las Redes Neuronales Artificiales, donde a partir de diferentes arquitecturas de red, tales como Perceptrones Multicapa [Eryurek and Upadhyaya, 1990] [Guo and Nurre, 1991] [Khadem et al., 1993] y Mapas Auto-Organizativos [Böhme et al., 1991] [Valentin et al., 2001], se aprenden los pesos de la red asociados al modelo y a partir de las redes entrenadas se detecta el comportamiento anormal de las variables. Sin embargo, estos trabajos requieren de la información completa de todas las variables involucradas en el modelo, por lo que algunos autores utilizan otras técnicas, tales como los Algoritmos Genéticos (AG), para tratar de predecir los valores de las variables faltantes.

Varios autores han investigado la detección de fallas, pero no todos tienen en cuenta la incertidumbre inherente que puede tener el fenónemo observado. En la práctica, la incertidumbre del modelo y el ruido en la medición pueden complicar la detección de fallas, de modo que los métodos desarrollados deben ser robustos a estas condiciones.

En este sentido, los trabajos encontrados en la literatura relacionados con la detección de fallas y que son robustos a la incertidumbre, se basan en el uso de técnicas de IA que se consideran robustas a la incertidumbre en los datos, en tal caso

destaca el uso de la Lógica Difusa [Holbert et al., 1994], cuya principal limitación radica en la necesidad del conocimiento experto para el aprendizaje de las funciones de membresía, y las Redes Bayesianas.

Las Redes Bayesianas (RBs), se presentan como un formalismo adecuado para la representación y el razonamiento de los sistemas bajo condiciones de incertidumbre [Sucar, 2015]. Constituyen una representación gráfica de dependencias entre las variables para el razonamiento probabilístico. Una RB es un grafo dirigido acíclico que describe la distribución de probabilidad conjunta de un conjunto de variables aleatorias.

En [Ibargüengoytia et al., 2005], los autores proponen un algoritmo para la validación de sensores, representando las relaciones entre las variables a partir de su modelación en una red bayesiana y actualizando las probabilidades posteriores a partir de un modelo de propagación probabilística. Los autores, estiman los valores de las variables, identificando la falla aparente a partir del valor esperado y el valor real, considerando al resto de las variables del modelo como evidencia. La novedad fundamental de este trabajo radica en que no necesita de información de fallas en el modelo para identificar el comportamiento anormal de las variables. Sin embargo, este trabajo no tiene en cuenta la complejidad del dominio del modelo y la naturaleza distribuida de sus diferentes componentes [Liu et al., 2015], aspectos que determinan la complejidad del mecanismo de inferencia en las RB.

Las Redes Bayesianas de Múltiples Secciones (MSBN), propuestas por [Xiang et al., 1993b], se presentan como una alternativa para el modelado de problemas de dominio grandes. MSBN identifica particiones del dominio en subdominios más pequeños que se comunican entre sí a partir de información compartida y donde el proceso de inferencia de la red global se realiza de manera eficiente.

1.2. Descripción de la problemática

Nuestra propuesta se centra principalmente en extender el trabajo presentado en [Ibargüengoytia et al., 2005] a un enfoque distribuido, utilizando la teoría de las Redes Bayesianas de Múltiples Secciones (MSBN). La principal limitación del enfoque centralizado radica en que no toma en cuenta la complejidad de los modelos de ciertos sistemas, tales como turbinas eólicas, circuitos electrónicos digitales, plantas eléctricas, entre otros, por lo que resulta difícil modelar y diagnosticar estos sistemas.

El modelo de un sistema complejo como una RB para detección de fallas no es eficiente en términos de la complejidad de la inferencia, es conocido que la inferencia en RBs en el peor de los casos es un problema NP-Duro [Cooper, 1990]. Por otro lado, este tipo de sistemas se pueden dividir naturalmente en partes, lo que facilita el uso de la técnica de MSBN para resolver este problema.

Las Redes Bayesianas de Múltiples Secciones constituyen una alternativa para modelar este tipo de sistemas por las siguientes razones:

- El modelo del dominio (por ejemplo los circuitos electrónicos digitales) pueden ser representados como un red bayesiana general.

- El dominio de aplicación puede ser particionado de forma natural en términos de localización (por ejemplo un circuito electrónico compuesto por varios componentes relativamente independientes que se comunican entre sí)

- Una falla o un conjunto de fallas aparentes en un componente se propagarán sólo sobre aquellos componentes con los que haya compartido información, aspecto que se puede considerar en el proceso de aislamiento de fallas.

1.3. Motivación y Justificación

Detectar oportunamente el comportamiento anormal de las lecturas de los sensores en el contexto de la detección de fallas en sistemas complejos es un aspecto importante en todos los sistemas de ingeniería, lo que se traduce en la reducción de costos de reparación o en la toma de decisiones asociadas a procesos de producción para evitar daños mayores sobre el equipo.

Las RB representan una técnica adecuada para la representación de los sistemas en condiciones de incertidumbre; sin embargo, para modelos complejos, la inferencia con este tipo de formalismo no es eficiente. La implementación de un algoritmo de detección de fallas distribuido permite modelar y diagnosticar en sistemas complejos, como circuitos electrónicos digitales o turbinas eólicas. El uso de las Redes Bayesianas de Múltiples Secciones (MSBN) permitirá la detección del comportamiento anormal de las variables de manera eficiente a través de su mecanismo de inferencia, lo que se traduce en una reducción del tiempo de ejecución para toda la red.

1.4. Pregunta de investigación

De manera general la pregunta de investigación que nos proponemos en esta tesis es: ¿En qué medida un algoritmo distribuido de detección de fallas basado en MSBN mejora al método centralizado basado en RB?

1.5. Hipótesis

Un algoritmo distribuido que utiliza las MSBN para la detección de fallas en sistemas complejos tendrá un desempeño comparable con el esquema centralizado

basado en Redes Bayesianas en cuanto a precisión con un menor tiempo de ejecución.

1.6. Objetivos

Objetivo general

Desarrollar un algoritmo distribuido para la detección de fallas en sistemas complejos usando Redes Bayesianas de Múltiples Secciones (MSBN).

Objetivos específicos

Los objetivos específicos derivados del objetivo general de esta tesis son:

- Adecuar los principales conceptos de las MSBN al contexto de la detección de fallas.
- Proponer una extensión del método de detección de fallas aparentes usando MSBN.
- Proponer un método distribuido para el aislamiento de las fallas.
- Aplicar el método de detección de fallas distribuido a diferentes dominios de aplicación tales como los circuitos lógicos digitales y las turbinas eólicas.

1.7. Solución Propuesta

A partir del trabajo presentado por [Ibargüengoytia et al., 2005], esta tesis se propone mejorar su principal limitación, relacionada con la ineficiencia de los mecanimos de inferencia de las Redes Bayesianas en correspondencia con la densidad

de la red. Para ello, proponemos el uso de la técnica de las Redes Bayesianas de Múltiples Secciones, donde se crean particiones del dominio y donde la inferencia se realiza de manera local para cada sección con las correspondientes actualizaciones a nivel global.

De manera general, en esta tesis se propone un método de detección de fallas aparentes con el uso de las MSBN, además de un método de aislamiento de fallas donde se toman las ideas propuestas en [Ibargüengoytia et al., 2005], pero se toma ventaja de la distribución en secciones del dominio del problema para realizar el proceso de aislamiento.

Los experimentos se realizaron sobre dos dominios de aplicación diferentes. Como un primer dominio se tomaron diferentes ejemplos de circuitos lógicos digitales separados por componentes relativamente independientes que constituyen las secciones. A partir de la simulación del comportamiento normal de los circuitos se obtuvo el modelo y posteriormente se modificó el diseño del circuito para simular tres tipos de fallas diferentes: compuertas lógicas pegadas a cero, pegadas a 1 y negar la salida de alguna compuerta. El segundo dominio sobre el cual se realizaron los experimentos lo constituyen las simulaciones de la Máquina Eólica Mexicana (MEM) en el simulador FOCUS6 [WMC, 2018], donde los casos de carga de diseño modelan el comportamiento normal de la turbina para diferentes condiciones de viento, a partir de estos datos se obtiene el modelo particionado en subdominios tratando de mantener la localización de las variables asociadas a cada componente físico de la turbina. El simulador permite además la simulación de diferentes comportamientos fallidos tales como: paso fuera de control en un aspa, aspa bloqueada, corto circuito en el aerogenerador, falla en el controlador del paso, sistema de guiñada fuera de control y desconexión de la red eléctrica.

Conforme a los resultados obtenidos nuestra propuesta muestra resultados similares en cuanto a precisión con el método propuesto por [Ibargüengoytia et al.,

2005], además de una reducción significativa del tiempo de ejecución asociado a su naturaleza distribuida.

1.8. Principales Contribuciones

La principal contribución de esta investigación resulta en un algoritmo distribuido de detección de fallas basado en las Redes Bayesianas de Múltiples Secciones que es más eficiente en términos de tiempo de ejecución y requerimientos de memoria que la versión centralizada.

1.9. Organización de la tesis

El resto del documento se organiza de la siguiente manera. En el capítulo 2 se presentan los principales conceptos relacionados con las RB y las MSBN, así como una descripción de los principales trabajos que desarrollan la teoría de las MSBN. El capítulo 3 resume los conceptos y trabajos encontrados en el estado del arte relativos a los métodos de detección de fallas. El capítulo 4 describe el método propuesto, especificando cada uno de los pasos a partir de un ejemplo. El capítulo 5 muestra el diseño experimental así como los principales resultados obtenidos. En el último capítulo se presentan las conclusiones y el trabajo futuro de esta tesis.

Capítulo 2

Redes Bayesianas de Múltiples Secciones

En este capítulo se introducen los conceptos básicos requeridos para la comprensión de la solución propuesta. Específicamente, se describen las generalidades de la teoría de las Redes Bayesianas y de las Redes Bayesianas de Múltiples Secciones (MSBN, por sus siglas en inglés) como una extensión de la primera, pues constituye la técnica utilizada para la solución propuesta, además, se detallan los principales trabajos que desarrollan la técnica de MSBN. Por último, se incluye un análisis de la escalabilidad de método propuesto con el uso de esta técnica de MSBN.

2.1. Redes Bayesianas

Esta sección describe los conceptos básicos de las Redes Bayesianas, tomados de [Sucar, 2015].

Una **Red Bayesiana** (RB) representa la distribución de probabilidad de un

conjunto de n variables[1] a partir de su representación como un grafo dirigido acíclico (DAG) y un conjunto de tablas de probabilidad condicional (CPT). Cada nodo o variable, tiene asociado una CPT que contiene la probabilidad de cada estado de la variable dado los estados de sus padres en el grafo. La estructura de la red implica relaciones de independencia condicional que se pueden verificar directamente en su estructura a partir del concepto de **Separación-D** que se describe a continuación.

Separación-D: Dado un grafo G, un conjunto de variables A es condicionalmente independiente de un conjunto B dado un conjunto C, si no existe una trayectoria en G entre A y B tal que:

1. Todos los nodos convergentes son o tienen descendientes en C.

2. Todos los demás nodos están fuera de C.

La figura 2.1 muestra un ejemplo trivial de Red Bayesiana que representa la distribución de probabilidad conjunta $P(A, B, C, D, E) = P(A)P(B)P(C|A)P(D|A, B)P(E|C)$ y las CPT correspondientes a cada variable (suponiendo variables binarias) donde, por ejemplo, $P(C = C1|A = A1) = 0.7$.

La representación de una Red Bayesiana R se define como una tripleta (V, G, P) donde V es el conjunto de variables del dominio, G es el DAG cuyos nodos están etiquetados por elementos de V y P es la distribución de probabilidad conjunta sobre V, donde para los nodos raíz será el vector de probabilidades marginales y para el resto de los nodos la CPT de cada variable dado sus padres en el grafo. La representación de G asume que cada variable x es independiente de sus no descendientes dados sus padres $Pa(x)$.

La **Cobija de Markov** (MB por sus siglas en inglés) de una variable X de una Red Bayesiana G, $MB(X)$, es el conjunto de nodos que hacen independiente a

[1]En lo que sigue asumimos variables discretas, aunque existen trabajos para variables continuas

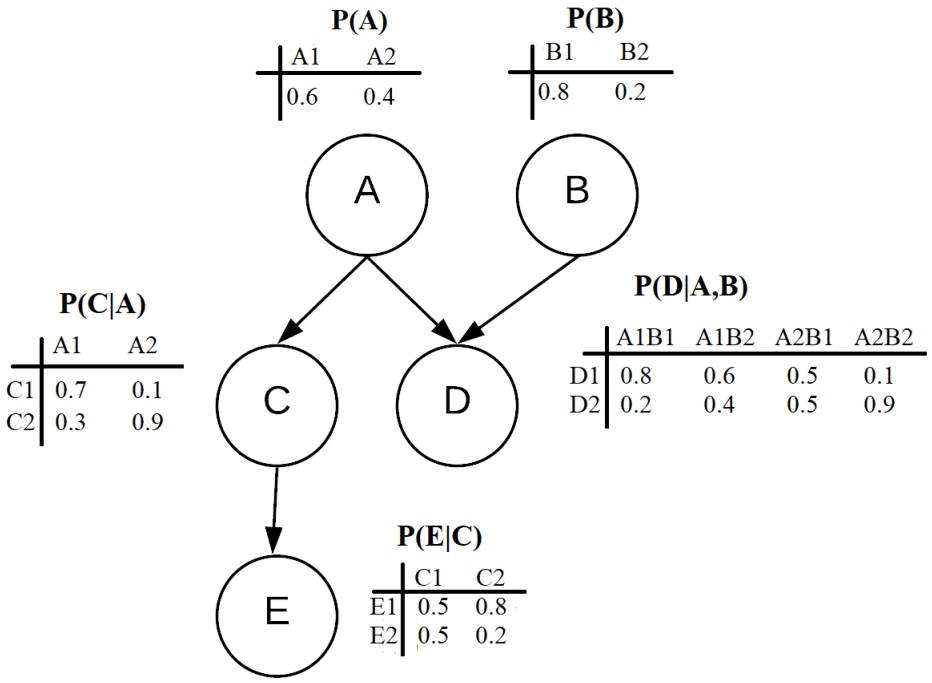

Figura 2.1: Ejemplo de Red Bayesiana.

X del resto de los nodos en G, es decir, $P(X|G-X)$=$P(X|MB(X))$. Para una RB, la Cobija de Markov de X se define como:

- los padres de X
- los hijos de X
- otros padres de los hijos de X

Para el ejemplo de la figura 2.1 la cobija de markov de A, $MB(A) = \{C, D, B\}$. La Cobija de Markov Extendida (EMB) de una variable es su Cobija de Markov incluyendo a la propia variable, $EMB(A) = \{A, C, D, B\}$.

En este trabajo asumimos las Redes Bayesianas con variables discretas o nominales para la representación de los nodos (existen trabajos de RB con variables continuas pero están limitados a variables gaussianas y relaciones lineales), por lo

que para aquellos dominios de problemas cuyas variables no los sean, será necesario discretizarlas antes de construir el modelo.

Discretización de variables continuas

Los **métodos de discretización** de variables continuas pueden clasificarse en métodos manuales, métodos no supervisados y métodos supervisados. En [Dougherty et al., 1995] se presenta una introducción general a los principales métodos de discretización que se resumirán a continuación.

- La discretización manual (también conocida como discretización de expertos) implica que el usuario seleccione manualmente los umbrales de discretización en función de su significado físico, conocimiento teórico o su interpretación experta del dominio del problema [Chen and Pollino, 2012]. Puede apoyarse en el uso de herramientas tales como histogramas o árboles de regresión para comprender mejor los umbrales presentes en los datos. En [Aguilera et al., 2011], los autores notaron, sobre la revisión de un conjunto de artículos sobre la aplicación de BN en entornos ambientales, que la discretización manual era el método más comunmente usado en estudios que requerían discretización de datos continuos. Este método a menudo se ve favorecido porque permite discretizar las variables continuas a intervalos interpretables y relevantes para los datos o los objetivos del modelo y no se requiere ningún algoritmo de discretización o cálculo adicional.

- La discretización no supervisada se basa en la distribución de datos intrínsecos de cada variable individual. Es popular porque es computacionalmente simple y objetiva. Los algoritmos de discretización no supervisada más populares son los de ancho equitativo (EW) e igual frecuencia (EF). La discretización EW divide los datos continuos en un número predefinido de intervalos de igual

ancho. Este método puede funcionar bien con distribuciones continuas aproximadamente uniformes, pero genera intervalos inapropiados de probabilidades desequilibradas cuando los datos son muy sesgados o contienen valores atípicos [Chen and Pollino, 2012]. La discretización *EF* divide los datos continuos en un número predefinido de intervalos de igual frecuencia. Esta discretización genera una distribución uniforme (no informativa) de los datos continuos pero puede ocultar valores atípicos en los datos y en los casos donde hay una alta frecuencia del mismo valor, ese valor puede ser forzado a dividirse en diferentes intervalos. Para aplicar la discretización *EW* o *EF*, se debe especificar el número de intervalos para dividir los datos. La mayoría de las aplicaciones de RB suelen utilizar entre 2 y 10 intervalos.

- La discretización supervisada es un método informativo de discretización que utiliza el estado de salida de la variable para informar y optimizar la discretización individual de cada variable de entrada. Al igual que la discretización no supervisada, un inconveniente de los algoritmos de discretización supervisada es que los umbrales que producen a menudo carecen de sentido físico. Además, los algoritmos supervisados pueden producir umbrales de discretización potencialmente adulterados que se ajustan al ruido en los datos en lugar de los umbrales que aumentan la capacidad predictiva de la RB. La discretización supervisada también requiere una variable de salida discreta para informar la discretización de las variables de entrada continua. Esto significa que, si la variable de salida es continua, se necesitarían conocimientos a priori, suposiciones o un método de discretización no supervisado para discretizarla antes de que las variables de entrada puedan discretizarse utilizando un método supervisado. Si bien hay muchos algoritmos de discretización supervisada disponibles, los algoritmos de *Fayyad & Irani* (F&I) [Fayyad and Irani, 1993] y *Kononenko* (KO) [Kononenko, 1995] son los más populares. Ambos algoritmos se basan en la minimización de la entropía y se iteran eficazmente a través de esquemas de

discretización para encontrar el que maximiza la predictibilidad de la variable de salida (minimiza la entropía). Las variables de entrada se optimizan individualmente y pueden tener diferentes números de intervalos de discretización. El algoritmo de KO difiere del algoritmo de F&I en que da cuenta del sesgo de entropía introducido en las configuraciones multivariadas cuando las variables tienen diferentes números de intervalos de discretización [Kononenko, 1995].

Inferencia en Redes Bayesianas

La inferencia o la propagación de las probabilidades en la Redes Bayesianas consiste en propagar por la estructura de la red los efectos de nuevo conocimiento, conocido como evidencia, para actualizar las probabilidades de las variables dado este nuevo conocimiento. En [Cooper, 1990], los autores demostraron que la inferencia en Redes Bayesianas multiconectadas constituye un problema NP-Duro dependiente de las variables y de las conexiones entre las mismas, por lo que, al aumentar la densidad de las conexiones de la red, la complejidad de la inferencia aumenta exponencialmente.

Dada la complejidad de los métodos de inferencia con respecto a la estructura de la red, se han desarrollado diferentes algoritmos de inferencia que dependen en gran medida de la estructura del grafo y de si obtienen las probabilidades posteriores de una o de todas las variables.

Entre los métodos de inferencia más conocidos destacan: el **Algoritmo de Propagación de Creencias** [Pearl, 1986a], para grafos con estructura de árbol o poliárboles (grafos sencillamente conectados donde un nodo puede tener más de un padre); el **Algoritmo de Eliminación de Variables** [Zhang and Poole, 1996], donde para cualquier estructura de grafo se calcula la probabilidad posterior de una variable a partir de la marginalización de su distribución de probabilidad conjunta; los algoritmos de agrupamiento donde destaca la **Técnica de Árbol de Unión**

(*Junction Tree*), que se describirá a más detalle a continuación pues constituye la técnica en la que se basa esta tesis; el **Algoritmo de Condicionamiento** [Díez, 1996] que se basa en transformar el grafo multiconectado en un poliárbol para aplicar el algoritmo de propagación de creencias; y la **Simulación Estocástica** [Pearl, 1987], que constituye un método aproximado donde se simula la red bayesiana, en cada simulación se inicializa con un valor posible todas las variables no instanciadas.

Algoritmo de Árbol de Uniones: Para grafos multiconectados uno de los métodos de inferencia más común lo constituye la técnica de Árbol de Unión [Lauritzen and Spiegelhalter, 1988], la cual constituye un método de agrupamiento a partir de la transformación de la RB en una nueva estructura llamada *junction tree* donde cada nodo en el árbol está formado por el agrupamiento de un grupo de variables de la red original y la inferencia probabilística se realiza sobre esta nueva estructura donde cada agrupamiento actúa como una unidad para el paso de mensajes.

La transformación de la red consta de los siguientes pasos:

1. Eliminación de la dirección de los arcos, ordenamiento de los nodos y moralización del grafo (adición de un nuevo arco entre padres con hijos comunes).

2. Triangulación del grafo. Un grafo se dice triangulado si cada circuito simple de longitud mayor que 3 en el grafo tiene un acorde. Un acorde es una arista que conecta dos de los vértices en el circuito y no es parte de ese circuito.

3. Obtención de los cliques del grafo (subconjunto maximal de nodos completamente conectados).

4. Construcción del *junction tree* donde cada nodo se corresponde con un clique de la red y su padre es cualquier nodo que contiene todas las variables previas comunes según el orden.

La figura 2.2 representa la transformación de la Red Bayesiana del ejemplo de la figura 2.1 a su correspondiente árbol de unión.

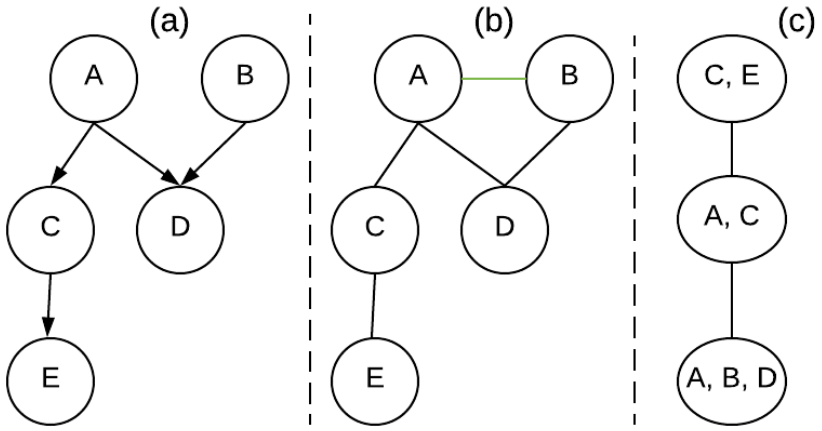

Figura 2.2: Obtención del árbol de unión correspondiente a la RB del ejemplo de la figura 2.1: **(a)** Red Bayesiana original, **(b)** grafo triangulado, **(c)** árbol de unión.

A partir de esta representación con estructura de árbol y la probabilidad conjunta o potencial de cada clique, la inferencia se realiza de forma similar al método de Propagación de Creencias para obtener la probabilidad posterior de cada unión, $P'(C_i)$, siendo C_i cada uno de los cliques del grafo. La probabilidad individual de cada variable se obtiene a partir de la marginalización de la probabilidad conjunta de la unión, $P(X) = \sum_{C_i - X} P'(C_i)$.

El algoritmo de Árbol de Unión, al constituir un agrupamiento de las variables del modelo de Red Bayesiana, permite la escalabilidad de los modelos a la hora de realizar la inferencia (la complejidad del método de árbol de uniones depende del tamaño del clique más grande) por lo que constituye la técnica en la que se basan las Redes Bayesiana de Múltiples Secciones, que a su vez constituye la técnica utilizada en esta tesis y cuyas generalidades se describirán en la siguiente sección.

2.2. Redes Bayesianas de Múltiples Secciones

Las Redes Bayesianas de Múltiples Secciones (MSBN por sus siglas en inglés) fueron propuestas por [Xiang et al., 1993b] para la representación de dominios de problemas grandes cuya representación de Red Bayesiana se caracteriza por tener una topología de red dispersa y por evidencia incremental. Constituye una extensión de la técnica de Árbol de Unión mencionada anteriormente, por lo que toma ventaja del agrupamiento de las variables en el proceso de inferencia para la representación y el procesamiento del dominio a modelar. Aunque el sistema general puede ser grande, los requisitos computacionales se rigen por el tamaño de los árboles de unión. El ejemplo de la figura 2.3 guiará el desarrollo de las definiciones relacionadas con las MSBN.

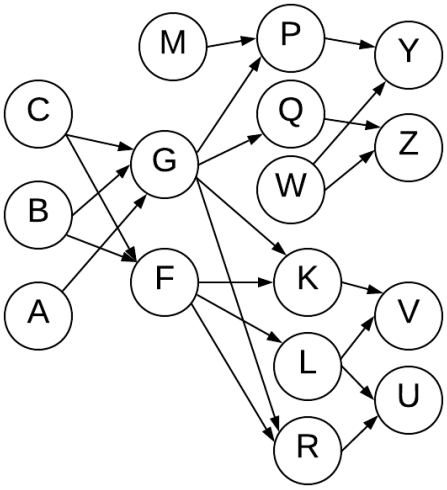

Figura 2.3: Red Bayesiana, M, que ejemplificará las definiciones relacionadas con las MSBN.

Una MSBN M es un conjunto de subredes bayesianas que juntas definen una Red Bayesiana general. M representa la dependencia probabilística de un dominio particionado en múltiples subdominios donde cada uno de ellos es representado por

una subred [Xiang et al., 2005].

Para garantizar la inferencia exacta en las MSBN, las subredes deben satisfacer ciertas condiciones. Primero, se introduce la terminología necesaria para describir estas condiciones. Los principios matemáticos, así como las definiciones fueron extraídos del trabajo presentado en [Xiang, 2003].

Sean $G_i = (V_i, E_i)(i = 0, 1)$ dos grafos distintos (dirigidos o no dirigidos). G_0 y G_1 se dice que son *grafos consistentes* si los subgrafos de G_0 y G_1 obtenidos de $V_0 \cap V_1$ son idénticos. Dados los grafos consistentes $G_i = (V_i, E_i)(i = 0, 1)$, el grafo $G = (V_0 \cup V_1, E_0 \cup E_1)$ es llamado *unión* de G_0 y G_1, denotado por $G = G_0 \sqcup G_1$.

Dado un grafo $G = (V, E)$, V_0 y V_1 tal que $V_0 \cup V_1 = V$ y $V_0 \cap V_1 \neq \emptyset$, y subgrafos G_i de G obtenidos de $V_i(i = 0, 1)$, podemos decir que G está *seccionado* en G_0 y G_1. La figura 2.4 muestra el DAG M de la figura 2.3 seccionado en $M1, M2, M3$.

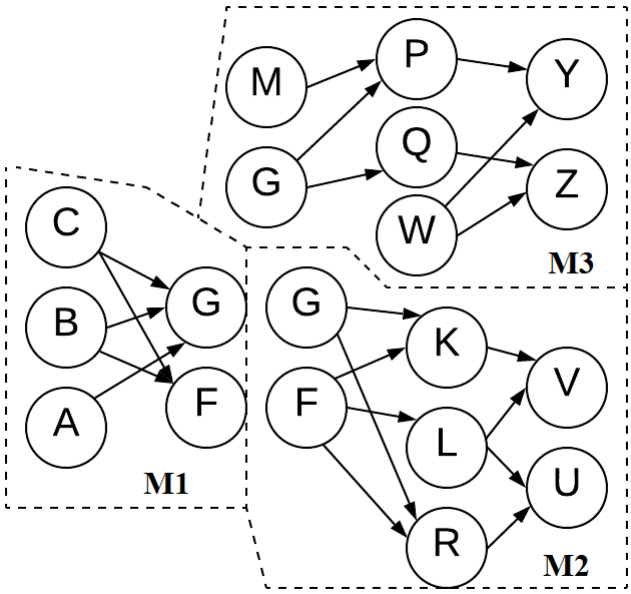

Figura 2.4: DAG M seccionado en $M1, M2, M3$.

Las MSBN deben satisfacer las siguientes condiciones:

1. Las subredes en una MSBN deben satisfacer la condición de hyper-árbol (*hypertree*).

   Definición 1. Sea $G = (V, E)$ un grafo conectado seccionado en subgrafos $G_i = (V_i, E_i)$ tal que G_i está asociado con el árbol Ψ con la siguiente propiedad: Cada nodo en Ψ está etiquetado por un G_i y cada enlace entre G_k y G_m está etiquetado por la *interface* $V_k \cap V_m$ tal que para cada i y j, $V_i \cap V_j$ está contenido en cada subgrafo en el camino entre G_i y G_j en Ψ. Entonces Ψ es el hyper-árbol (*hypertree*) sobre G. Cada G_i es un hyper-nodo y cada interface es un hyper-enlace (*hyperlink*).

   La figura 2.5 muestra el hyper-árbol asociado al ejemplo de la figura 2.4.

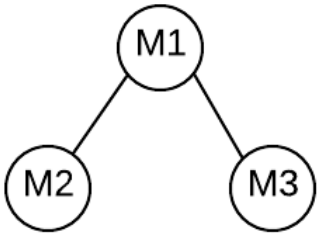

Figura 2.5: Estructura de hyper-árbol del ejemplo de la figura 2.4.

2. La *interface* entre subredes en una MSBN debe formar un *d-sepset*.

   Definición 2. Sea G un grafo dirigido tal que un hyper-árbol sobre G existe. Sea x un nodo que está contenido en más de un subgrafo y $\pi(x)$ son sus padres en G. Entonces x es un d-sepnodo si existe un subgrafo que contiene a $\pi(x)$. Una interface I es un *d-sepset* si toda $x \in I$ es un d-sepnodo.

   Para el ejemplo de la figura 2.4 el *d-sepset* entre $M1$ y $M2$ son los d-sepnodos G, F y el *d-sepset* entre $M1$ y $M3$ es el d-sepnodo G.

3. La estructura en una MSBN es un Grafo Dirigido Acíclico(DAG) con múltiples secciones (MSDAG) con una organización de hyper-árbol.

Definición 3. Un hyper-árbol MSDAG $G = \bigsqcup_i G_i$, donde cada G_i es un DAG, es un DAG conectado tal que (1) existe un hyper-árbol Ψ sobre G, y (2) cada hyper-enlace en Ψ es un *d-sepset*.

En estas condiciones, una MSBN se define como:

Definición 4. Una MSBN M es una tripleta (V, G, P). $V = \bigcup_i V_i$ es el *dominio* donde cada V_i es un conjunto de variables, llamadas *subdominio*. $G = \bigsqcup_i G_i$ (un hyper-árbol MSDAG) es la *estructura* donde los nodos en cada DAG G_i están etiquetados por elementos de V_i. Sea x una variable y $\pi(x)$ todos los padres de x en G. Para cada x, exactamente una de sus ocurrencias (en un G_i) contiene $\{x\} \cup \pi(x)$, a esta se le asigna $P(x|\pi(x))$, y a cada ocurrencia en otros DAGs se le asigna un potencial uniforme. $P = \prod_i P_i$ es la distribución de probabilidad conjunta (*jpd*), donde cada P_i es el producto de los potenciales asociados con los nodos en G_i. La tripleta $S_i = (V_i, G_i, P_i)$ es una *subred* de M. Dos subredes S_i y S_j son adyacentes si G_i y G_j son adyacentes.

El mecanismo de inferencia en las MSBN se realiza a través del paso de mensajes. Cada mensaje es un potencial sobre un subconjunto de variables. La inferencia local dentro de cada sección pasa mensajes intra-subredes lo que garantiza la consistencia en las subredes. La comunicación entre agentes o secciones transfiere mensajes entre subredes, lo que hace que el sistema multiagente tenga consistencia global.

Para garantizar la inferencia exacta de forma eficiente, cada sección (vistas a partir de este momento como un agente independiente) compila su subred en un árbol de uniones (JT). La compilación es un proceso cooperativo entre todos los agentes. Primero, se realiza una moralización y triangulación distribuidas. La figura 2.6 muestra los grafos moralizados y triangulados correspondientes a cada sección del ejemplo de la figura 2.4. Luego cada agente construye su JT local. Posteriormente se realiza una asignación de potencial local: para cada variable en la subred del agente, su tabla de probabilidad condicional está asociada con un grupo en el JT que contiene la variable y sus padres en la subred. Cada grupo en cada JT tiene asignado un único

potencial sobre su conjunto de variables miembro, que es el producto de todas las tablas de probabilidad condicionales asociadas con el grupo.

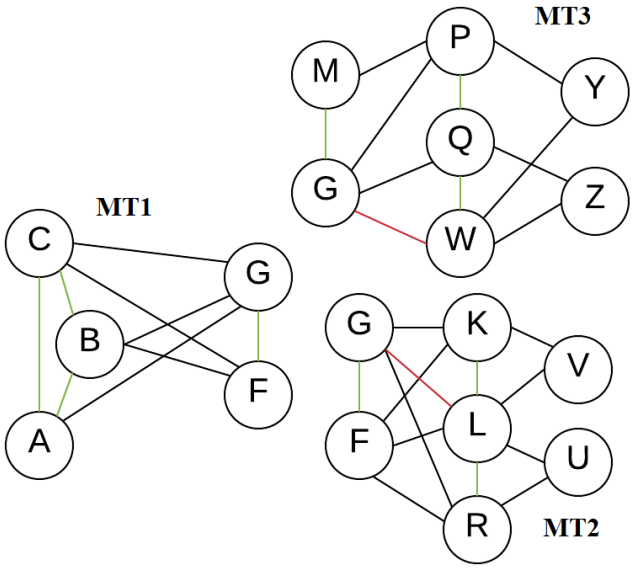

Figura 2.6: Grafos moralizados y triangulados del ejemplo de la figura 2.4. Las aristas verdes indican arcos adicionados en el proceso de moralización. Las rojas son las agregadas en la triangulación.

Para la comunicación entre secciones adyacentes se construye una estructura de árbol de enlace o *linkage tree*, esta nueva estructura constituye un árbol de uniones sobre el *d-sepset* de secciones adyacentes. Si el agente tiene adyacentes k secciones en el hyper-árbol, entonces está asociado con k *d-sepsets* y los compilará en k árboles de enlace. Las principales propiedades de los árboles de enlace se listan a continuación. El proceso de construcción de los árboles de enlace se detalla en [Xiang et al., 1993b].

- Un árbol de enlace es un JT sobre un *d-sepset*, donde cada nodo es un subconjunto del *d-sepset*, llamado enlace, y cada arco se denomina separador de enlace.
- Dado un JT de una sección y uno de sus árboles de enlace, cada nodo en el

árbol de enlace es un subconjunto de al menos un nodo en el JT. Uno de estos grupos se designa como el host de enlace.

- Un árbol de enlace expresa la misma separación gráfica dentro del *d-sepset* por lo tanto, codifica relaciones de independencia idénticas dentro del *d-sepset* como su JT correspondiente.

- Como un *d-sepset* implica dos agentes adyacentes, el árbol de enlace derivado por un agente es equivalente al derivado por el otro.

La figura 2.7 muestra los JT de cada sección del ejemplo 2.4 y sus correspondientes árboles de enlace.

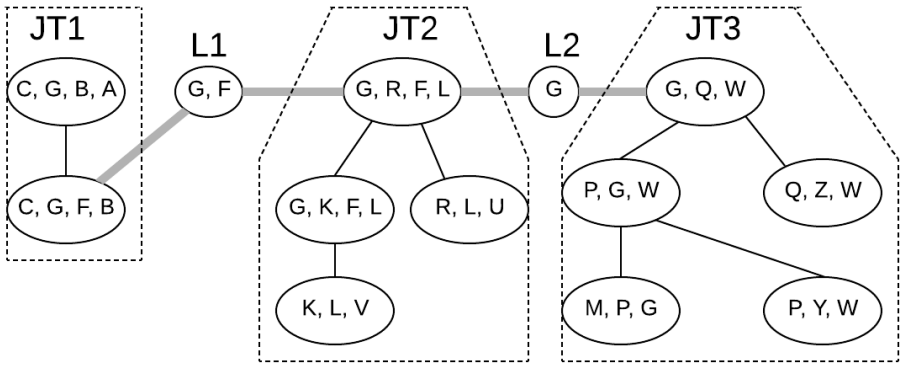

Figura 2.7: JT correspondientes a cada sección del ejemplo de la figura 2.4 y los árboles de enlace entre secciones adyacentes.

La estructura de árboles de uniones locales relacionados por árboles de enlace se denomina bosque de uniones de enlaces (*linked junction forest*).

El último paso de la compilación es la **inicialización de creencias**. En este paso se realiza una comunicación de todo el sistema para que los agentes intercambien conocimientos previos sobre variables compartidas. La comunicación consiste en una ronda de propagación de mensajes hacia adentro a lo largo del hyper-árbol y una

ronda de propagación hacia afuera. El mensaje enviado por un agente a un agente adyacente consiste en un conjunto de potenciales de vinculación, los detalles de este paso de mensajes se explican en [Xiang et al., 1993b]. Después de la inicialización de creencias, cada agente puede realizar una inferencia autónoma y responder a consultas exactas en relación con el conocimiento previo de todos los agentes y las observaciones propias.

De manera general, el uso de las MSBN, a pesar de que puede requerir mayor tiempo en la construcción del modelo (pues requiere del cumplimiento de las restricciones antes expuestas), el razonamiento evidencial es significativamente más rápido pues explota el principio de localidad sobre el cual se basa el modelo.

2.3. Avances en la teoría de las Redes Bayesianas de Múltiples Secciones

En esta sección se expone un resumen de los principales trabajos encontrados en el estado del arte relacionados con las Redes Bayesianas de Múltiples Secciones (MSBN).

Las MSBN fueron propuestas por [Xiang et al., 1993b] para la representación de problemas de dominio grande. En este trabajo se dan los fundamentos matemáticos que demuestran las limitaciones de las Redes Bayesianas a medida que aumenta el tamaño del modelo del dominio. Además se explican los fundamentos teóricos sobre los cuales se basa esta técnica de MSBN, así como se introducen las restricciones necesarias para garantizar la inferencia exacta en este tipo de red. Por otra parte, se explican las generalidades del mecanismo de inferencia tanto a nivel local como global. Este trabajo constituye la introducción teórica de este tipo de redes.

En [Xiang et al., 1993a] se presenta una aplicación a un problema de diagnósti-

co neuromuscular donde se introducen los principales problemas de representación del conocimiento que surgen en muchos dominios de aplicación y que motivaron el surgimiento de esta técnica de MSBN. En este trabajo se muestran los resultados de esta técnica desde el enfoque de un agente único; es decir, el modelo del dominio se particiona en subdominios, pero el procesamiento de cada subdominio se realiza de manera centralizada en un único agente.

Posteriormente, en [Xiang, 1996], se extiende el uso de esta técnica a un enfoque multiagente donde cada agente constituye una sección del dominio del problema, lo que permite el procesamiento de forma paralela. En este trabajo se introducen nuevas operaciones de comunicación para mantener la consistencia global; además, se redefine el mecanismo de inferencia tomando ventaja de la representación distribuida de los agentes.

Relativo a los mecanismos de inferencia en las MSBN, se han propuesto mejoras en el método. Por ejemplo, en [Xiang, 2000] se propone una mejora en cuanto a las operaciones de actualización de las creencias entre secciones adyacentes. Se redefinen estas operaciones de forma tal que sólo se requieren dos operaciones de actualización de creencias entre dos secciones adyacentes. Además, se demuestra que estas nuevas operaciones no comprometen la consistencia global de la red, a la vez que mejoran la eficiencia del mecanismo de inferencia. Los resultados con esta nueva modificación se resaltan a partir de su conexión con los métodos de inferencia tradicionales para las Redes Bayesianas.

En [Xiang et al., 2005], partiendo de que las MSBN constituyen una extensión de la técnica de árbol de unión y que en esta técnica la comunicación se realiza a través del paso de mensajes, se comparan los distintos métodos de paso de mensajes existentes en la literatura para los árboles de uniones y su posible extensión en el mecanismo de inferencia en las MSBN. Se realiza además un análisis de la complejidad de estos métodos así como una comparación análitica y experimental de los mismos.

El análisis proporciona a quienes implementan sistemas de inferencia probabilística multiagente una guía sobre los pros y los contras de los métodos alternativos, además de revelar problemas que deben considerarse al investigar otros métodos de inferencia orientados a agentes únicos.

Otros trabajos se enfocan en mejoras del método a nivel global [Xiang and Jensen, 2013], relativo a la comunicación entre los agentes; mientras otros proponen mejoras relativas a la inferencia dentro de las subredes [Xiang, 2013].

Un aspecto interesante a tener en consideración resulta en trabajos que aborden el tema de agrupar las Redes Bayesianas con el fin de hacer el proceso de inferencia de manera menos costosa.

En este sentido existen intentos de descomponer las Redes Bayesianas homogéneas para lograr diferentes objetivos, sin embargo, no abordan explícitamente la localización de las redes en subredes más pequeñas. El trabajo inicial relevante en ese sentido incluye un método de Pearl [Pearl, 1986b] que permite posponer la propagación de la evidencia en una red jerárquica de hipótesis.

El podado de las redes bayesianas [Baker and Boult, 1990] constituye otro de los trabajos donde se reduce el costo computacional con respecto a cada instancia de consulta; sin embargo, el método no proporciona soporte general para pruebas incrementales.

[Heckerman, 1990] divide las redes bayesianas en pequeños grupos de variables relacionadas naturalmente para facilitar la construcción de redes grandes. Pero una vez que se finaliza la construcción, la representación del tiempo de ejecución sigue siendo homogénea.

Por otra parte [Suermondt et al., 2013] combinan el condicionamiento de conjunto de corte con el método de inferencia de árbol de uniones y convierten la red original en un conjunto de árboles de cliques para obtener ahorros computacionales.

El conjunto de corte es elegido principalmente basado en la topología de la red. Este método no conduce a la explotación de algunas formas de localización. Esto se debe a que las variables que preservan la localización pueden estar separadas en diferentes particiones. Una consecuencia puede ser cambios frecuentes de cálculo entre las particiones que contienen estas variables.

De manera general estos trabajos pueden constituir el punto de partida para el desarrollo de un método automático de construcción de las secciones que no se incluye en esta investigación.

2.4. Análisis de escalabilidad.

En esta sección se realiza un análisis del término de escalabilidad de un sistema así como de la complejidad computacional asociada al método propuesto. Para ello se parte de la definición de escalabilidad asumida en esta investigación, además de realizar una demostración a partir de la cual se sustenta la investigación realizada en esta tesis, donde se desea aumentar la escalabilidad del método de detección de fallas que usa redes bayesianas a partir del uso de la técnica de MSBN.

La escalabilidad es un término bastante impreciso, incluso diferentes autores asumen la definición de escalabilidad que más se ajuste al problema a tratar. El análisis general del término que se asume en esta investigación se basa en las ideas propuestas en [Hill, 1990], [Wilkinson et al., 2005] y [Lehrig et al., 2015].

La **escalabilidad** es un término usado para referirse a la propiedad de aumentar la capacidad de trabajo o de tamaño de un sistema sin comprometer el funcionamiento y calidad normales del mismo.

El rendimiento de un sistema depende, en gran medida, del tamaño del mismo, por ejemplo del número de procesadores. En general mientras más grande sea el

sistema mayor redimiento, pero eso conlleva un costo. Esto es lo que se conoce como arquitectura o escalabilidad de hardware.

La escalabilidad también se usa para indicar que un algoritmo paralelo puede acomodar elementos de datos incrementados con un aumento bajo y limitado en pasos computacionales. Esto se describe como escalabilidad de algoritmo y constituye el tipo de escalabilidad que se desea mejorar en esta tesis.

2.4.1. Demostración teórica de la escalabilidad del método propuesto

En [Cooper, 1990] se demostró que la inferencia probabilística usando Redes Bayesianas con múltiples conexiones constituye un problema NP-duro. La inferencia de las redes bayesianas conectadas individualmente en el tiempo es lineal en función del tamaño de la red de creencias. Sin embargo, la inferencia usando redes bayesianas con múltiples conexiones constituye un problema más complejo.

Los algoritmos que implementan la inferencia en este tipo de redes tienen una complejidad, en el peor caso, exponencial en función de la topología de la red.

Partiendo que la tesis propuesta tiene como dominio de aplicación los sistemas complejos y su representación de este tipo de problemas como una red bayesiana, la complejidad de la inferencia de este tipo de redes resulta exponencial.

Sea S una red bayesiana de conexiones múltiples. Sea $S_1, ..., S_n$ particiones o subdominios de S (constituyen a su vez redes bayesianas con conexiones múltiples).

Suponiendo que $T(x)$ es la función que expresa el tiempo de ejecución de la inferencia de S, del análisis anterior se conoce que $T(x) = O(f(x))$ siendo $f(x)$ de orden exponencial.

Si $T_1(x), T_2(x), ..., T_n(x)$ son las funciones que expresan los tiempos de ejecución asociados a la inferencia de las particiones $S_1, ..., S_n$, y se acotan de forma que se tie-

ne: $T_1(x) = O(f_1(x)), T_2(x) = O(f_2(x)), ..., T_n(x) = O(f_n(x))$ siendo $f_i(i = 1, ..., n)$ de orden exponencial.

Entonces $T_1(x) + T_2(x) + ... + T_n(x) = O(max(f_1(x), f_2(x), ..., f_n(x)))$ por la regla de la suma

$O(max(f_1(x), f_2(x), ..., f_n(x))) \leq O(f(x))$ pues $T_1(x), T_2(x), ..., T_n(x) \leq T(x)$ pues $S1, ..., Sn$ constituyen particiones de S.

Todo lo descrito anteriormente demuestra que el uso de las MSBN (que constituyen particiones del dominio del problema en subdominios más pequeños) permite la escalabilidad de algoritmo del proceso de detección de fallas.

2.5. Resumen del capítulo

En este capítulo se presentaron los principales conceptos requeridos para esta investigación, inicialmente se expusieron las generalidades de la Redes Bayesianas sobre las cuales se basa el trabajo presentado por [Ibargüengoytia et al., 2005], y que constituye el antecedente directo de esta investigación. Además, se explica el método de inferencia basado en árboles de uniones. Por otra parte se resumen los conceptos fundamentales de MSBN, pues constituye la técnica a utilizar en esta tesis y se describen los principales trabajos encontrados en el estado del arte que desarrollan la teoría de MSBN. Por último se realiza un análisis teórico de la complejidad computacional del método propuesto que permite mayor escalabilidad del método de detección de fallas usando redes bayesianas.

En el siguiente capítulo se presenta el trabajo relacionado con los temas de esta investigación, incluyendo el estado del arte relativo a los métodos de detección de fallas.

Capítulo 3

Métodos de detección de fallas

En este capítulo se resumen los principales trabajos relacionados con esta investigación. Inicialmente, se describe de manera general en que consiste el proceso de detección de fallas y posteriormente se describen los trabajos encontrados en la literatura científica que desarrollan métodos para la detección de fallas aplicados a diferentes escenarios, poniendo especial énfasis en el método propuesto en [Ibargüengoytia et al., 2005] sobre el cual se basa esta investigación.

3.1. Detección de fallas

La complejidad de la mayoría de los procesos industriales actuales requieren de altos grados de confiabilidad, lo que se traduce en la necesidad de sistemas de control que le aporten seguridad al proceso. Una de las tareas fundamentales de estos sistemas de control lo constituye la detección y el diagnóstico del comportamiento fallido. Los principales conceptos relativos a la detección de fallas se resumirán a continuación, las ideas expuestas se tomaron de [Isermann, 2006].

Una **falla** es una desviación no permitida de al menos una propiedad carac-

terística del sistema de su condición estándar, aceptable o usual [Isermann, 2006]. Las principales características de las fallas son:

- es un estado dentro del sistema,
- la desviación no permitida es la diferencia entre el valor fallido y el límite violado de una zona de tolerancia para sus valores usuales,
- es una condición anormal que puede causar una reducción o pérdida de la capacidad de una unidad funcional para realizar una función requerida,
- una falla en el sistema es independiente de si el resto del sistema está en funcionamiento o no,
- puede iniciar una avería (*failure*) o un mal funcionamiento,
- frecuentemente, las fallas son difíciles de detectar, especialmente si las desviaciones de los valores permitidos son muy cercanos a los umbrales o en aquellos casos donde el comportamiento fallido resulta intermitente.

Por otra parte una **avería** o *failure* es una interrupción permanente de la capacidad de un sistema para realizar una función requerida bajo condiciones de operación especificadas. Las principales características de las averías son:

- es la terminación de la capacidad de una unidad funcional para realizar una función requerida,
- es un evento,
- es el resultado de una o más fallas,
- se pueden distinguir diferentes tipos de averías: en dependencia del número de averías: simples o múltiples, o de acuerdo a que tan previsibles sean:

    - aleatoria (impredecible, como, por ejemplo, estadísticamente independiente del tiempo de operación u de otras averías),
    - determinística (predecible para ciertas condiciones),
    - sistemática o casual (dependiente de condiciones conocidas);
- generalmente surgen después del comienzo de la operación o al estresar demasiado el sistema.

Otro concepto importante en el proceso de detección de fallas es el de mal funcionamiento. Un **mal funcionamiento** es una irregularidad intermitente en el cumplimiento de la función deseada de un sistema. Sus principales características son:

- es una interrupción temporal de una función del sistema,
- es un evento,
- es el resultado de una o más fallas,
- generalmente surgen después del comienzo de la operación o al estresar demasiado el sistema.

La figura 3.1 muestra la relación entre fallas, averías y mal funcionamiento. Una falla puede ocurrir abruptamente. Se supone que la característica correspondiente del sistema relacionada con la falla es proporcional al desarrollo de la falla. Después de exceder la tolerancia de los valores normales, la función indica un error. Dependiendo de su tamaño, se produce una avería o un mal funcionamiento del sistema.

En resumen, en esta tesis nos proponemos la detección de fallas para minimizar el tiempo de mal funcionamiento de un sistema y así evitar la ocurrencia de las averías o mal funcionamiento.

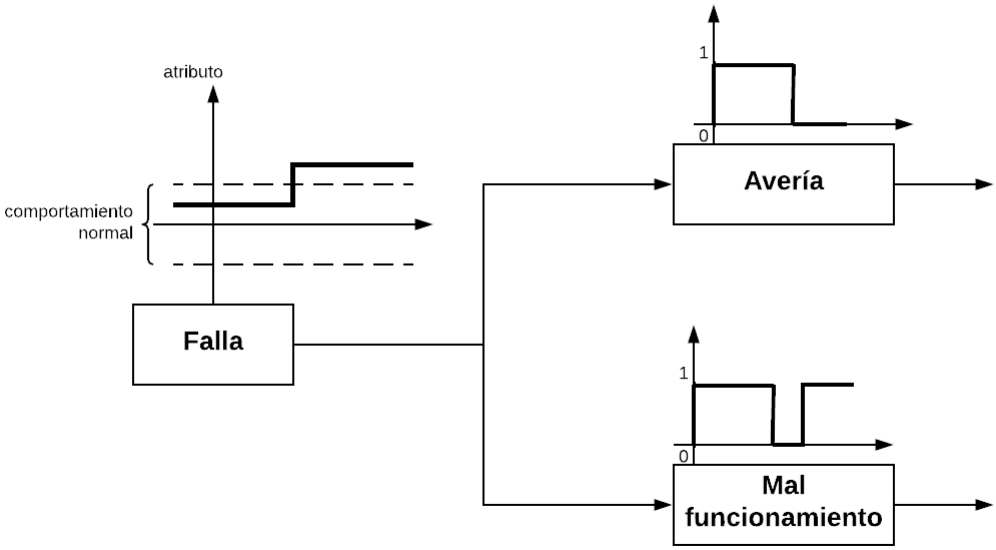

Figura 3.1: Relación entre fallas, averías y mal funcionamiento. Fuente [Isermann, 2006].

3.2. Métodos de Detección de Fallas

Esta sección resume los principales trabajos en el área de la detección de fallas. Para organizar la escritura seguimos la taxonomía propuesta por [Pires et al., 2016], incluyendo otros trabajos encontrados y algunos de los propuestos por los autores. La figura 3.2 representa la nueva taxonomía propuesta (la rama de la corrección de datos no se desarrolla pues no constituye objetivo de esta investigación).

El proceso de validación de los datos consiste de dos principales etapas: la detección de fallas en los datos y la corrección de estos datos fallidos. La detección de datos defectuosos identifica valores dudosos o errores en los datos y el proceso de corrección proporciona métodos para tratar datos problemáticos [Sun et al., 2011]. En cada categoría, existen en la literatura científica diferentes herramientas y métodos; nos centraremos en los métodos de detección de fallas, siendo el objetivo principal de nuestra investigación.

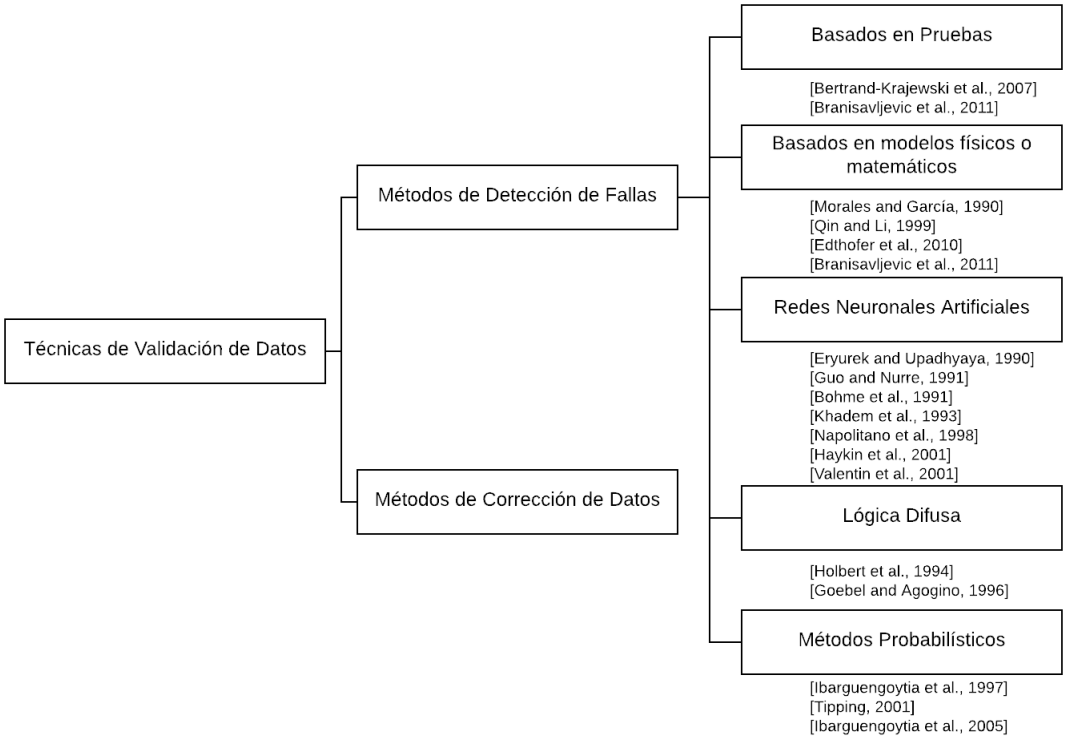

Figura 3.2: Taxonomía del estado del arte en el área de los métodos de detección de fallas.

Entre los trabajos encontrados en el estado del arte para la detección de fallas existen métodos simples basados en pruebas, clasificando los datos en válidos, inválidos o faltantes [Bertrand-Krajewski et al., 2007] [Branisavljević et al., 2011]. En [Bertrand-Krajewski et al., 2007] los autores realizan una investigación utilizando pruebas de laboratorio de la incertidumbre de diferentes métodos de análisis a partir de los datos del objeto de estudio. Por otro lado, en [Branisavljević et al., 2011] presentaron un orden lógico de métodos a aplicar para la obtención de mejores resultados en la detección de datos fallidos, entre ellos proponen: detección de valor cero, detección de línea plana, detección de valores mínimo y máximo, umbrales mínimo y máximo basado en los últimos valores, las pruebas estadísticas que siguen ciertas distribuciones, pruebas estadísticas multivariadas, redes neuronales artificia-

les (RNA) [Hassoun, 1995] , máquina de soporte vectorial (SVM) [Boser et al., 1992], y modelos físicos y de clasificación.

La principal limitación de estos trabajos, como su nombre lo indica, es que se basan en el análisis de diferentes pruebas sobre los datos por lo que requieren del conocimiento del fenómeno observado, además de que son dependientes del dominio de aplicación por lo que resulta complejo la generalización de estos métodos a diferentes dominios.

Los métodos basados en modelos físicos o matemáticos son estrategias que hacen uso de un modelo formulado a partir del conocimiento de las dinámicas involucradas en el proceso. Consisten en el análisis de las diferencias entre las salidas del proceso y un modelo del proceso, de donde se infiere la presencia de una falla. Este enfoque únicamente puede ser aplicado a procesos donde es posible obtener dicho modelo de forma analítica, lo que limita su aplicación en sistemas no lineales. La obtención de los parámetros del proceso también representa una gran dificultad. Incluyen verificación de valores extremos usando estadísticas, métodos de consistencia espacial, métodos de redundancia analítica y pruebas estadísticas multivariadas usando análisis de componentes principales (PCA). Entre los trabajos incluidos en esta clasificación destacan: [Morales and Garcia, 1990], [Qin and Li, 1999], [Edthofer et al., 2010], [Branisavljević et al., 2011].

En [Morales and Garcia, 1990] los autores proponen una estrategia modular para un sistema de diagnóstico de múltiples fallas basado en modelos físicos (formados por un conjunto de ecuaciones) y en el sistema de propagación de restricciones propuesto en [De Kleer and Williams, 1987]. Para ello proponen una variante basada en la creación de grupos de restricciones que reduce el costo computacional asociado al proceso al dividir en regiones el sistema.

En [Qin and Li, 1999] los autores proponen un método para la detección, identificación y reconstrucción de sensores usando un modelo del comportamiento

normal construido a partir de métodos estadísticos como los mínimos cuadrados parciales o el análisis de componentes principales. El modelo residual se usa para detectar fallas del sensor que demuestran una desviación de la operación normal.

Los trabajos presentados en [Edthofer et al., 2010] y [Branisavljević et al., 2011] constituyen también aplicaciones a diferentes dominios donde a partir del modelo matemático del proceso se analizan las desviaciones de los datos con respecto al modelo construido.

Las Redes Neuronales Artificiales han sido ampliamente utilizadas en tareas de identificación y diagnóstico de fallas dadas sus capacidades de aproximar cualquier función multivariada lineal o no lineal, a partir de datos. Dicha cualidad ha sido explotada en dos tendencias principales: (i) las técnicas que hacen uso de un modelo de regresión del proceso, para luego ser contrastado con el proceso real y obtener un residuo; (ii) y las técnicas que modelan directamente los datos de salida tanto para condiciones de operación normal como anormal.

Diferentes autores han investigado el uso de las redes neuronales para la detección de fallas. Algunas de las arquitecturas de redes utilizadas incluyen el uso de Perceptrones Multicapa para la estimación de algunas variables a partir de otras conocidas [Eryurek and Upadhyaya, 1990], [Guo and Nurre, 1991], [Khadem et al., 1993]. En otro trabajo, [Napolitano et al., 1998], utiliza Perceptrones Multicapa para estimar variables basadas en sus valores de épocas anteriores, para ello comparan dos enfoques para el proceso de detección de fallas. El primero se basa en el uso de un conjunto de Redes Neuronales de aprendizaje en línea y el segundo enfoque se basa en el uso de los filtros de Kalman [Haykin et al., 2001]. El estudio revela que las arquitecturas neuronales de aprendizaje en línea producen mejores resultados para propósitos de estimación en un esquema de validación de sensores.

Otra de las técnicas propuestas son los Mapas Auto-Organizativos, que pueden usarse para organizar los datos en grupos de datos relacionados para la detección

de fallas, [Böhme et al., 1991], [Valentin et al., 2001]. Las lecturas del sensor se pueden clasificar para pertenecer al grupo más cercano y la detección de un sensor defectuoso se basa en la distancia de una lectura del prototipo que representa el grupo más cercano.

Todos estos enfoques que se basan en el uso de Redes Neuronales tienen como principal limitación la necesidad de aprender el modelo a partir de la información completa de las variables del mismo, por lo que será necesario el uso de técnicas alternativas para predecir los valores de las variables en presencia de información incompleta o incertidumbre en los datos.

Otros estudios proponen el uso de la teoría de la Lógica Difusa. Los sistemas basados en Lógica Difusa han sido eficientemente aplicados en problemas de detección y diagnóstico de fallas. Una de las principales cualidades de dichos sistemas radica en la interpretabilidad del diagnóstico efectuado que se obtiene al observar las distintas reglas activadas en el proceso de inferencia. Las técnicas difusas han sido implementadas tanto en arquitecturas que hacen uso de modelos y residuos, como en estructuras que modelan y procesan directamente las entradas y salidas del proceso sin un modelo explícito.

En [Holbert et al., 1994] se introduce el uso de la lógica difusa para la validación de sensores redundantes, en este caso la lógica difusa se utiliza para transformar las abstracciones humanas en el dominio de los números reales y de esta forma los principios de análisis expresados lingüísticamente se pueden codificar en una base de reglas de clasificación para la detección e identificación de fallas de señal.

Por otra parte en [Goebel and Agogino, 1996], los autores proponen una arquitectura basada en lógica difusa para la validación y fusión de sensores en tiempo real en tareas de seguimiento de vehículos para autopistas automatizadas.

De manera general las principales limitaciones asociadas al uso de la Lógica Difusa están relacionadas con la definición de las funciones de membresía y la

construcción de las reglas de inferencia, que requieren del conocimiento experto.

Otra de las técnicas utilizadas para la validación y detección de fallas en los datos, y que son robustas a la incertidumbre lo constituyen los métodos probabilísticos. Los autores de [Ibarguengoytia et al., 1997] proponen algoritmos de validación de sensores que combinan diferentes métodos probabilísticos, entre ellos las Redes Bayesianas. Por el contrario, [Tipping, 2001] propuso la validación de datos utilizando *Sparse Bayesian Learning* y *Relevance Vector Machine (RVM)*, que constituye una especialización de SVM.

Nuestra investigación constituye una extensión del trabajo presentado en [Ibargüengoytia et al., 2005], que utiliza las Redes Bayesianas para la validación de sensores. La siguiente sección expone las generalidades de este método pues constituye el antecedente directo del método propuesto en esta tesis.

3.2.1. Validación de sensores usando Redes Bayesianas

En [Ibargüengoytia et al., 2005] los autores proponen un algoritmo de validación de sensores representando las relaciones entre las variables mediante el uso de una Red Bayesiana y a partir de la propagación probabilística estiman la presencia de una falla aparente a partir de la diferencia entre el valor estimado y el valor real. El método consiste de dos etapas fundamentales: la etapa de detección de fallas aparentes y la etapa de aislamiento de las fallas reales.

En la **etapa de detección**, las fallas aparentes se detectan comparando el valor actual con el predicho a través de la propagación de creencias, considerando como evidencia a las variables que pertenecen a la cobija de markov de la variable a analizar. Este proceso se repite para todas las variables, identificando aquellas cuyo valor difiere del valor predicho como fallas aparentes. El algoritmo 3.1 resume los pasos fundamentales de esta etapa.

Algoritmo 3.1 Método de detección de fallas aparentes propuesto por [Ibargüengoytia et al., 2005]

1: Obtener la Red Bayesiana de las variables del dominio a validar.
2: De la lista de variables a validar (usualmente todas) construir la tabla de cobijas de markov extendidas (EMB).
3: Tomar cada una de las variables a validar como hipótesis. Instanciar las variables que forman la cobija de markov de la hipótesis y propagar las probabilidades para obtener la distribución de probabilidad posterior de la variable dada la evidencia.
4: Comparar el valor predicho (máxima probabilidad posterior) con el valor actual de la variable y decidir si hay un error.
5: Repetir los pasos 3 y 4 hasta que se examinen las variables de la lista y se obtenga el conjunto de variables con fallas aparentes.

Relacionado con la **etapa de aislamiento de las fallas**, en [Ibargüengoytia et al., 2005] se demostró que a partir de la Cobija de Markov Extendida (EMB) de una variable, es posible aislar las fallas reales de un conjunto de fallas aparentes. Sea S el conjunto de fallas aparentes obtenidas de la etapa anterior

1. Si $S = \emptyset$ entonces no existen fallas reales.

2. Si S es igual al EMB de una variable X y no existe otro EMB que sea un subconjunto de S, entonces existe una falla real simple en X.

3. Si S es igual al EMB de una variable X y existen uno o más EMB que son subconjuntos de S, entonces existe una falla real en X y posiblemente fallas reales en variables cuyos EMB son subconjuntos de S.

4. Si S es igual a la unión de diferentes EMB y esta combinación es **única**, entonces existen múltiples fallas reales distinguibles en todas las variables cuyo EMB está en S.

5. Si ninguno de los casos anteriores se satisfacen, entonces existen múltiples fallas

pero son indistinguibles. Todas las variables cuyo EMB es un subconjunto de S pudieran ser fallas reales.

Teniendo en cuenta los antes descrito, en la etapa de aislamiento, de las fallas aparentes identificadas en la primera etapa y basadas en la propiedad de la Cobija de Markov (MB), se crea una red bayesiana adicional compuesta de dos niveles: los nodos raíz representan las fallas reales y los nodos en el nivel inferior representan las fallas aparentes. Los arcos entre los nodos de los dos niveles se construyen a partir de las relaciones de una variable con falla real y su Cobija de Markov Extendida. Las probabilidades a priori se le asignan distribuciones uniformes. Para obtener las probabilidades posteriores de las fallas reales se considera como evidencia las fallas aparentes obtenidas en la etapa anterior. La figura 3.3 representa un ejemplo sencillo de Red Bayesiana y su correspondiente Red de Aislamiento.

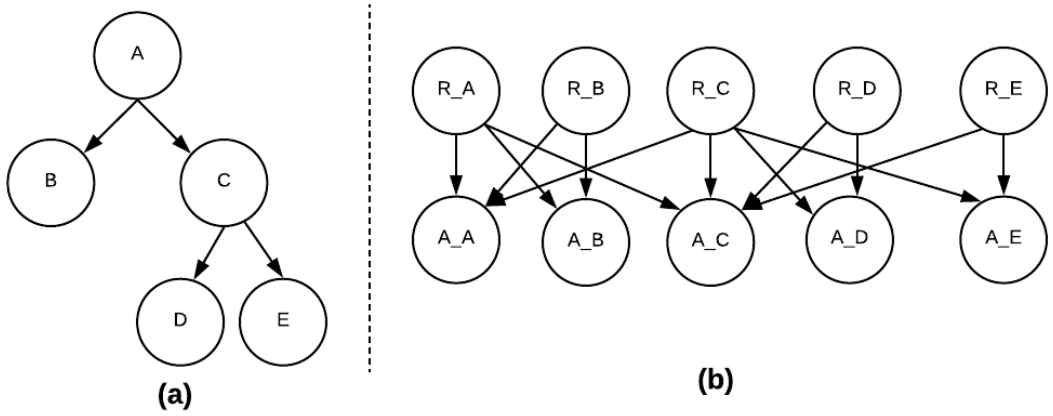

Figura 3.3: Ejemplo de construcción de Red de aislamiento. **(a)**Red Bayesiana **(b)** Red de aislamiento construida a partir del ejemplo descrito en (a); los nodos raíz representan las fallas reales y los nodos en el nivel inferior representan las fallas aparentes.

A partir del análisis del método propuesto en [Ibargüengoytia et al., 2005], las preguntas requeridas en esta etapa de la investigación serían: ¿por qué usar la técnica

de redes bayesianas para la detección de fallas? y ¿cuál es el hueco en el estado del arte para el desarrollo de esta investigación?

En la literatura científica consultada queda demostrada las ventajas del uso de las redes bayesianas sobre todo para aquellos problemas donde se requiere del manejo de la incertidumbre en los datos. En el caso de la detección de fallas en la mayoría de los casos los datos a analizar pueden estar incompletos, además de la necesidad en este tipo de problemas de sistemas que arrojen respuestas en el menor tiempo posible. Por otra parte, el uso de esta técnica no necesita de información de fallas anteriores para la detección en el modelo aprendido, solo de los datos en condición de "operación normal".

El hueco en el estado del arte que justifica esta investigación está relacionado con la complejidad de los sistemas, lo que se deriva en estructuras de redes bayesianas con topologías de redes con múltiples conexiones. Los trabajos que usan las redes bayesianas para la detección de fallas no toman en cuenta la complejidad de los modelos en el proceso de inferencia. En [Cooper, 1990], el autor demostró que el mecanismo de inferencia en las Redes Bayesianas con múltiples conexiones constituye un problema *NP-duro* por lo que al aumentar el dominio del problema (densidad de la red bayesiana) aumenta la complejidad en orden exponencial. Es por ello que en el trabajo presentado en [Ibargüengoytia et al., 2005], el dominio del problema a modelar constituye un aspecto importante tanto en el modelo probabilístico que se obtiene en la etapa de detección como en la red bayesiana que se construye en la etapa de aislamiento de las fallas.

En esta tesis nos proponemos solventar esta limitación a partir de extender el método presentado por [Ibargüengoytia et al., 2005] a un enfoque distribuido usando las Redes Bayesianas de Múltiples Secciones.

3.3. Resumen del capítulo

De manera general en este capítulo se resumen los principales trabajos relacionados con la detección de fallas. Inicialmente se definen los conceptos fundamentales relacionados con el proceso de detección de fallas. Relativos a los trabajos que desarrollan este tema, existen en la literatura científica varias propuestas que usan diferentes técnicas, destacando como más comunes aquellos que usan las Redes Neuronales Artificiales. Cada técnica presenta ventajas o limitaciones sobre las demás, dependiendo en gran medida de su aplicación. Finalmente se resumen las generalidades del método de detección de fallas basado en Redes Bayesianas propuesto en [Ibargüengoytia et al., 2005] sobre el cual se basa esta tesis.

En el siguiente capítulo se describen las generalidades del método propuesto en esta tesis que busca extender el trabajo presentado en [Ibargüengoytia et al., 2005] a un enfoque distribuido con el uso de las Redes Bayesianas de Múltiples Secciones para disminuir el tiempo de ejecución relacionado con el proceso de inferencia y que constituye una de las principales limitaciones del trabajo de [Ibargüengoytia et al., 2005].

Capítulo 4

Método de Detección de fallas distribuido

En este capítulo se presenta una descripción del método propuesto a partir de extender el trabajo presentado en [Ibargüengoytia et al., 2005] a un enfoque distribuido con el uso de la técnica de las Redes Bayesianas de Múltiples Secciones (MSBN). La figura 4.1 describe el proceso general de detección de fallas seguido para la implementación de nuestro método distribuido. Señalar que cada partición del dominio resulta en subdominios más pequeños en los que la intersección de los mismos lo constituyen las variables compartidas entre las secciones adyacentes.

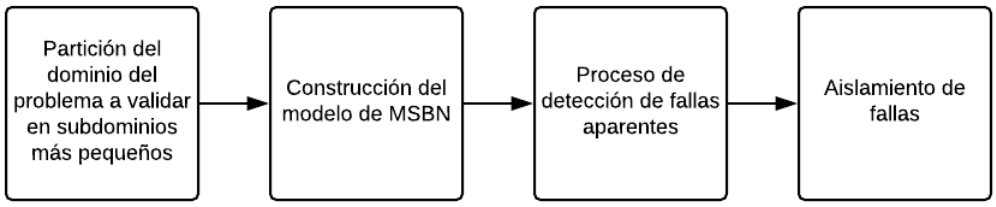

Figura 4.1: Descripción del proceso general de detección de fallas propuesto.

Para organizar el trabajo las siguientes secciones se dividen tomando en con-

sideración las etapas propuestas por los autores en [Ibargüengoytia et al., 2005]: la etapa de detección de fallas aparentes y la etapa de aislamiento de fallas.

4.1. Etapa de detección de fallas aparentes

En la etapa de detección de fallas aparentes y a partir de las capacidades de las Redes Bayesianas de Múltiples Secciones para la modelación e inferencia eficiente de problemas de dominio grande, se construye una MSBN para representar las relaciones entre las variables del dominio a validar, particionada a su vez en secciones que satisfacen las condiciones descritas en el marco teórico: condición de hyper-árbol, *d-sepset* y DAG con múltiples secciones con organización de hyper-árbol.

Para mostrar las etapas del algoritmo se muestra a continuación un ejemplo de circuito lógico combinacional que servirá de guía para ejemplificar cada uno de los pasos a seguir. La figura 4.2 muestra un ejemplo sencillo de circuito lógico compuesto por 5 componentes independientes que se comunican entre sí. Cada componente está formado a su vez por un conjunto de compuertas AND, OR y NOT.

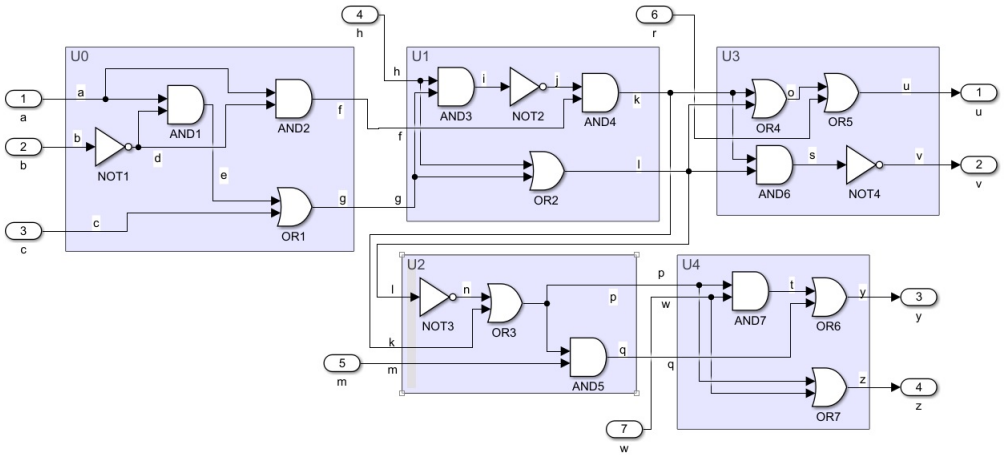

Figura 4.2: Ejemplo de circuito lógico combinacional particionado en 5 componentes.

La construcción de las Redes Bayesianas de Múltiples Secciones se realizó

apoyándose en la herramienta de investigación WebWeavr-IV [Xiang, 2006], desarrollada por el Dr. Y. Xiang para el trabajo con este tipo de redes y disponible para investigadores del área.

Las principales etapas para la construcción de una MSBN se resumen a continuación:

1. Construcción de la Red Bayesiana correspondiente a cada sección (nivel de agente individual). Para esta etapa los parámetros necesarios a definir son:

   - el conjunto de variables que corresponde con el subdominio a representar,
   - el grafo que representa las relaciones de independencia entre las variables, y
   - las distribuciones de probabilidad condicional de cada variable dado sus padres en el grafo.

   Para el ejemplo, cada componente constituye una sección individual. Las variables representan las señales de entrada y salida de cada compuerta que se relacionan a partir del diseño del circuito. La figura 4.3 representa las redes bayesianas obtenidas para cada componente. Las distribuciones de probabilidad condicional se obtienen a partir de la simulación del comportamiento normal del circuito.

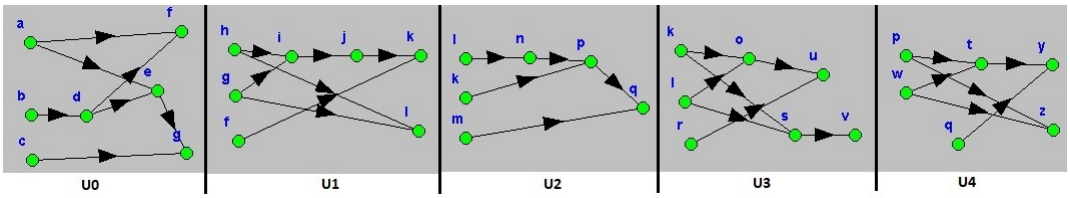

Figura 4.3: Redes Bayesianas correspondientes a cada compomente individual del circuito ejemplo de la figura 4.2.

2. Representación del conocimiento al nivel de sociedad entre los agentes. En esta etapa se define la estructura para la comunicación entre agentes adyacentes. Los principales pasos para esta etapa son:

   - Definición de la organización de las secciones o agentes a nivel global (hyper-árbol).
   - Definición de las variables públicas entre secciones adyacentes (*interface* entre las secciones).
   - Revisión de la condición de hyper-árbol, condición necesaria para garantizar la inferencia exacta. En esta estructura de hyper-árbol cada nodo se corresponde con una sección definida en el paso anterior y los enlaces se corresponden con las variables compartidas entre dos secciones adyacentes.
   - En esta etapa también se realiza la conexión entre las secciones a nivel físico para garantizar la comunicación, ya sea de manera local en una misma computadora o de manera distribuida, donde cada agente realiza su procesamiento en computadoras diferentes. Para realizar esta comunicación se construye un nuevo agente denominado *Binder* que es el encargado de realizar este enlace.

     Binder es una agente especial y su dirección física es conocida por todos los agentes. Este agente *Binder* tiene acceso a la organización entre los agentes y espera por que cada agente se registre. Para este registro cada agente envía su dirección IP y el puerto por el cual se realiza la comunicación.

     Luego de que todos los agentes se han registrado, *Binder* notifica a cada agente la dirección física de los agentes adyacentes a él así como las variables públicas compartidas entre ellos.

La figura 4.4 muestra la organización global entre las secciones del ejemplo de

la figura 4.2, en la parte inferior se resumen las variables compartidas entre secciones adyacentes. La estructura de árbol entre las secciones adyacentes garantiza la condición de hyper-árbol.

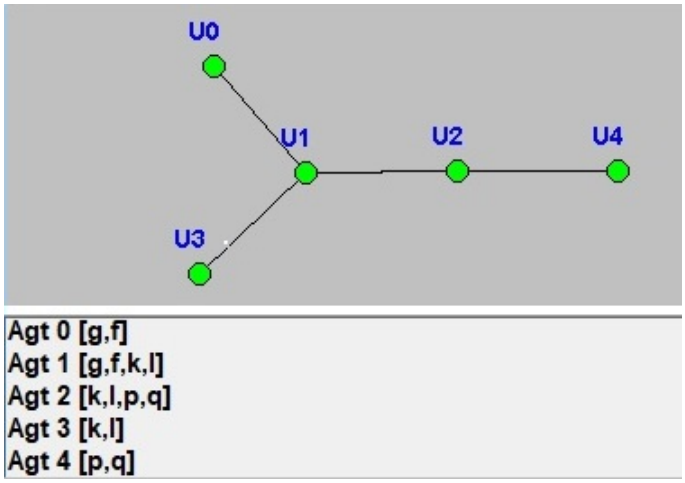

Figura 4.4: Organización a nivel de cooperación entre los agentes para el ejemplo de la figura 4.2. En el nivel inferior se muestran las variables públicas de cada sección (Agt 0 corresponde al componente U0, Agt 1 al componente U1, Agt 2 al componente U2, Agt 3 al componente U3 y Agt 4 al componente U4).

3. Verificación del modelo. En este paso se verifican el resto de las condiciones necesarias para garantizar la inferencia exacta. Esta etapa incluye la formación de los árboles de uniones (*junction tree*) asociados a cada sección (los pasos necesarios para la formación de los JT se resumieron en el capítulo 2). Las pruebas necesarias en este paso son:

   - La prueba de aciclicidad global. Como su nombre lo indica esta prueba verifica que no se formen ciclos entre las variables de las subredes de manera global. Cada subred es privada por los que la aciclicidad global no se puede verificar a partir de la fusión física de las subredes independendientes pues requiere de la divulgación de la estructura interna de cada subred. Teniendo en cuenta esta limitación, en [Xiang, 1998] los autores

propusieron un método de verificación que sólo requiere que cada agente pase mensajes a sus vecinos en el hyper-árbol sobre si su subred contiene algún padre o hijo de sus variables compartidas (pero no cuántos ni quienes son). Con base a estos mensajes los agentes pueden cooperar para detectar la aciclicidad global cada vez que ocurra.

- La condición de d-sepnodo para las variables compartidas. Para verificar esta condición, que requiere que para cada d-sepnodo sus padres pertenezcan a una misma sección, será necesaria también la cooperación entre los agentes. Para ello, en [Xiang and Chen, 2004] los autores desarrollaron un método que requiere solamente que un agente diga a sus vecinos, para cada una de sus variables públicas, si contiene algún padre de estas variables, pero no cuántas ni cuales son.

Para realizar esta verificación global cada agente realiza una copia de las pruebas de aciclicidad global y de d-sepnodo. Un agente seleccionado arbitrariamente inicializa la verificación. Durante este proceso los agentes transmitirán mensajes a lo largo del hyper-árbol, intercalando con el cómputo local en cada agente. Al final de la verificación, el agente que la inició anuncia si se ha pasado o no la verificación.

4. Compilación en Grupos de Árboles de Enlace. La inferencia en una MSBN consiste en la inferencia local de los agentes individuales y la comunicación entre los agentes. La inferencia local implica la actualización de la creencia de la subred (distribución de probabilidad conjunta) sobre su subdominio con base en las observaciones asociadas al subdominio.

   Durante la comunicación, la operación básica de un agente implica pasar a otro agente su distribución de probabilidad conjunta sobre su interfaz (el mensaje). Los dos cálculos están entrelazados: un mensaje de comunicación debe derivarse de la distribución local del agente remitente sobre su subdominio, y un

mensaje recibido debe procesarse para actualizar la distribución local sobre el subdominio del agente receptor.

Como ya se ha mencionado la inferencia local se realiza a través del método de JT. La estructura de **Árbol de enlace** es la responsable del paso de mensajes de manera eficiente entre las secciones o agentes adyacentes. Constituye un JT del *d-sepset* entre dos secciones adyacentes. Un árbol de enlace está compuesto sólo de las variables compartidas entre secciones adyacentes y se usa sólo para el paso de mesajes entre estas secciones. Se deriva del JT local y hereda todas las relaciones de independencia condicionales entre las variables compartidas.

Luego de realizado todo este proceso la MSBN puede actualizar de manera local la nueva evidencia con las correspondientes actualizaciones en las secciones adyacentes. Los pasos fundamentales para la construcción de la MSBN se listan en el algoritmo 4.1.

A partir del modelo construido, el próximo paso en el algoritmo sería la detección de las fallas a partir de nuevos datos considerados como nueva evidencia.

El proceso de detección de fallas aparentes consiste en la validación de manera local para cada sección de las variables pertenecientes a la misma a partir de la inferencia de cada variable por separado, considerando la cobija de Markov de la variable como evidencia. Si el valor real de la variable difiere del valor predicho dado cierto umbral, esta variable será considerada como una falla aparente. Los pasos fundamentales se resumen en el algoritmo 4.2.

Como se puede observar en el algoritmo 4.2, la salida de esta etapa será un conjunto de variables S que representan aquellas variables con fallas aparentes y que además pertenecen a una sección independiente o a un *d-sepset* (para el caso en el cual la falla aparente se encuentra en una variable compartida). Hay que destacar que cada agente realiza la validación de las variables que pertenecen a su subdominio de manera independiente, con las correspondientes actualizaciones por la estructura

Algoritmo 4.1 Construcción de la MSBN para la Detección de Fallas.

Entrada: Datos del modelo del dominio a construir y seccionamiento de los datos.

Salida: MSBN construida a partir de las secciones y de las estructuras de RB locales.

1: Construcción de las Redes Bayesianas para cada sección (estos modelos se pueden aprender a partir de los datos)

2: Representación de la organización entre los agentes o secciones y de las variables compartidas entre secciones adyacentes

3: Comunicación física entre los agentes a partir de la gestión de un agente especial $Binder$ que es el encargado de establecer esta comunicación

4: Construcción de los JT y de las tablas de probabilidad conjunta para cada sección

5: Verificación del modelo (aciclicidad global y *d-sepset*)

6: Compilación en grupos de árboles de enlace (árboles de uniones de las variables compartidas entre secciones adyacentes)

7: Inferencia multiagente. Dada una nueva evidencia se actualizan las redes bayesianas locales a las que corresponde la nueva evidencia y posteriormente se actualizan las secciones adyacentes de aquellas donde ocurren cambios.

Algoritmo 4.2 Detección de fallas aparentes

Entrada: Conjunto de datos del modelo a validar.

Salida: Conjunto S de fallas aparentes asociada a una sección.

1: Construcción de la MSBN que representa el dominio del problema a validar
2: **para** cada nueva evidencia **hacer**
3: **para** cada sección **hacer**
4: **para** cada variable a validar x_i (usalmente todas) **hacer**
5: Propagar las probabilidades para obtener las distribuciones de probabilidad posterior de la variable dada la nueva evidencia, considerando como evidencia los valores de las variables pertenecientes a su cobija de Markov. Esta propagación se realiza a través de la estructura de árbol de unión. Las probabilidades posteriores de cada variable se obtienen vía marginalización.
6: **si** $x_i \in d - sepset$ **entonces**
7: paso de mensajes por la estructura de árbol de enlace a sus secciones adyacentes con la actualización de la probabilidad posterior (siguiendo el método de actualización de creencias para poliárboles).
8: **fin si**
9: **si** $|valor\_predicho - valor\_actual| > umbral$ **entonces**
10: agregar x_i a S
11: **fin si**
12: **fin para**
13: **fin para**
14: **fin para**

del hyper-árbol.

4.2. Etapa de aislamiento de fallas

El objetivo principal de la etapa de aislamiento de las fallas es, a partir del conjunto de fallas aparentes obtenidas de la etapa anterior S, obtener un conjunto de variables $X, (X \subseteq S)$ que representa las fallas reales del sistema.

Teniendo en cuenta lo anterior, en la etapa de aislamiento de las fallas, se construyen nuevas redes bayesianas que toma ventaja de lo antes descrito. Cada red bayesiana representa una sección del modelo a validar. Estas nuevas redes bayesianas están compuestas de dos niveles. Los nodos en el primer nivel representan, para todas las variables del subdominio a modelar, los eventos con falla real y los nodos en el segundo nivel representan las fallas aparentes en todas las variables. La relación entre los dos niveles corresponde con la Cobija de Markov Extendida para cada variable. Las probabilidades a priori de cada variable del primer nivel (fallas reales R) se asignan siguiendo una distribución uniforme y las del segundo nivel (fallas aparentes A) se asignan siguiendo el esquema *noisy-or* donde $c_{ij} = P(A_j|R_i)$. Considerando como nueva evidencia las fallas aparentes identificadas en la etapa anterior ($Y \subseteq S$, siendo Y las variables que pertenecen al subdominio a modelar), se actualizan las probabilidades posteriores asociadas a las fallas reales de cada variable.

Para el caso donde se identifican fallas aparentes en variables compartidas, la red de aislamiento se forma a partir de la unión de las redes de aislamiento correspondientes a cada sección a la que pertenece la variable compartida. La figura 4.5 representa la red de aislamiento obtenida para el componente U2 del ejemplo de la figura 4.3.

El algoritmo 4.3 resume los pasos fundamentales de la etapa de aislamiento de fallas. Como se puede observar, a la salida de la etapa se tendrá una lista de variables

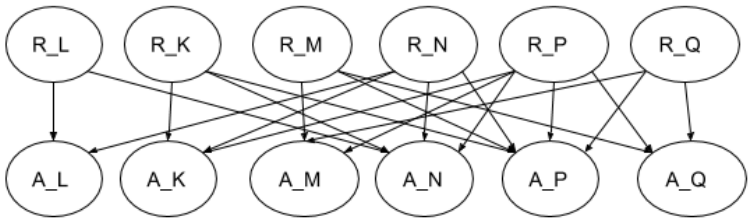

Figura 4.5: Red de aislamiento correspondiente al componente U2 del ejemplo de la figura 4.3.

ordenadas por la probabilidad de la ocurrencia de una falla real en ellas.

El *umbral* utilizado en el algoritmo 4.3 es un parámetro que indica la tolerancia para considerar una variable con falla real o no. Un análisis de los diferentes umbrales investigados se muestra en el siguiente capítulo.

4.3. Resumen del capítulo

En este capítulo se presentó un algoritmo novedoso para la detección de fallas. De manera general se toman la ideas propuestas por [Ibargüengoytia et al., 2005] y a partir de ellas se construyó un algoritmo distribuido para la detección de fallas, el cual solventa su principal limitación relacionada con la escalabilidad del método al aumentar el tamaño y la complejidad del modelo del dominio.

Dentro de los aspectos novedosos de nuestra investigación podemos destacar:

- El uso de las Redes Bayesianas de Múltiples Secciones para la detección de fallas aparentes.
- El procesamiento distribuido en la etapa de aislamiento de las fallas, tomando ventaja de las secciones del modelo.

Algoritmo 4.3 Aislamiento de las fallas reales

Entrada: Redes bayesianas de cada sección, conjunto S de fallas aparentes.

Salida: Conjunto X de fallas reales ordenadas por la probabilidad de que ocurra una falla real en ellas.

1: **para** cada sección o agente **hacer**
2: Construcción de las redes de aislamiento. R_i representa las variables con fallas reales y A_i las variables con fallas aparentes.
3: **fin para**
4: **si** existe en S variables compartidas **entonces**
5: Representar la red de aislamiento de la unión entre las secciones adyacentes cuya variable compartida es una falla aparente
6: **fin si**
7: **para** cada red de aislamiento **hacer**
8: Actualización de las probabilidades posteriores de R_i, tomando como evidencia $A_i \in S$.
9: **para** $x_i \in R_i$ **hacer**
10: **si** $valor\_predicho > umbral$ **entonces**
11: agregar x_i a X
12: **fin si**
13: **fin para**
14: **fin para**

En el siguiente capítulo se presentan los experimentos realizados con el algoritmo propuesto. Se muestra el desempeño del algoritmo aplicado a dos dominios diferentes: los circuitos lógicos combinacionales y las simulaciones de la Máquina Eólica Mexicana (MEM), además de su comparación con los resultados obtenidos con el método propuesto en [Ibargüengoytia et al., 2005].

Capítulo 5

Experimentos y resultados

En este capítulo se describen los experimentos sobre los cuales se realizaron las pruebas del método propuesto en esta tesis. Para ello, inicialmente se presentan las métricas utilizadas para evaluar los resultados obtenidos. Posteriormente el capítulo se divide de acuerdo a los dominios de aplicación sobre los cuales se realizaron los experimentos: los circuitos lógicos combinacionales y las simulaciones de la Máquina Eólica Mexicana. Dentro de cada sección se expone el diseño de los experimentos y los principales resultados obtenidos.

Cabe señalar que las pruebas se realizaron en una computadora con 32GB de memoria RAM, un procesador Intel Core i7 a 3.4GHz, sistema operativo Windows 10 y el lenguaje de programación JAVA, por lo que los resultados obtenidos en relación al tiempo de ejecución están sujetos a estas características.

El método propuesto requiere del análisis de dos parámetros diferentes, un primer parámetro asociado a la diferencia entre el valor real y el predicho ($p\_aparente$) en la etapa de detección de fallas y un segundo parámetro asociado a la probabilidad de considerar una falla real (p_real). Se realizó un análisis de los resultados variando los valores de ambos parámetros. Los mejores resultados se obtuvieron para

$p\_aparente = 0.01$ y $p\_real = 0.7$.

De manera general los experimentos realizados tienen dos objetivos fundamentales: comparar el método distribuido propuesto, con el método centralizado presentado en [Ibargüengoytia et al., 2005] en cuanto a la precisión de la detección de fallas y en cuanto al tiempo de ejecución.

5.1. Métricas de evaluación

Para la evaluación del método propuesto, se utilizaron adecuaciones de métricas existentes en el área de Recuperación de Información. Teniendo en cuenta que la salida del sistema constituye una lista de las variables del dominio del problema, ordenadas a partir de la probabilidad de que sean fallas reales. Para cada experimento se conocen las variables defectuosas que constituyen nuestras variables relevantes y deseamos conocer la eficacia de la recuperación de estas variables relativo a sus posiciones.

Considerando como precisión y recuerdo

$$precision = \frac{\#\_variables\_relevantes\_devueltas}{total\_variables\_devueltas} \tag{5.1}$$

$$recuerdo = \frac{\#\_variables\_relevantes\_devueltas}{total\_variables\_relevantes} \tag{5.2}$$

Se emplearon dos métricas de evaluación, que de cierta forma tienen en cuenta el orden de la recuperación, ellas son:

- P@ i. La precisión de nuestro sistema de devolver dentro de las i primeras variables aquellas con fallas reales, para el estudio se tomaron los valores de $i = 1, 3, 5$.

- *Mean Average Precision* (MAP). Esta es una métrica que toma en cuenta el orden en el que se devuelven la variables relevantes. Se define como:

$$MAP = AVG(\frac{\sum_{i=1}^{N} P(i) * rel(i)}{|variables\_relevantes|}) \tag{5.3}$$

donde N es el número de variables devueltas, $P(i)$ es la precisión de las primeras i variables y $rel(i)$ es una función binaria que indica si la variable en la i-ésima posición es relevante o no.

Cada una de las métricas persigue objetivos diferentes. $P@i$ busca evaluar la precisión de los métodos de devolver en las primeras i posiciones las variables con fallas reales y en el caso de MAP evaluar el orden en el que se devuelven las variables con fallas reales para un conjunto de consultas.

5.2. Experimentos con circuitos lógicos combinacionales

Los circuitos lógicos combinacionales se presentan como un ejemplo de sistemas complejos donde es posible particionar el dominio en subdominios más pequeños a partir de su distribución física en componentes relativamente independientes.

Las variables del dominio son las señales digitales por lo que constituyen variables discretas con dos posibles estados: 0 para los niveles de tensión eléctrica bajos y 1 para el caso contrario. La siguiente sección resume los pasos fundamentales seguidos para el diseño de los experimentos aplicados a este tipo de sistemas.

5.2.1. Diseño Experimental

Como parte de los experimentos realizados se tomaron como caso de estudio 3 ejemplos de circuitos lógicos combinacionales cuya complejidad va en aumento teniendo el cuenta la cantidad de compuertas lógicas involucradas y por tanto el

número de variables y de relaciones entre las mismas en su representación como redes bayesianas. La tabla 5.1 muestra un resumen de los casos de prueba.

Tabla 5.1: Resumen de las compuertas lógicas y las variables de cada ejemplo de prueba.

Ejemplo	compuertas OR	compuertas AND	compuertas NOT	Variables
1	8	6	4	25
2	20	24	8	97
3	75	96	44	391

Cada ejemplo de prueba está compuesto por diferentes componentes físicos que determinan el número de secciones en los cuales se particiona el dominio. El ejemplo 1 está compuesto por tres componentes independientes, el ejemplo 2 por cuatro y el ejemplo 3 por cinco componentes.

A continuación se detalla el proceso de construcción y evaluación del ejemplo 3 por ser el más complejo en cuanto al número de variables y secciones en las cuales se particiona el dominio.

Este ejemplo fue tomado de [Xiang, 2008] como un ejemplo de circuito digital compuesto por 5 componentes (U0, ..., U4). La tabla 5.2 muestra un resumen de las compuertas lógicas así como las variables particionadas para los diferentes componentes, que a su vez constituyen las secciones.

Para la construcción del modelo se simuló el comportamiento normal del circuito con la herramienta ModelSim [Graphics, 2015] y a partir de los datos obtenidos se construyó el modelo de MSBN. Como se ha mencionado en el capítulo anterior, el proceso de construcción de la MSBN implica la construcción de manera independiente de las redes bayesianas de cada sección así como la integración, comunicación física y verificación de las diferentes secciones en un sistema multi-agente.

Las representación de la redes bayesianas de cada sección se realizó a partir

Tabla 5.2: Resumen de las compuertas lógicas y las variables de cada sección para el ejemplo 3.

Componente	compuertas AND	compuertas OR	compuertas NOT	Variables
U0	15	12	6	59
U1	32	22	16	132
U2	12	10	4	46
U3	24	20	8	97
U4	13	11	10	57

del diseño del circuito, donde cada nodo representa una señal ya sea de entrada o de salida y las relaciones entre las variables se construyen a partir de las compuertas lógicas que las relacionan.

Las figuras 5.1 - 5.5 muestran el diseño de los componentes U0 - U4 del ejemplo, la figura 5.6 la red bayesiana correspondiente al componente U0 y la figura 5.7 la distribución de los agentes con las correspondientes variables compartidas entre las secciones adyacentes.

Luego de la construcción de las redes bayesianas locales y de la integración del sistema con la creación del agente especial *Binder* para establecer la comunicación física entre las secciones, así como de la verificación del modelo (prueba de aciclicidad global y *d-sepnode*) y de la inicialización de las creencias con la correspondiente actualización de las creencias a través de los árboles de enlace, ya el sistema está en condiciones de responder consultas de manera local asociadas a cada uno de los componentes.

Para las pruebas con el algoritmo propuesto se simuló el comportamiento fallido de uno o más componentes del circuito. Para ello se modificó el diseño del circuito para simular tres casos de fallas: compuertas pegadas a 1, compuertas pegadas a 0 y la negación de las salidas de las compuertas.

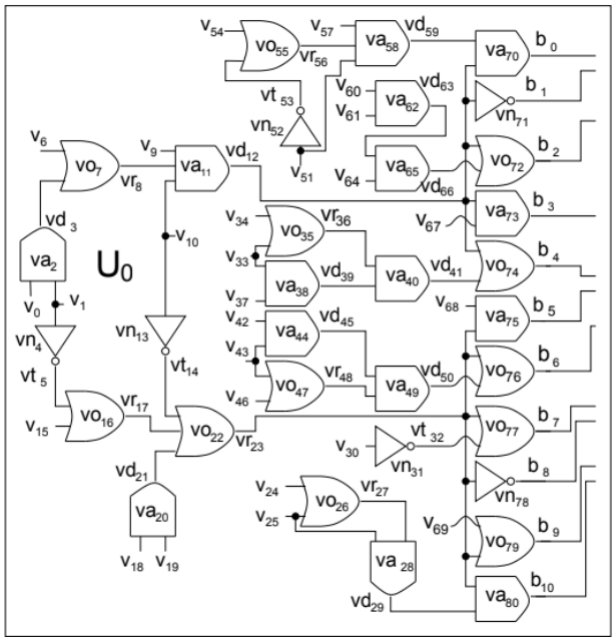

Figura 5.1: Diseño del circuito lógico del componente U0 del ejemplo 3.

Para cada ejemplo se generaron 20 casos de fallas, el 50 % de fallas simples (sólo falla una compuerta dentro del circuito) y el 50 % de fallas múltiples (las pruebas se realizaron con 2 y 3 fallas múltiples). Cada caso de prueba corresponde con el comportamiento anormal de uno o más componentes del circuito y está compuesto por 100 instancias. La selección del componente donde ocurre la falla así como el tipo de falla se realizó de manera aleatoria.

El objetivo fundamental de estos experimentos resulta es comparar tanto el método centralizado basado en RB como el método distribuido propuesto en cuanto a precisión y tiempo de ejecución en la detección de fallas en los circuitos lógicos combinacionales diseñados como casos de estudio.

De manera general se espera que el método distribuido propuesto tenga un comportamiento similar en cuanto a precisión con respecto al método centralizado pero con una reducción significativa del tiempo de ejecución asociada al proceso de detección.

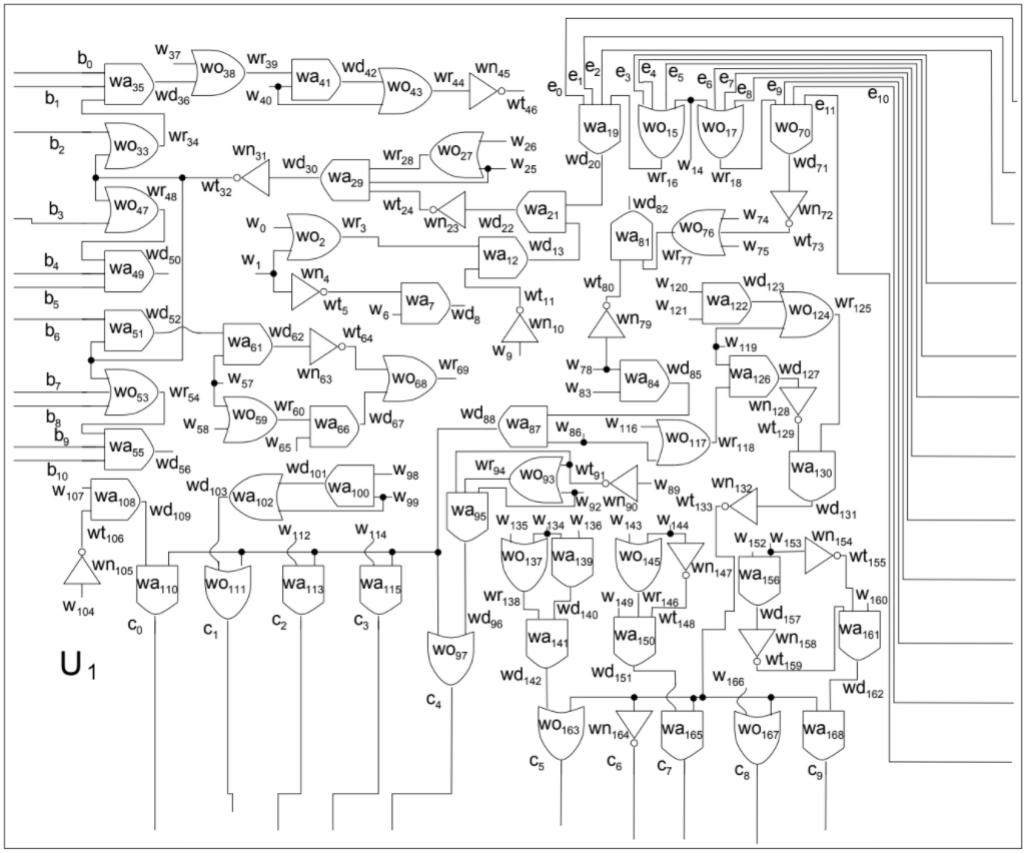

Figura 5.2: Diseño del circuito lógico del componente U1 del ejemplo 3.

5.2.2. Resumen de los resultados con los circuitos lógicos combinacionales

Los resultados obtenidos se muestran comparando la efectividad de nuestra propuesta (que llamaremos método distribuido) con una implementación del trabajo presentado en [Ibargüengoytia et al., 2005], que llamaremos método centralizado (recordar que este método realiza de manera centralizada todo el proceso de detección de fallas).

Los primeros experimentos realizados buscan comparar nuestro método con el método centralizado en cuanto a la precisión de ambos trabajos para la detección de fallas de manera general. La tabla 5.3 muestra los resultados en cuanto a P@3,

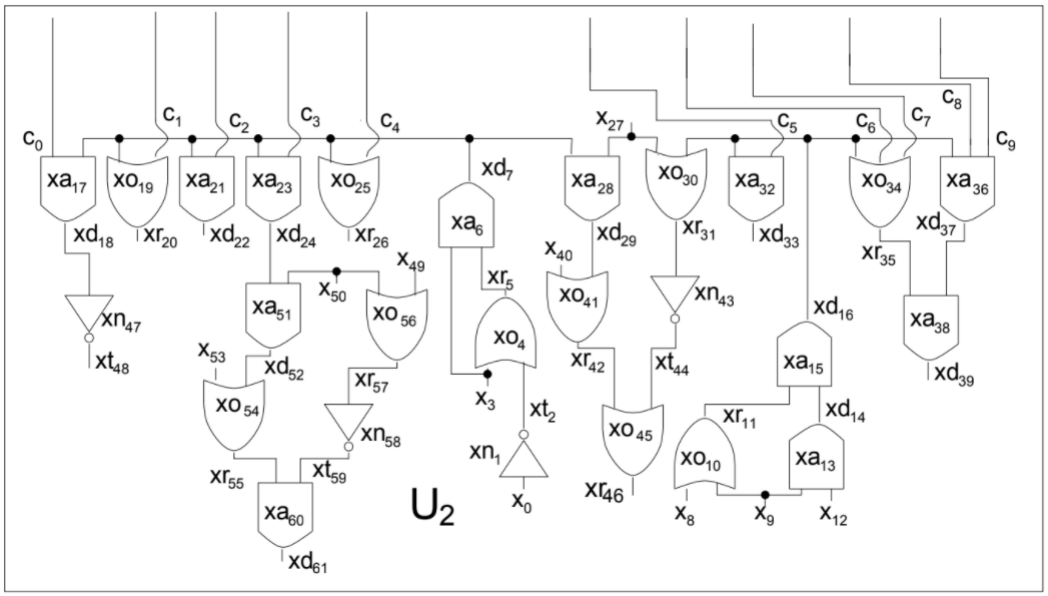

Figura 5.3: Diseño del circuito lógico del componente U2 del ejemplo 3.

P@5 y MAP de los tres ejemplos antes mencionados.

Como se muestra en la tabla para los ejemplos 1 y 2, los resultados son los mismos, lo cual tiene sentido porque la diferencia principal de nuestra propuesta con el trabajo presentado en [Ibargüengoytia et al., 2005] es la representación y la forma de hacer la inferencia, que se traduce en la reducción del tiempo de ejecución.

Para el ejemplo 3, dada la complejidad del problema y las capacidades de cómputo disponibles, no fue posible obtener una solución para el caso centralizado. Realizando los experimentos con el máximo de la capacidad de memoria del equipo de cómputo utilizado, luego de aproximadamente 252 horas en promedio se interrumpe el procesamiento. Este comportamiento de cierta forma justifica la investigación realizada.

De manera general, para ambos métodos, los mejores resultados se obtienen para P@5 lo que significa que los métodos encuentran en el 92 % de los casos en promedio las variables con fallas dentro de las 5 primeras posiciones. Los valores del

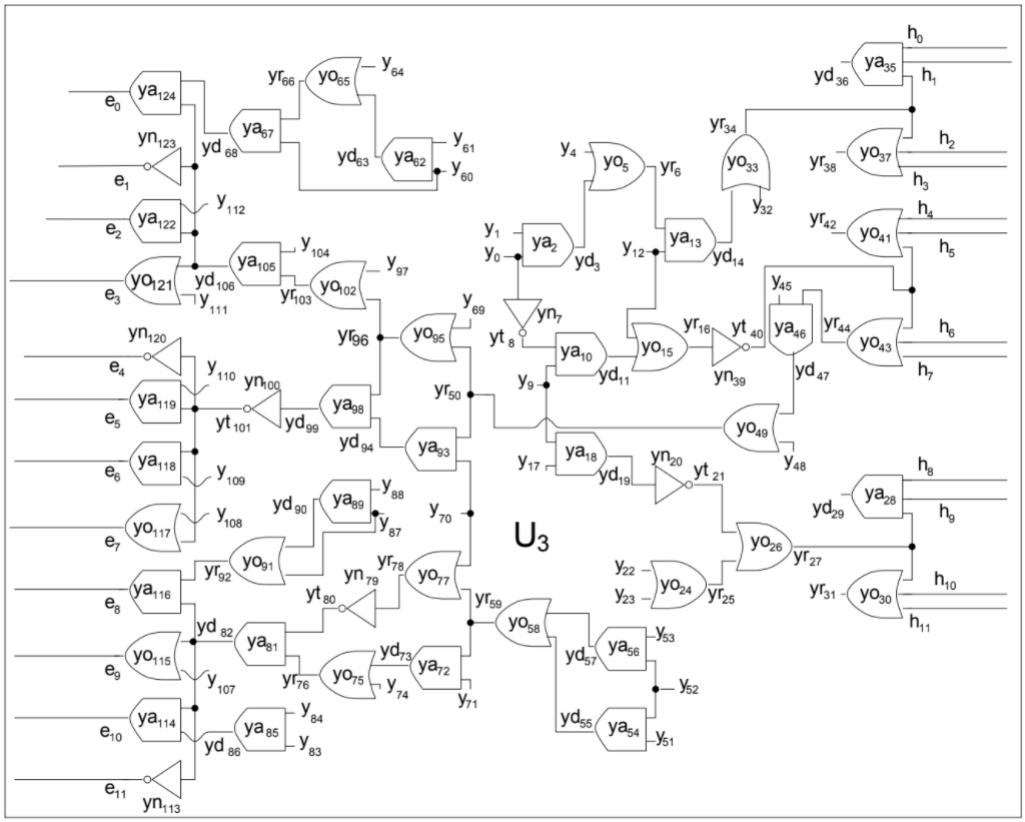

Figura 5.4: Diseño del circuito lógico del componente U3 del ejemplo 3.

MAP indican que a pesar de devolver en las primeras posiciones las variables con fallas reales, no siempre estas variables se encuentran en las primeras posiciones de la lista de variables.

Se incluye además entre los resultados obtenidos el análisis de la precisión del método propuesto separado de acuerdo a la detección de fallas simples o múltiples pues es conocido que el proceso de detección de fallas múltiples resulta más complejo que el de fallas simples.

Las tablas de la 5.4 a la 5.6 muestran los resultados, para las diferentes métricas de evaluación, de la precisión para la detección de fallas simples y fallas múltiples para cada uno de los ejemplos. Como se puede ver, los mejores resultados se obtienen para la detección de fallas simples, donde en el 96 % de los casos la variable con

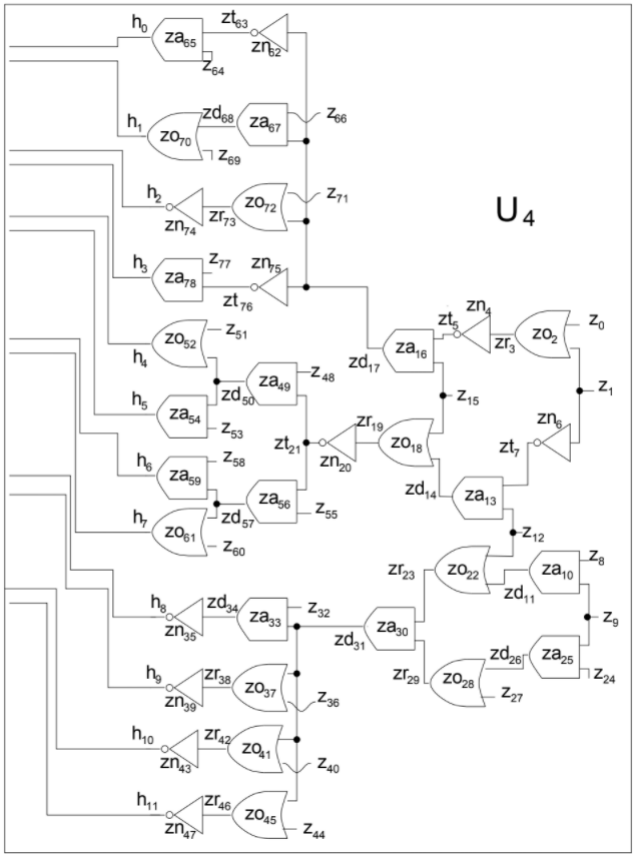

Figura 5.5: Diseño del circuito lógico del componente U4 del ejemplo 3.

falla se devuelve dentro de las 5 primeras posiciones de las variables con la mayor probabilidad de tener una falla real.

Sin embargo, los resultados de P@1 muestran que solo en el 30 % de los casos para los ejemplos de fallas simples, el método propuesto no devuelve como variable con mayor probabilidad (primera posición de la lista ordenada de variables) aquella que se corresponde con la falla real, lo que explica los resultados del MAP, pues esta métrica sí tiene en cuenta el orden en el cual se devuelven las variables relevantes.

Para el caso particular del ejemplo 3, los resultados del MAP son mejores para las fallas múltiples, lo que significa que a pesar de no devolver para todos los casos las fallas reales dentro de las 5 primeras posiciones (lo que si sucede en las fallas

Tabla 5.3: Comparación de nuestra propuesta con el trabajo presentado en [Ibargüengoytia et al., 2005] (método centralizado) en cuanto a P@3, P@5 y MAP para 3 ejemplos de circuitos.

Ejemplo	1			2			3		
	P@3	P@5	MAP	P@3	P@5	MAP	P@3	P@5	MAP
Método centralizado	0.75	0.9417	0.5788	0.65	0.8917	0.6321	-	-	-
Método distribuido	0.75	0.9417	0.5788	0.65	0.8917	0.6321	0.6667	0.9333	0.5171

Tabla 5.4: Comparación en términos de P@1, P@3, P@5 y MAP entre los casos de fallas simples y los de fallas múltiples para el ejemplo 1.

Métrica de evaluación	P@1	P@3	P@5	MAP
Fallas simples	0.3	0.8	1	0.575
Fallas múltiples	-	0.6	0.8833	0.5826

Tabla 5.5: Comparación en términos de P@1, P@3, P@5 y MAP entre los casos de fallas simples y los de fallas múltiples para el ejemplo 2.

Métrica de evaluación	P@1	P@3	P@5	MAP
Fallas simples	0.4	0.7	0.9	0.6
Fallas múltiples	-	0.6	0.8833	0.6643

Tabla 5.6: Comparación en términos de P@1, P@3, P@5 y MAP entre los casos de fallas simples y los de fallas múltiples para el ejemplo 3.

Métrica de evaluación	P@1	P@3	P@5	MAP
Fallas simples	0.2	0.8	1	0.495
Fallas múltiples	-	0.5533	0.8667	0.5393

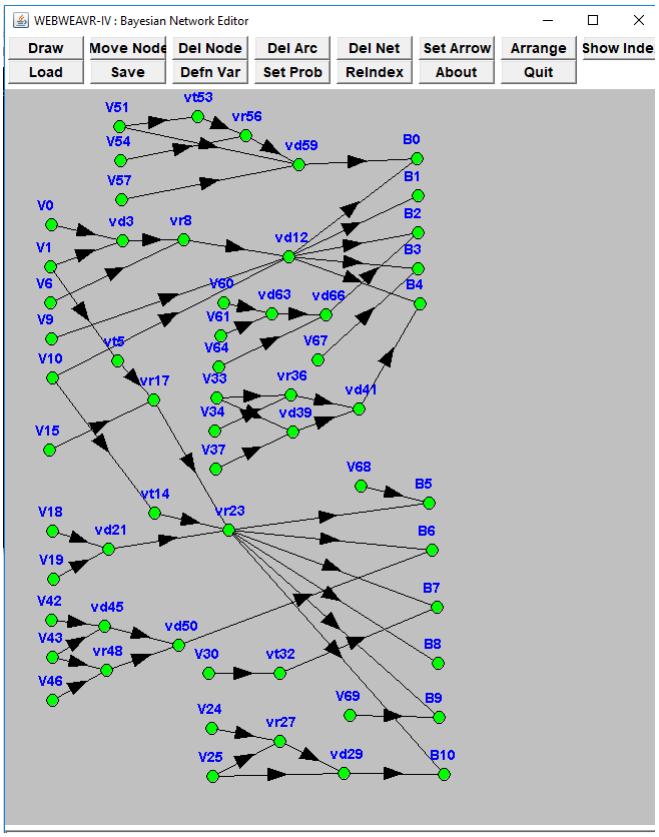

Figura 5.6: Red Bayesiana correspondiente al componente U0 del ejemplo 3.

simples), el orden en el que se devuelven los casos de prueba para las fallas múltiples es mejor.

Este comportamiento está en correspondencia con los principales resultados reportados en el estado del arte, donde los mejores resultados se obtienen para aquellos casos de falla simple.

La Tabla 5.7 muestra la comparación en términos de tiempo de ejecución entre el trabajo presentado en [Ibargüengoytia et al., 2005] y nuestra propuesta de todo el proceso de ejecución. Los tiempos se indican en minutos. Este análisis incluye el aprendizaje de los parámetros (este paso se realiza una sola vez para cada ejemplo), la etapa de detección de fallas y la etapa de aislamiento. Señalar que tanto la etapa de detección como la etapa de aislamiento se realizaron tantas veces como casos de

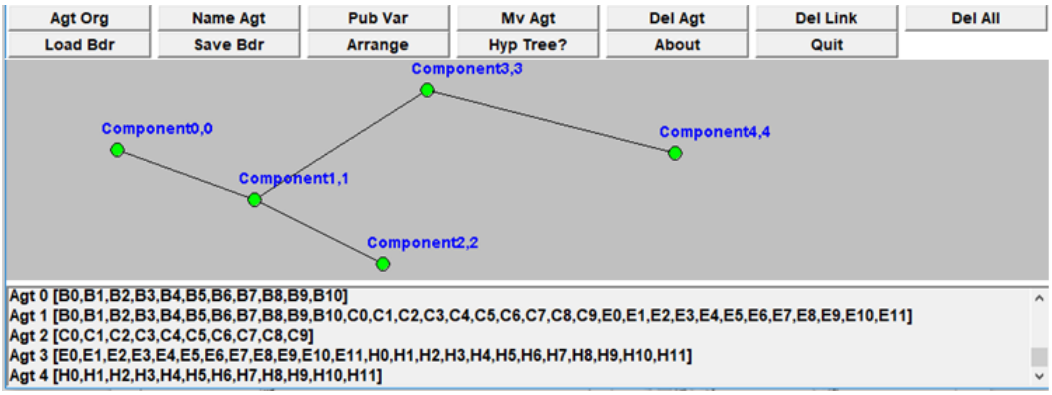

Figura 5.7: Distribución de los agentes a nivel global y variables compartidas del ejemplo 3.

fallas se modelaron para cada caso de prueba, en este sentido serían 20 casos de fallas por ejemplo, por lo que el promedio del tiempo de ejecución está en correspondencia con estos casos de fallas. Como puede verse, a medida que aumenta la complejidad del problema aumenta el tiempo de ejecución.

La reducción del tiempo del algoritmo propuesto es considerable y a medida que aumenta la complejidad del problema, la diferencia se hace más evidente. En el ejemplo 3, dado el tamaño del problema, no es posible obtener una solución con el método centralizado, por lo que no se realiza la comparación. Para este ejemplo el tiempo asociado a todo el proceso de detección de fallas con el método propuesto está en el orden de los días, lo que puede interpretarse como que tampoco es muy escalable. Sin embargo, cabe destacar que este tiempo está asociado a la máquina de cómputo empleada, además de que todo el procesamiento se realiza en la misma computadora; el algoritmo propuesto es fácilmente paralelizable.

La tabla 5.8 muestra el promedio del tiempo de ejecución asociado a cada una de las etapas para el ejemplo 3. Como se puede observar la mayor cantidad de tiempo está asociado al aprendizaje de los parámetros (incluye la representación de las redes bayesianas para cada sección, la comunicación entre las secciones adyacentes, la verificación del modelo y la inicialización de las creencias); una vez aprendido el

Tabla 5.7: Comparación en cuanto al promedio del tiempo de ejecución de [Ibargüengoytia et al., 2005] (centralizado) vs nuestra propuesta. Los tiempos se indican en minutos.

Ejemplo	1	2	3
Método Centralizado	0.1173	23.717	-
Método Distribuido	0.0325	9.07	4368.9

modelo la reducción del tiempo asociado al proceso de detección y aislamiento de las fallas es significativo. Este aspecto es importante, ya que la etapa de aprendizaje sólo se realiza una vez.

Tabla 5.8: Comparación en cuanto al promedio del tiempo de ejecución de cada etapa del algoritmo para el ejemplo 3. Los tiempos se indican en minutos.

Etapas del algoritmo	Aprendizaje de los parámetros	Etapa de detección	Etapa de aislamiento
Promedio del tiempo de ejecución	4350	11.3	7.6

De manera general los experimentos muestran que nuestra propuesta tiene un comportamiento similar en cuanto a precisión con el trabajo presentado en [Ibargüengoytia et al., 2005], ambos mostrando mejores resultados para aquellos casos donde ocurren fallas simples.

En cuanto al tiempo de ejecución nuestra propuesta es superior en comparación con el método de referencia pues la reducción a medida de que aumenta el tamaño del problema es significativa. Incluso se observa que para el caso más complejo, el método centralizado, con las capacidades de cómputo con las cuales se realizaron las pruebas, no es capaz de obtener una solución.

Para realizar un análisis más a fondo de los principales resultados obtenidos, se realizaron las pruebas de significancia estadística. Para ello y a partir de nuestra

hipótesis de investigación se realizaron dos pruebas, una para ver el comportamiento de las observaciones de acuerdo a la precisión y otro de acuerdo al tiempo de ejecución. La significancia estadística que se seleccionó fue del 5 % y la prueba de significancia seleccionada de acuerdo a las características de las observaciones (40 observaciones para el método centralizado y 60 para el método distribuido) fue la prueba de Mann-Whitney.

Para el caso de la precisión la significación asintótica obtenida fue $p = 0.774$ por lo que no se rechaza la hipótesis nula y por tanto no existen diferencias significativas entre los dos métodos.

Para el caso del análisis del tiempo de ejecución la significación asintótica obtenida fue $p = 0.000$ por lo que se acepta la hipótesis alternativa y por tanto existen diferencias significativas entre los dos métodos en cuanto al tiempo de ejecución.

Además se realizó un análisis del poder estadístico de la selección de la cantidad de fallas. De acuerdo a los resultados obtenidos para el método centralizado conocemos que la eficacia del mismo para estos ejemplos está en torno al 93 % y se espera que, en cuanto a la precisión, con el método distribuido se mantenga igualmente sobre el 93 %. Tras finalizar el estudio tenemos 40 observaciones de cada método. Al realizar el análisis estadístico, se objetivó que no hay diferencias significativas en la efectividad de ambos métodos de acuerdo a la precisión. Podemos calcular cuál ha sido finalmente el poder del estudio. Aplicando la fórmula para el cálculo del poder estadístico de comparación de dos proporciones ante un planteamiento unilateral se obtiene una $Z_{1-\beta} = 1.645$ lo que se corresponde con un poder estadístico del 95 %.

5.3. Experimentos con simulaciones de turbinas eólicas

En esta sección, como parte de la incorporación de este trabajo al proyecto P-12 de CEMIE-Eólico [CEMIE, 2014], se muestran los resultados del método propuesto

en el dominio de diagnóstico de fallas para turbinas eólicas. Para esto se utilizó la herramienta de diseño de turbinas eólicas FOCUS que permite la simulación de diferentes comportamientos.

Para ello en la siguiente sección se describen las generalidades de las turbinas eólicas así como del simulador FOCUS6 sobre el cual se realizaron las simulaciones de la Máquina Eólica Mexicana (MEM). Posteriormente se resume el diseño de los experimentos y por último los principales resultados obtenidos.

5.3.1. Generalidades de las Turbinas Eólicas y del simulador FOCUS

Una de las fuentes renovables de energía de más rápido crecimiento en los últimos años lo constituye la energía eólica que al mismo tiempo ha tenido una participación notable en los mercados de energía. Las turbinas eólicas son dispositivos que, mediante su rotación, convierten la energía cinética del viento en energía mecánica. La mayoría de las turbinas eólicas son unidades de 3 aspas cuyos componentes principales se muestran en la figura 5.8.

Impulsados por la fuerza del viento, las **aspas** de la turbina y el **rotor** convierten la energía eólica en energía mecánica, a través de la **caja de engranajes** hasta el **generador**. El eje principal soportado por los **rodamientos** (*bearings*) y la caja de engranajes tiene la función de optimizar la velocidad del generador para la generación de electricidad [Liu et al., 2015]. La alineación de la turbina con la dirección del viento está controlada por un **sistema de guiñada** y la **góndola** está montada en la parte superior de la **torre**.

Algunas de las fallas de las turbinas pueden ser detectadas a partir de la inspección visual de la misma, por ejemplo la decoloración de los componentes puede indicar variaciones de temperatura o condiciones de deterioro; sin embargo, la detección de la mayoría de las fallas requieren del monitoreo de los componentes de la

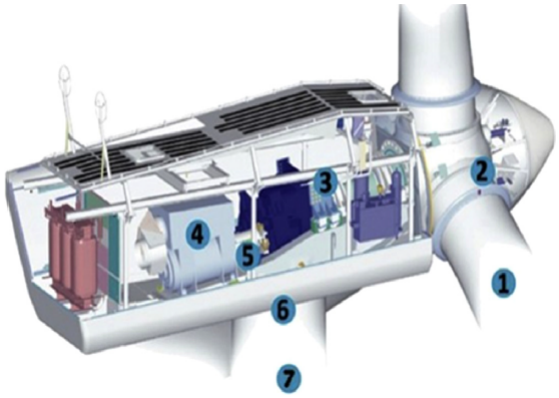

Figura 5.8: Componentes principales de una turbina eólica. (1) aspas, (2) rotor, (3) caja de engranajes, (4) generador, (5) rodamientos, (6) sistema de guiñada y (7) torre. Imagen tomada de [Liu et al., 2015].

misma. Es por ello que el objetivo de nuestro trabajo es la detección de algunas de las fallas típicas de las turbinas a partir del monitoreo de las variables involucradas en el modelo.

Para la simulación del comportamiento de una turbina, y a partir de la colaboración con el Instituto Nacional de Electricidad y Energías Limpias (INEEL) nos apoyamos en la herramienta de diseño de turbinas eólicas **FOCUS** para la simulación de diferentes casos de carga de diseño (DLC), a partir de variaciones tanto en las condiciones de viento como de los componentes de la turbina. Con el uso de la herramienta FOCUS y a partir del diseño de la Máquina Eólica Mexicana (MEM), rotor aerodinámico de eje horizontal de 3 aspas, desarrollada en el INEEL, realizaremos las simulaciones para diferentes casos de carga de diseño.

Resulta importante señalar que esta herramienta permite además la simulación de diferentes escenarios transitorios originados por alguna falla en el sistema mientras el aerogenerador produce electricidad. Dentro de las fallas posibles de simular en FOCUS se encuentran:

1. Operación normal con paso (*pitch*) fuera de control del aspa 2.

2. Operación normal con aspa 2 bloqueda.

3. Operación normal con cortocircuito en el aerogenerador.

4. Operación normal con falla en el controlador de paso.

5. Operación normal con sistema de guiñada fuera de control (positivo).

6. Operación normal con sistema de guiñada fuera de control (negativo).

7. Operación normal con desconexión de la red eléctrica.

8. Operación normal con falla de control que no conducen a paro. Dentro de este tipo de falla se destacan tres estados: operación con falla en el sistema de guiñada, demora en el paso del aspa 2 y sobrevelocidad.

De manera genaral, con el uso de la herramienta FOCUS y a partir de la implementación de la Máquina Eólica Mexicana desarrollada en el INEEL, se obtienen los datos del comportamiento normal de la turbina para diferentes condiciones, así como datos del comportamiento para diferentes casos de fallas. En la siguiente sección se resume el diseño de los experimentos realizados.

5.3.2. Diseño Experimental

El objetivo fundamental de los experimentos en las simulaciones de la MEM sería igualmente que en el caso de los circuitos lógicos combinacionales la comparación en cuanto a precisión y tiempo de ejecución del método centralizado basado en RB con el método distribuido, aplicados a otro dominio donde las variables son continuas y por tanto se requiere de esfuerzos adicionales como la discretización de las variables.

De manera similar a los experimentos en circuitos lógicos digitales, se esperan resultados similares en cuanto a precisión de ambos métodos con una reducción significativa del tiempo de ejecución para el caso del método distribuido propuesto.

La descripción general de los experimentos realizados se detallan a continuación.

El primer paso para realizar los experimentos con las simulaciones de los casos de carga de diseño es la obtención de los datos para diferentes casos. Se seleccionaron 5 casos de carga de diseño de producción normal implementados en el simulador. La selección de estos casos de carga de diseño se corresponde con algunos de los comportamientos normales más comunes de acuerdo a la experiencia del experto consultado. La tabla 5.9 resume cada uno de estos DLC.

Cada uno de los DLC de comportamiento normal, tiene asociado algunas de las fallas antes descritas (para un caso de carga no se tienen implementados todos los tipos de falla). La tabla 5.10 resume para cada DLC de comportamiento normal, los DLC de comportamiento con falla que le corresponde.

Cada una de las simulaciones tiene como salida 95 variables continuas que será necesario discretizar para la construcción del modelo de MSBN. Para el proceso de discretización y en base a la experiencia de un experto se utilizó la discretización no supervisada de ancho equitativo en 10 intervalos.

Para la construcción del modelo de MSBN, se crearon 6 secciones que intentan agrupar las variables de acuerdo a su localización física dentro de la turbina, las secciones construidas así como la cantidad de variables dentro de cada sección se listan en la tabla 5.11. Para cada sección se aprendió el modelo de red bayesiana por separado. La *interface* entre secciones adyacentes, al no contar con el conocimiento experto para la construcción de la misma, se obtuvo a partir de las relaciones del modelo centralizado de Red Bayesiana. La estructura de hyper-árbol obtenida se muestra en la figura 5.9.

Tabla 5.9: Descripción de los casos de carga de diseño de comportamiento normal.

Caso de carga de diseño (DLC)	**Descripción del DLC**
01300	Operación normal en condición extremo con viento de 13 m/s, ángulo de orientación de -8^{o} y ángulo azimutal 0^{o}.
12150	Operación normal en condición de fatiga con viento de 15 m/s, ángulo de orientación de -8^{o} y ángulo azimutal 0^{o}.
12110	Operación normal en condición de fatiga con viento de 11 m/s, ángulo de orientación de -8^{o} y ángulo azimutal 0^{o}.
12200	Operación normal en condición de fatiga con viento de 20 m/s, ángulo de orientación de -8^{o} y ángulo azimutal 0^{o}.
12131	Operación normal en condición de fatiga con viento de 13 m/s, ángulo de orientación de 8^{o} y ángulo azimutal 180^{o}.

Tabla 5.10: Casos de carga con falla implementados para cada uno de los DLC de comportamiento normal descritos en la tabla 5.9. Los números indican los tipos de fallas: **(1)** paso fuera de control del aspa 2, **(2)** aspa 2 bloqueda, **(3)** cortocircuito en el aerogenerador, **(4)** falla en el controlador de paso, **(5)** sistema de guiñada fuera de control (positivo), **(6)** sistema de guiñada fuera de control (negativo), **(7)** desconexión de la red eléctrica, **(8)** falla en el sistema de guiñada, **(9)** demora en el paso del aspa 2 y **(10)** sobrevelocidad.

DLC Producción Normal	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
01300	21140	21240	22140	-	22440	-	23400	-	24240	24340
12150	21160	21260	22160	-	22460	-	23600	-	24260	24360
12110	21120	21220	22120	-	-	-	23200	-	24220	24320
12200	-	-	-	-	22470	-	-	-	-	-
12131	-	-	-	-	-	22546	23414	-	-	-

Tabla 5.11: Cantidad de variables para cada sección del modelo de turbina.

	Aspa 1	**Aspa 2**	**Aspa 3**	**Torre**	**Aerogenerador**	**Rotor**
Cantidad de variables	14	14	14	18	23	12

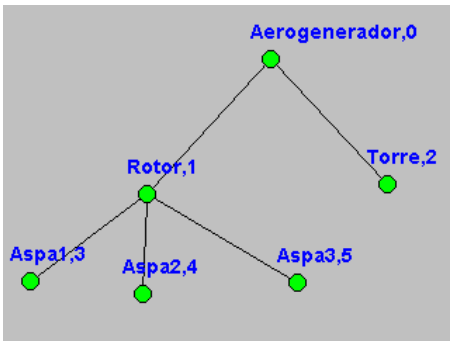

Figura 5.9: Organización de hyper-árbol para el ejemplo de turbina eólica.

Luego de la construcción de los modelos de MSBN para cada caso de carga de comportamiento normal, se probó la eficacia de nuestro método para la detección de fallas usando los datos simulados para cada DLC de falla. Los principales resultados obtenidos, se describen en la siguiente sección.

5.3.3. Resumen de los resultados con simulaciones de turbinas eólicas.

Para evaluar la efectividad de nuestra propuesta para la detección de fallas en turbinas eólicas, para cada caso de falla, evaluamos la precisión de nuestro método para devolver dentro de las primeras posiciones la variable que por diseño en FOCUS6 activa los sistemas de control al sobrepasar los umbrales permitidos. Nuestro sistema devuelve aquellas variables con mayor probabilidad de fallas reales, sin embargo el simulador sólo nos ofrece la etiqueta del tipo de falla que ocurre, no aquellas variables sobre las cuales actúa dicha falla.

Las tablas 5.12 - 5.16 muestran los resultados en cuanto a P@1, P@3, P@5 y MAP del método centralizado vs nuestra propuesta (método distribuido) para cada ejemplo, que se identifica con el DLC de comportamiento normal que le corresponde. De igual manera los resultados obtenidos muestran que en cuanto a precisión ambos métodos tienen resultados similares, en el 96 % de los casos ambos métodos devuelven

dentro de las 5 primeras posiciones la variable que identifica la falla (consideramos como variable relevante la que por diseño en el simulador activa el control para cada tipo de falla). Los resultados del MAP indican que a pesar de devolver la variable relevante, no siempre esto ocurre en la primera posición

Tabla 5.12: Resultados en cuanto a P@1, P@3, P@5 y MAP del método centralizado vs nuestra propuesta (método distribuido) para el ejemplo del DLC 01300.

	P@1	**P@3**	**P@5**	**MAP**
Método centralizado	0.4286	0.8571	1	0.6786
Método distribuido	0.4286	0.8571	1	0.6786

Tabla 5.13: Resultados en cuanto a P@1, P@3, P@5 y MAP del método centralizado vs nuestra propuesta (método distribuido) para el ejemplo del DLC 12150.

	P@1	**P@3**	**P@5**	**MAP**
Método centralizado	0.4285	1	1	0.667
Método distribuido	0.4285	1	1	0.667

Tabla 5.14: Resultados en cuanto a P@1, P@3, P@5 y MAP del método centralizado vs nuestra propuesta (método distribuido) para el ejemplo del DLC 12110.

	P@1	**P@3**	**P@5**	**MAP**
Método centralizado	0.3333	0.6667	0.8333	0.5333
Método distribuido	0.3333	0.6667	0.8333	0.5333

Las Tablas 5.17 - 5.21 muestra los resultados en cuanto al tiempo de ejecución del método centralizado vs nuestro método distribuido. Los tiempos se indican en minutos. Para este análisis se realizó una división entre el tiempo aproximado del aprendizaje de los parámetros (para cada DLC el aprendizaje de los parámetros se realiza una sola vez) y el tiempo promedio del proceso de detección y aislamiento de la falla. Señalar que los promedios de los tiempos de ejecución asociados a la detección y aislamiento de las fallas están en correspondencia con la cantidad de

Tabla 5.15: Resultados en cuanto a P@1, P@3, P@5 y MAP del método centralizado vs nuestra propuesta (método distribuido) para el ejemplo del DLC 12200.

	P@1	**P@3**	**P@5**	**MAP**
Método centralizado	0	1	1	0.5
Método distribuido	0	1	1	0.5

Tabla 5.16: Resultados en cuanto a P@1, P@3, P@5 y MAP del método centralizado vs nuestra propuesta (método distribuido) para el ejemplo del DLC 12131.

	P@1	**P@3**	**P@5**	**MAP**
Método centralizado	0.5	1	1	0.75
Método distribuido	0.5	1	1	0.75

casos de fallas asociadas a cada DLC de comportamiento normal. Como puede verse, para cada uno de los ejemplos la reducción del tiempo de ejecución con el método distribuido resulta significativa.

Tabla 5.17: Resultados en cuanto al tiempo de ejecución del método centralizado vs nuestro método distribuido para el ejemplo del DLC 01300. Los tiempos se indican en minutos.

	Aprendizaje de los parámetros	**Detección y aislamiento**
Método centralizado	9.54	5.029
Método distribuido	3.652	0.982

Relativo a estos experimentos también se realizaron las pruebas de significancia estadística. De manera similar al diseño de los experimentos con circuitos lógicos, se realizaron dos pruebas, una para ver el comportamiento de las observaciones de acuerdo a la precisión y otro de acuerdo al tiempo de ejecución. La significancia estadística que se seleccionó fue del $5\,\%$ y la prueba de significancia seleccionada de acuerdo a las características de las observaciones (23 observaciones para el método centralizado y 23 para el método distribuido) fue la prueba de Wilcoxon.

Para el caso de la precisión la significación asintótica obtenida fue $p = 0.180$ por

Tabla 5.18: Resultados en cuanto al tiempo de ejecución del método centralizado vs nuestro método distribuido para el ejemplo del DLC 12150. Los tiempos se indican en minutos.

	Aprendizaje de los parámetros	**Detección y aislamiento**
Método centralizado	9.983	5.2614
Método distribuido	3.531	1.231

Tabla 5.19: Resultados en cuanto al tiempo de ejecución del método centralizado vs nuestro método distribuido para el ejemplo del DLC 12110. Los tiempos se indican en minutos.

	Aprendizaje de los parámetros	**Detección y aislamiento**
Método centralizado	9.67	5.755
Método distribuido	3.287	1.389

Tabla 5.20: Resultados en cuanto al tiempo de ejecución del método centralizado vs nuestro método distribuido para el ejemplo del DLC 12200. Los tiempos se indican en minutos.

	Aprendizaje de los parámetros	**Detección y aislamiento**
Método centralizado	9.387	4.59
Método distribuido	3.581	0.894

Tabla 5.21: Resultados en cuanto al tiempo de ejecución del método centralizado vs nuestro método distribuido para el ejemplo del DLC 12131. Los tiempos se indican en minutos.

	Aprendizaje de los parámetros	**Detección y aislamiento**
Método centralizado	9.628	5.6865
Método distribuido	3.569	1.297

lo que no se rechaza la hipótesis nula y por tanto no existen diferencias significativas entre los dos métodos.

Para el caso del análisis del tiempo de ejecución la significación asintótica obtenida fue $p = 0.000$ por lo que se acepta la hipótesis alternativa y por tanto existen diferencias significativas entre los dos métodos en cuanto al tiempo de ejecución.

De igual forma se realizó un análisis del poder estadístico de la selección de la cantidad de fallas. De acuerdo a los resultados obtenidos para el método centralizado conocemos que la eficacia del mismo para estos ejemplos está en torno al 96 % y se espera que, en cuanto a la precisión, con el método distribuido se mantenga igualmente sobre el 96 %. Tras finalizar el estudio tenemos 23 observaciones de cada método. Al realizar el análisis estadístico, se concluyó que no hay diferencias significativas en la efectividad de ambos métodos de acuerdo a la precisión. Podemos calcular cuál ha sido finalmente el poder del estudio. Aplicando la fórmula para el cálculo del poder estadístico de comparación de dos proporciones ante un planteamiento unilateral se obtiene una $Z_{1-\beta} = 1.645$ lo que se corresponde con un poder estadístico del 95 %.

5.4. Resumen del capítulo

En este capítulo se describieron los principales experimentos realizados aplicados a dos dominios de aplicación: los circuitos lógicos combinacionales y las simulaciones de la Máquina Eólica Mexicana. De manera general el método propuesto muestra un desempeño similar en cuanto a precisión con el método centralizado para ambos dominios. Los mejores resultados se obtienen en cuanto a P@5, lo que significa que ambos métodos encuentran en el 95 % de los casos en promedio las variables con comportamiento fallido dentro de las 5 primeras variables devueltas. Además, el método propuesto disminuye considerablemente el tiempo de ejecución asociado, sobre todo, a la inferencia de la red y a la evaluación de las redes de aislamiento.

En el siguiente capítulo se describen las principales conclusiones de esta tesis, así como el trabajo futuro derivado de la misma.

Capítulo 6

Conclusiones y trabajo futuro

6.1. Conclusiones

La detección de fallas es un aspecto importante a tener en cuenta en todos los sistemas actuales, pues de su detección oportuna depende la reducción de los costos relacionados con la reparación o la pérdida de la producción.

La mayoría de los trabajos en el área enfocan sus esfuerzos al uso de diferentes técnicas de Inteligencia Artificial para hacer de forma más precisa el proceso de detección de las fallas. Dentro de los trabajos encontrados en el estado del arte destaca el uso de las Redes Neuronales Artificiales donde a partir de un modelo del comportamiento aprendido se detectan las desviaciones de las variables considerándolas como fallas.

El uso de las Redes Bayesianas para la detección de fallas tiene importantes ventajas relacionadas con el manejo de la incertidumbre, no requiere de información de fallas para la detección, además de la detección de valores atípicos en etapas tempranas; sin embargo, su principal limitación es que, a medida que aumenta la densidad de la red del modelo del dominio, los mecanismos de inferencia son in-

eficientes, por lo que las Redes Bayesianas de Múltiples Secciones constituyen una alternativa pues permite modelar problemas de dominio grande donde los modelos de RB son muy tardados y donde el mecanismo de inferencia se realiza de manera eficiente.

En general, los sistemas complejos tales como las turbinas eólicas o los circuitos lógicos combinacionales, pueden dividirse naturalmente en sub-sistemas relativamente independientes que facilitan el uso del enfoque distribuido de detección de fallas propuesto.

En este documento, se propuso una extensión distribuida del trabajo presentado en [Ibargüengoytia et al., 2005], para la detección de fallas en problemas cuya densidad de red bayesiana hace muy tardado el proceso de detección. Con este propósito, usamos las MSBN, que particionan el dominio en subdominios más pequeños y donde la inferencia se realiza de manera local para cada sección con las correspondientes actualizaciones a las secciones adyacentes.

El método propuesto fue probado para la detección de fallas en circuitos lógicos combinacionales y en simulaciones de una turbina eólica. Con base en los experimentos, podemos concluir que el método propuesto mantiene la efectividad en términos de precisión con respecto al trabajo centralizado, mientras que reduce significativamente el tiempo de ejecución (para el caso de los circuitos lógicos combinacionales la reducción es de alrededor del 67 % mientras que para las simulaciones de la turbina eólica la reducción del aprendizaje de los parámetros es aproximadamente del 63 % y en la detección y aislamiento es del 78 %), lo que hace posible tratar con modelos de dominio más complejos.

6.2. Trabajo Futuro

Entre las ideas de trabajo futuro se encuentran: el desarrollo de un método automático para definir las secciones en las Redes Bayesianas de Múltiples Secciones pues hasta el momento este proceso se realiza de forma manual teniendo en cuenta la localidad de los modelos del dominio. El aprendizaje de patrones de fallas a partir de la salida de nuestro sistema para, a partir de las variables identificadas como fallas reales, aprender el tipo de falla que le corresponde. Además, considerar la aplicación del método propuesto a otros dominios de aplicación.

Bibliografía

[Aguilera et al., 2011] Aguilera, P., Fernández, A., Fernández, R., Rumí, R., and Salmerón, A. (2011). Bayesian networks in environmental modelling. *Environmental Modelling & Software*, 26(12):1376–1388.

[Baker and Boult, 1990] Baker, M. and Boult, T. E. (1990). Pruning bayesian networks for efficient computation. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, pages 225–232. Elsevier Science Inc.

[Bar-Yam, 1997] Bar-Yam, Y. (1997). *Dynamics of complex systems*, volume 213. Addison-Wesley Reading, MA.

[Bertrand-Krajewski et al., 2007] Bertrand-Krajewski, J.-L., Winkler, S., Saracevic, E., Torres, A., and Schaar, H. (2007). Comparison of and uncertainties in raw sewage cod measurements by laboratory techniques and field uv-visible spectrometry. *Water Science and Technology*, 56(11):17–25.

[Böhme et al., 1991] Böhme, T., Cox, C., Valentin, N., and Denoeux, T. (1991). Comparison of autoassociative neural networks and kohonen maps for signal failure detection and reconstruction. *Intelligent Engineering Systems through Artificial Neural Networks*, 9:637–644.

[Boser et al., 1992] Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM.

[Branisavljević et al., 2011] Branisavljević, N., Kapelan, Z., and Prodanović, D. (2011). Improved real-time data anomaly detection using context classification. *Journal of Hydroinformatics*, 13(3):307–323.

[CEMIE, 2014] CEMIE (2014). Proyecto p12: Desarrollo de tecnología basada en inteligencia artificial y mecatrónica, para integrar un parque de generación de energía eólica a una red inteligente. `http://www.cemieeolico.org.mx/Proyectos/Proyecto-P12`.

[Chen and Pollino, 2012] Chen, S. H. and Pollino, C. A. (2012). Good practice in bayesian network modelling. *Environmental Modelling & Software*, 37:134–145.

[Cooper, 1990] Cooper, G. F. (1990). The computational complexity of probabilistic inference using bayesian belief networks. *Artificial intelligence*, 42(2-3):393–405.

[De Kleer and Williams, 1987] De Kleer, J. and Williams, B. C. (1987). Diagnosing multiple faults. *Artificial intelligence*, 32(1):97–130.

[Díez, 1996] Díez, F. J. (1996). Local conditioning in bayesian networks. *Artificial Intelligence*, 87(1-2):1–20.

[Dougherty et al., 1995] Dougherty, J., Kohavi, R., and Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In *Machine Learning Proceedings 1995*, pages 194–202. Elsevier.

[Edthofer et al., 2010] Edthofer, F., Van den Broeke, J., Ettl, J., Lettl, W., and Weingartner, A. (2010). Reliable online water quality monitoring as basis for fault tolerant control. In *Control and Fault-Tolerant Systems (SysTol), 2010 Conference on*, pages 57–62. IEEE.

[Eryurek and Upadhyaya, 1990] Eryurek, E. and Upadhyaya, B. (1990). Sensor validation for power plants using adaptive backpropagation neural network. *IEEE Transactions on Nuclear Science*, 37(2):1040–1047.

[Fayyad and Irani, 1993] Fayyad, U. and Irani, K. (1993). Multi-interval discretization of continuous-valued attributes for classification learning.

[Goebel and Agogino, 1996] Goebel, K. and Agogino, A. (1996). An architecture for fuzzy sensor validation and fusion for vehicle following in automated highways. In *Proceedings of the 29th International Symposium on Automotive Technology and Automation*.

[Graphics, 2015] Graphics, M. (2015). Modelsim simulator.

[Guo and Nurre, 1991] Guo, T.-H. and Nurre, J. (1991). Sensor failure detection and recovery by neural networks. In *Neural Networks, 1991., IJCNN-91-Seattle International Joint Conference on*, volume 1, pages 221–226. IEEE.

[Hassoun, 1995] Hassoun, M. H. (1995). *Fundamentals of artificial neural networks*. MIT press.

[Haykin et al., 2001] Haykin, S. S. et al. (2001). *Kalman filtering and neural networks*. Wiley Online Library.

[Heckerman, 1990] Heckerman, D. (1990). Probabilistic similarity networks. *Networks*, 20(5):607–636.

[Hill, 1990] Hill, M. D. (1990). What is scalability? *ACM SIGARCH Computer Architecture News*, 18(4):18–21.

[Holbert et al., 1994] Holbert, K. E., Heger, A. S., and Alang-Rashid, N. K. (1994). Redundant sensor validation by using fuzzy logic. *Nuclear Science and Engineering*, 118(1):54–64.

[Ibarguengoytia et al., 1997] Ibarguengoytia, P. et al. (1997). *Any time probabilistic sensor validation*. PhD thesis, University of Salford, UK.

[Ibargüengoytia et al., 2005] Ibargüengoytia, P. H., Vadera, S., and Sucar, L. E. (2005). A probabilistic model for information and sensor validation. *The Computer Journal*, 49(1):113–126.

[Isermann, 2006] Isermann, R. (2006). *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer Science & Business Media.

[Jovanovic, 2014] Jovanovic, D. P. (2014). Fault detection in complex and distributed systems.

[Khadem et al., 1993] Khadem, M., Alexandro, F., and Colley, R. (1993). Sensor validation in power plants using neural networks. *Neural Network Computing for the Electric Power Industry*, pages 51–54.

[Kononenko, 1995] Kononenko, I. (1995). On biases in estimating multi-valued attributes. In *Ijcai*, volume 95, pages 1034–1040. Citeseer.

[Lauritzen and Spiegelhalter, 1988] Lauritzen, S. L. and Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 157–224.

[Lehrig et al., 2015] Lehrig, S., Eikerling, H., and Becker, S. (2015). Scalability, elasticity, and efficiency in cloud computing: A systematic literature review of definitions and metrics. In *Proceedings of the 11th International ACM SIGSOFT Conference on Quality of Software Architectures*, pages 83–92. ACM.

[Liu et al., 2015] Liu, W., Tang, B., Han, J., Lu, X., Hu, N., and He, Z. (2015). The structure healthy condition monitoring and fault diagnosis methods in wind turbines: A review. *Renewable and Sustainable Energy Reviews*, 44:466–472.

[Morales and Garcia, 1990] Morales, E. and Garcia, H. (1990). A modular approach to multiple faults diagnosis. In *Artificial intelligence in Process Engineering*, pages 161–187. Elsevier.

[Napolitano et al., 1998] Napolitano, M. R., Windon, D. A., Casanova, J. L., Innocenti, M., and Silvestri, G. (1998). Kalman filters and neural-network schemes for sensor validation in flight control systems. *IEEE transactions on control systems technology*, 6(5):596–611.

[Pearl, 1986a] Pearl, J. (1986a). Fusion, propagation, and structuring in belief networks. *Artificial intelligence*, 29(3):241–288.

[Pearl, 1986b] Pearl, J. (1986b). On evidential reasoning in a hierarchy of hypotheses. *Artificial intelligence*, 28(1):9–15.

[Pearl, 1987] Pearl, J. (1987). Evidential reasoning using stochastic simulation of causal models. *Artificial Intelligence*, 32(2):245–257.

[Pires et al., 2016] Pires, I. M., Garcia, N. M., Pombo, N., Flórez-Revuelta, F., and Rodríguez, N. D. (2016). Validation techniques for sensor data in mobile health applications. *Journal Sensors*, 2016:2839372:1–2839372:9.

[Qin and Li, 1999] Qin, S. J. and Li, W. (1999). Detection, identification, and reconstruction of faulty sensors with maximized sensitivity. *AIChE journal*, 45(9):1963–1976.

[Sucar, 2015] Sucar, L. E. (2015). *Probabilistic Graphical Models - Principles and Applications*. Advances in Computer Vision and Pattern Recognition. Springer.

[Suermondt et al., 2013] Suermondt, J., Cooper, G. F., and Heckerman, D. (2013). A combination of cutset conditioning with clique-tree propagation in the pathfinder system. *arXiv preprint arXiv:1304.1114*.

[Sun et al., 2011] Sun, S., Bertrand-krajewski, J.-l., Lynggaard-Jensen, A., Broerke, J., Edthofer, F., Almeida, M., Ribeiro, A., and Menaia, J. (2011). Literature review for data validation methods. *Science and Technology*, 47(2):95–102.

[Tipping, 2001] Tipping, M. E. (2001). Sparse bayesian learning and the relevance vector machine. *Journal of machine learning research*, 1(Jun):211–244.

[Valentin et al., 2001] Valentin, N. et al. (2001). A neural network-based software sensor for coagulation control in a water treatment plant. *Intelligent Data Analysis*, 5(1):23–39.

[Wilkinson et al., 2005] Wilkinson, B., Allen, C., et al. (2005). *Parallel programming: techniques and applications using networked workstations and parallel computers*. Upper Saddle River, NJ: Pearson/Prentice Hall.

[WMC, 2018] WMC (2018). Focus6: The integrated modular wind turbine design tool. `https://wmc.eu/about.php`.

[Xiang, 1996] Xiang, Y. (1996). A probabilistic framework for cooperative multiagent distributed interpretation and optimization of communication. *Artificial Intelligence.*, 87(1-2):295–342.

[Xiang, 1998] Xiang, Y. (1998). Verification of dag structures in cooperative belief network-based multiagent systems. *Networks: An International Journal*, 31(3):183–191.

[Xiang, 2000] Xiang, Y. (2000). Belief updating in multiply sectioned bayesian networks without repeated local propagations. *International Journal of Approximate Reasoning*, 23(1):1–21.

[Xiang, 2003] Xiang, Y. (2003). Comparison of multiagent inference methods in multiply sectioned bayesian networks. *Int. J. Approx. Reasoning*, 33(3):235–254.

[Xiang, 2006] Xiang, Y. (2006). Webweavr-iv research toolkit.

[Xiang, 2008] Xiang, Y. (2008). Building intelligent sensor networks with multiagent graphical models. In *Intelligent decision making: An AI-based approach*, pages 289–320. Springer.

[Xiang, 2013] Xiang, Y. (2013). Optimization of inter-subnet belief updating in multiply sectioned bayesian networks. *CoRR*, abs/1302.4991.

[Xiang and Chen, 2004] Xiang, Y. and Chen, X. (2004). Interface verification for multiagent probabilistic inference. In *Advances in Bayesian networks*, pages 19–38. Springer.

[Xiang and Jensen, 2013] Xiang, Y. and Jensen, F. V. (2013). Inference in multiply sectioned bayesian networks with extended shafer-shenoy and lazy propagation. *CoRR*, abs/1301.6749.

[Xiang et al., 2005] Xiang, Y., Jensen, F. V., and Chen, X. (2005). Inference in multiply sectioned bayesian networks: Methods and performance comparison. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 36(3):546–558.

[Xiang et al., 1993a] Xiang, Y., Pant, B., Eisen, A., Beddoes, M. P., and Poole, D. (1993a). Multiply sectioned bayesian networks for neuromuscular diagnosis. *Artificial Intelligence in Medicine*, 5(4):293–314.

[Xiang et al., 1993b] Xiang, Y., Poole, D., and Beddoes, M. P. (1993b). Multiply sectioned bayesian networks and junction forests for large knowledge-based systems. *Computational Intelligence*, 9(2):171–220.

[Zhang and Poole, 1996] Zhang, N. L. and Poole, D. (1996). Exploiting causal independence in bayesian network inference. *Journal of Artificial Intelligence Research*, 5:301–328.