UNIVERSIDAD NACIONAL DEL CENTRO DE LA PROVINCIA DE BUENOS AIRES
FACULTAD DE CIENCIAS EXACTAS - DPTO. DE COMPUTACION Y SISTEMAS
DOCTORADO EN CIENCIAS DE LA COMPUTACION

El enfoque IBMAP para aprendizaje de estructuras de redes de Markov

por
Federico Schlüter

Director: Prof. Dr. Facundo Bromberg

Lab. DHARMa de Inteligencia Artificial

Departamento de Sistemas de Información, Facultad Regional Mendoza,
Universidad Tecnológica Nacional

Tesis sometida a evaluación para la obtención del grado de

Doctor en Ciencias de la Computación

2 de diciembre de 2014

Prefacio

Las *redes de Markov* son modelos probabilísticos gráficos, una herramienta computacional para el modelado eficiente de distribuciones de probabilidad que utiliza grafos para facilitar la representación de problemas complejos. Estos modelos han sido diseñados para ser manipulados por sistemas expertos, utilizando la teoría de la probabilidad para razonar eficientemente bajo condiciones de incertidumbre. Sin embargo, una limitación importante para el uso de estos modelos es que en la práctica resulta complejo diseñarlos manualmente, ya que el conocimiento de expertos no siempre es suficiente, sumado al hecho de que muchos dominios reales poseen una gran dimensionalidad. Por esto, el *aprendizaje* de estos modelos a partir de datos es un tópico que ha tomado gran relevancia, ya que la disponibilidad de datos digitales es cada vez mayor, resultando además en un mecanismo interesante para descubrir nuevo conocimiento a partir de datos digitales.

En este trabajo, la investigación se centra en un enfoque específico de aprendizaje de redes de Markov: los algoritmos *basados en independencia*. Estos algoritmos están diseñados para aprender la estructura de independencias del modelo, que es la componente que codifica de un modo compacto el conocimiento sobre la distribución. Los algoritmos basados en independencia han sido diseñados bajo lineamientos teóricos que permiten aprender la estructura de un modo eficiente y robusto, a partir de la ejecución de un conjunto de *tests estadísticos de independencia* sobre los datos. Comúnmente, los resultados de dichos tests son utilizados como restricciones que guían una búsqueda en el espacio de las estructuras de independencia posibles, convergiendo a una estructura que satisface los resultados de todos los tests. Estos algoritmos garantizan que la estructura aprendida es correcta bajo la suposición de que los tests estadísticos son confiables. No obstante, un hecho muy común en la práctica suele ser que los datos disponibles no son suficientes para obtener resultados correctos desde los tests estadísticos. Cuando esto sucede, los algoritmos basados en independencia acumulan

y propagan suposiciones de independencia incorrectas, resultando en un aprendizaje con gran cantidad de errores estructurales. Este hecho, que resulta en una limitación de real importancia a la hora de aplicar esta tecnología, es la motivación de la presente tesis.

En este trabajo se presenta el enfoque de máximo a posteriori basado en independencias para aprendizaje de estructuras de redes de Markov (IB-MAP, del inglés *independence-based maximum-a-posteriori*). Dado que los algoritmos tradicionales descartan la estructura correcta cada vez que ejecutan un test erróneo, se propone un enfoque que asigna probabilidades a las distintas estructuras, sin descartar ninguna. Para esto, se propone una función de puntaje de estructuras basada en tests estadísticos denominada *IB-score* (puntaje basado en independencias). Esta función permite computar de un modo aproximado la probabilidad a posteriori de una estructura dados los datos $Pr(G|D)$, combinando los resultados de un conjunto de tests estadísticos. De este modo, las diferentes estructuras poseen un puntaje más alto o más bajo según las probabilidades de las independencias que codifican. En resumen, el enfoque propuesto consiste en la maximización de la función IB-score sobre el espacio de todas las estructuras posibles. A modo de instanciación de este enfoque se presenta una serie de algoritmos que maximizan dicha función con diversos mecanismos de optimización.

Para validar el enfoque se evaluó el desempeño de las distintas instanciaciones de la búsqueda, evaluando la calidad de las estructuras aprendidas con respecto a algoritmos basados en independencia que pertenecen al estado del arte. Se presentan resultados sistemáticos sobre una gran variedad de dimensiones del problema, demostrando que este enfoque permite mejorar significativamente la calidad de las estructuras aprendidas. Se demuestra también que dichas mejoras pueden obtenerse a un costo computacional competitivo respecto de los algoritmos del estado del arte. Dicha experimentación fue llevada a cabo sobre datos sintéticos y datos del mundo real, y en una aplicación de aprendizaje de estructuras para algoritmos evolutivos.

Palabras clave: Redes de Markov, Aprendizaje de estructuras, Tests de independencia, MAP

“Dedicado a todas las personas que actúan siguiendo la voz de su corazón, a todos los que creen en la fuerza del trabajo colectivo, y a los que trabajan con pasión persiguiendo la misión de servirle al mundo...”

Agradecimientos

Existe una gran cantidad de gente que merece mi agradecimiento, luego de toda la dedicación y cariño que he volcado en el desarrollo de esta tesis. Primero que nada, quiero expresar mi gratitud más sincera a Facundo Bromberg, por ser mi mentor y guía. Su sencillez y genialidad particulares me han sido siempre un ejemplo a seguir. El cruce de nuestros caminos no ha sido casual, estoy seguro. También quiero agradecer a algunos compañeros especiales, de los cuales he aprendido muchas cosas en lo profesional y humano. En particular quiero agradecer a David Monge, Alejandro Edera, Sebastián Pérez y Ana Diedrichs por ser excelentes colegas-amigos, por las charlas inspiradoras, por lo compartido en general, y por compartir ese amor por el trabajo desafiante y apasionante, el interés por superar nuestras propias barreras.

Quiero expresar además mi gratitud hacia el comité de becas de la Universidad Tecnológica Nacional, y el FONCyT, por la beca que se me otorgó para formarme profesionalmente como investigador. Esta beca dio un puntapié inicial a este doctorado, que es lo más interesante que me ha sucedido profesionalmente, apoyándome también con viáticos en bibliografía y en pasajes para asistir a valiosos cursos de posgrado en UNICEN. Extiendo este agradecimiento al CONICET, por completar la financiación de esta investigación. De ambas becas me siento muy agradecido, durante estos años he descubierto un mundo fascinante, conociendo a gente excelente en lo profesional y lo humano.

Por último, agradezco profundamente a mis padres Enrique y María Elena, por darme todo y enseñarme con el ejemplo lo importante que es sacrificarse por lo que uno ama. A mis hermanas y mis sobrinos, por el amor incondicional. A mi compañera de la vida, Rocío, mi combustible, la razón de mi equilibrio. Por último, agradezco a todos mis hermanos y compañeros de la música, el universo paralelo que llena mi vida de sonidos, colores y vivencias invaluables.

Fede

Contenido

Página

Lista de Figuras vii

Lista de Tablas xiii

Lista de Algoritmos xvi

1. Introducción 1

1.1. Motivación 4

1.2. Tesis 5

1.3. Organización 6

2. Marco teórico 9

2.1. Redes de Markov 9

2.1.1. La estructura de independencias 10

2.1.2. Parametrización del modelo 11

2.1.3. Correctitud de la estructura 12

2.1.4. La manta de Markov de una variable 15

2.2. El aprendizaje de redes de Markov 15

2.2.1. Objetivos del aprendizaje 16

2.2.2. Estimación paramétrica 17

2.2.3. Aprendizaje de la estructura de independencias 19

2.2.3.1. Enfoque basado en puntaje 19

2.2.3.2. Enfoque basado en independencia 22

3. Análisis del estado del arte **27**
3.1. El algoritmo GSMN 27
3.2. El algoritmo GSIMN 29
3.3. El algoritmo PFMN 30
3.4. El algoritmo DGSIMN 31
3.5. El test argumentativo de independencia 32
3.6. Conclusiones 32

4. El enfoque IBMAP **35**
4.1. Una función de puntaje de estructuras basada en independencias . 36
4.2. El conjunto de cierre basado en mantas de Markov 38
4.3. Optimización de la función de puntaje basada en independencias . 40
4.3.1. Búsqueda por fuerza bruta 42
4.3.2. Búsqueda local por ascensión de colinas 43
4.3.3. Ascensión de colinas con reinicios múltiples 45
4.3.4. Búsqueda con un algoritmo genético simple 47
4.3.5. Búsqueda heurística de ascensión de colinas 51
4.3.6. Búsqueda heurística de ascensión de colinas con reinicios múltiples 54

5. Evaluación experimental **57**
5.1. Evaluación de calidad estructural en distribuciones pequeñas sobre datos sintéticos 58
5.2. Evaluación de calidad estructural con el enfoque basado en algoritmo genético 66
5.3. Evaluación de escalabilidad de la calidad estructural sobre la dimensionalidad de las distribuciones 76
5.4. Evaluación de calidad estructural en datos reales 84
5.5. Análisis de complejidad temporal sobre datos sintéticos 86
5.6. Análisis de superficie de la función IB-score 97
5.7. Aplicando IBMAP-HHC a EDAs 102
5.8. Resumen 106

6. Resumen y conclusiones **109**
6.1. Investigación complementaria realizada 110
6.1.1. Algoritmo GSS 111
6.1.2. Acelerando la ejecución masiva de tests 111
6.1.3. Mejorando estrategias para algoritmos LGL 112
6.1.4. Distribuciones con independencias específicas del contexto 113
6.2. Trabajo futuro 113
6.2.1. Relajación de IB-score 113
6.2.2. Diseño de conjuntos de cierre alternativos 115
6.2.3. Diseño de métodos de búsqueda alternativos 116
6.2.4. Aplicación de AD-tree para aceleración de enfoque IBMAP 117
6.2.5. Comentarios finales 118

A. Test Bayesiano de independencia condicional **119**
A.1. Verosimilitud del modelo dependiente 121

B. Correctitud del conjunto de cierre basado en mantas de Markov **125**

C. El Algoritmo HHC-MN **129**

Bibliografía **131**

Lista de Figuras

2.1. Ejemplo de dos estructuras de independencia. (a) Un grafo irregular con diferentes grados de conectividad sobre sus nodos, y (b) una rejilla regular donde las variables pertenecen al dominio de un problema espacial. 11

4.1. Ejemplo de tabla de adyacencias para la Figura 2.1(a) 41
4.2. Ejemplo de espacio de estados entre posibles estructuras de independencia. 44
4.3. Esquema general de un algoritmo genético. 48
4.4. Un grafo de ejemplo para un problema de 3 variables, su correspondiente matriz de adyacencias, y el cromosoma que codificaría el grafo de ejemplo. 48
4.5. Ejemplo de cruce uniforme para dos cromosomas de tamaño 18. . 49

5.1. Distancia de Hamming en problemas con $n = 6$ variables. Menor distancia de Hamming es mejor. Lista de algoritmos: [1] GSMN, [2] HHC-MN, [3] IBMAP-BF, [4] IBMAP-HC, [5] IBMAP-HC-RR(100), [6] IBMAP-HC-RR(500), [7] IBMAP-HC-RR(1000), [8] IBMAP-HHC, [9] IBMAP-HHC-RR(100), [10] IBMAP-HHC-RR(500), [11] IBMAP-HHC-RR(1000). 61

5.2. Distancia de Hamming en problemas con $n = 12$ variables. Menor distancia de Hamming es mejor. Lista de algoritmos: [1] GSMN, [2] HHC-MN, [4] IBMAP-HC, [5] IBMAP-HC-RR(100), [6] IBMAP-HC-RR(500), [7] IBMAP-HC-RR(1000), [8] IBMAP-HHC, [9] IBMAP-HHC-RR(100), [10] IBMAP-HHC-RR(500), [11] IBMAP-HHC-RR(1000). 64

5.3. Distancia de Hamming en problemas con $n = 20$ variables. Menor distancia de Hamming es mejor. Lista de algoritmos: [1] GSMN, [2] HHC-MN, [4] IBMAP-HC, [5] IBMAP-HC-RR(100), [6] IBMAP-HC-RR(500), [7] IBMAP-HC-RR(1000), [8] IBMAP-HHC, [9] IBMAP-HHC-RR(100), [10] IBMAP-HHC-RR(500), [11] IBMAP-HHC-RR(1000). 65

5.4. Evolución de IBMAP-GA utilizando una población de 10 individuos y selección de supervivientes steady-state. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales. 67

5.5. Evolución de IBMAP-GA utilizando una población de 100 individuos y selección de supervivientes steady-state. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales. 68

5.6. Evolución de IBMAP-GA utilizando una población de 500 individuos y selección de supervivientes steady-state. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales. 69

5.7. Evolución de IBMAP-GA utilizando una población de 10 individuos y selección de supervivientes D-crowding. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales. . . . 70

5.8. Evolución de IBMAP-GA utilizando una población de 100 individuos y selección de supervivientes D-crowding. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales. . . . 71

5.9. Evolución de IBMAP-GA utilizando una población de 500 individuos y selección de supervivientes D-crowding. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales. . . . 72

5.10. Distancia de Hamming en problemas con $n \in \{12, 16, 20\}$ variables para GSMN, IBMAP-HHC y IBMAP-GA. Menor distancia de Hamming es mejor. 74

5.11. Distancia de Hamming de problemas con $n = 50$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). Menor distancia de Hamming es mejor. 77
5.12. Distancia de Hamming de problemas con $n = 100$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). Menor distancia de Hamming es mejor. 78
5.13. Distancia de Hamming de problemas con $n = 200$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). Menor distancia de Hamming es mejor. 79
5.14. Distancia de Hamming de problemas con $n = 500$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). Menor distancia de Hamming es mejor. 81
5.15. Distancia de Hamming de problemas con $n \in \{750\}$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). Menor distancia de Hamming es mejor. 82
5.16. Distancia de Hamming de problemas con $n \in \{50, 100, 200, 500, 750\}$ variables (filas) para estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (columnas). Menor distancia de Hamming es mejor. . . 83
5.17. Tiempo de corrida para problemas con $n = 50$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). 87
5.18. Tiempo de corrida para problemas con $n = 100$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). 88

5.19. Tiempo de corrida para problemas con $n = 200$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). 89
5.20. Tiempo de corrida para problemas con $n = 500$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). 90
5.21. Tiempo de corrida para problemas con $n = 750$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). 91
5.22. Tiempo de corrida para problemas con $n \in \{50, 100, 200, 500, 750\}$ variables (filas), con estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (columnas). 92
5.23. Cantidad de ascensos M de IBMAP-HHC (eje Y) para problemas con $n \in \{6, 12, 16, 20, 24, 30, 50, 75, 100, 200, 500, 750\}$ variables (eje X), utilizando conjuntos de datos de tamaños crecientes $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$. 95
5.24. Cantidad de ascensos M de IBMAP-HC (eje Y) para problemas con $n \in \{6, 12, 16, 20, 24, 30\}$ variables (eje X), utilizando conjuntos de datos de tamaños crecientes $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$. 96
5.25. Superficie de IB-score para datos sintéticos de $n = 6$ variables, con tamaños $D \in \{10, 100, 1000\}$ en las columnas, y densidades $\tau \in \{1, 2, 4\}$ en las filas. El eje X ordena las estructuras por distancia de Hamming respecto de la estructura correcta. El eje Y muestra el IB-score de todas las estructuras del espacio de estados. La estructura encontrada por IBMAP-BF se indica con un círculo grande. Las estructuras encontradas por IBMAP-HHC y IBMAP-GA se indican con triángulos. 98

5.26. Superficie de IB-score para datos sintéticos de $n = 20$ variables, con tamaños $D \in \{10, 100, 1000\}$ en las columnas, y densidades $\tau \in \{1, 2, 4\}$ en las filas. El eje X ordena las estructuras por distancia de Hamming respecto de la estructura correcta. El eje Y muestra el IB-score de todas las estructuras del espacio de estados. Las estructuras encontradas por IBMAP-HHC y IBMAP-GA se indican con triángulos. 100

5.27. Superficie de IB-score para datos sintéticos de $n = 50$ variables, con tamaños $D \in \{10, 100, 1000\}$ en las columnas, y densidades $\tau \in \{1, 2, 4\}$ en las filas. El eje X ordena las estructuras por distancia de Hamming respecto de la estructura correcta. El eje Y muestra el IB-score de todas las estructuras del espacio de estados. Las estructuras encontradas por IBMAP-HHC y IBMAP-GA se indican con triángulos. 101

Lista de Tablas

3.1. Resumen de algoritmos basados en independencia para redes de Markov 34

5.1. Accuracy sobre diversos conjuntos de datos reales. La estructura se aprende utilizando un subconjunto del 75 % llamado conjunto de entrenamiento, y la accuracy se computa utilizando el 25 % restante (datos de testeo). Para cada conjunto de datos, se indica en negrita el mejor resultado. 86

5.2. Resultados para MOA y MOA' (que usa IBMAP-HHC) para OneMax, para problemas de tamaño creciente (filas) en términos de tamaño crítico de población D^*, y el promedio y desviación estándar sobre 10 repeticiones del número de evaluaciones de la función de fitness f^* requerido para obtener el óptimo global. Menores valores de D^* y f^* son mejores. 105

5.3. Resultados para MOA y MOA' (que usa IBMAP-HHC) para Royal Road, para problemas de tamaño creciente (filas) en términos de tamaño crítico de población D^*, y el promedio y desviación estándar sobre 10 repeticiones del número de evaluaciones de la función de fitness f^* requerido para obtener el óptimo global. Menores valores de D^* y f^* son mejores. 105

Lista de Algoritmos

1. Estrategia LGL para redes de Markov 23
2. GS($X, \mathbf{V}$). 28
3. IBMAP-BF($D, \mathbf{V}$). 42
4. IBMAP-HC($D, \mathbf{V}$). 44
5. IBMAP-HC-RR($D, \mathbf{V}, k$). 46
6. IBMAP-HHC($D, \mathbf{V}$). 52
7. heurística-mejor-vecino(G, IB-score(G)) 52
8. IBMAP-HHC-RR($D, \mathbf{V}, k$). 55
9. Test estadístico Bayesiano($D, X, Y, \mathbf{Z}$) 122
10. HITON-PC($X, \mathbf{V}$). 130

Capítulo 1

Introducción

En la actualidad el conocimiento podría considerarse una de las piezas de mayor valor para la humanidad. Conjuntamente, la existencia de datos en formato digital es cada vez más ubicua. Para aprovechar esto, dentro del área de aprendizaje de máquinas se ha desarrollado una gran cantidad de mecanismos para extraer conocimiento de modo automatizado, a partir del análisis de datos digitales. Una de las estrategias más poderosas que ha surgido en las últimas décadas consiste en aprender *modelos probabilísticos gráficos* desde datos (Pearl, 1988; Lauritzen, 1996; Koller y Friedman, 2009). Esta representación permite generalizar la lógica proposicional y almacenar distribuciones de probabilidad conjunta de un modo compacto. Con estos modelos se conforman bases de conocimiento probabilísticas, las cuales comúnmente son utilizadas para razonar bajo condiciones de incertidumbre en sistemas inteligentes.

Los modelos probabilísticos gráficos están conformados por dos componentes: una *estructura de independencias*, y un *conjunto de parámetros numéricos*. La estructura codifica las independencias que se encuentran presentes en la distribución, y define una familia de distribuciones de probabilidad. La codificación de dichas independencias es la razón por la cual la distribución puede almacenarse compactamente. Para definir una distribución única, el conjunto de parámetros numéricos cuantifica las relaciones de la estructura. Los dos tipos de modelos gráficos más populares son las *redes de Bayes* y las *redes de Markov*. Las redes de Bayes han sido ampliamente utilizadas para codificar distribuciones donde las

dependencias pueden representarse con un grafo dirigido acíclico. En cambio, las redes de Markov (también conocidas como *Markov Random Fields* ó *undirected graphical models*) permiten codificar distribuciones donde las dependencias pueden representarse con un grafo no dirigido. Las lecturas introductorias más recomendables a nivel teórico sobre modelos probabilísticos gráficos en general son los libros de Pearl (1988) y Lauritzen (1996). Adicionalmente, el libro de Koller y Friedman (2009) es el libro de mayor envergadura en la actualidad, cubriendo un amplio espectro de los aspectos del área.

Respecto de la utilización de estos modelos en aplicaciones prácticas, actualmente la literatura posee una vasta cantidad de ejemplos en un amplio rango de campos de la ciencia. Por ejemplo, en las áreas de visión computacional y análisis de imágenes, Besag et al. (1991) muestra su aplicación en arqueología y epidemiología; ó Anguelov et al. (2005), donde se ataca el problema de segmentación de objetos 3D. También resulta interesante el libro de Li (2001), que presenta una exposición del uso de redes de Markov para restauración de imágenes y detección de bordes. Hay más ejemplos en el área de minería de datos espaciales y geoestadística. El libro de Cressie (1992) describe cómo utilizar redes de Markov para modelar rejillas de datos espaciales. También el trabajo de Shekhar et al. (2004) presenta métodos de análisis espacial y aplicaciones de redes de Markov en campos como biología, economía espacial, ciencias del clima y la tierra, ecología, geografía, agronomía, entre otros. También hay ejemplos de uso de modelos gráficos en el área de medicina, como Schmidt et al. (2008) que presenta un método para detectar enfermedades coronarias a través del procesamiento de imágenes de eco-cardiogramas, ó el trabajo de Van Haaren et al. (2013) que utiliza aprendizaje de redes de Markov para estudiar la co-ocurrencia entre enfermedades; y en el área de biología computacional Friedman et al. (2000) propone el uso de redes de Bayes para aprender interacciones entre genes. Hay más aplicaciones de modelos gráficos en el área de optimización evolutiva (Mühlenbein y Paaß, 1996; Larrañaga et al., 2012), como Larrañaga y Lozano (2002) que describe el uso de redes de Bayes para modelar la distribución de probabilidades de individuos con alto valor de fitness en algoritmos evolutivos, o más recientemente Alden (2007); Shakya et al. (2012), donde aplican redes de Markov para el mismo propósito. También hay aplicaciones novedosas en el área de Recuperación de la información,

como Metzler y Croft (2005); Cai et al. (2007), donde utilizan redes de Markov para modelar las dependencias entre los términos de las consultas, o en Karyotis (2010), donde también se utilizan redes de Markov para modelar las dependencias contextuales y espaciales de la propagación de software malicioso. Esta lista de ejemplos podría extenderse por varias páginas más.

El amplio impacto que han tenido los modelos probabilísticos gráficos se debe a que éstos proveen de un marco de trabajo que soporta tres capacidades consideradas como "críticas" para los sistemas inteligentes, como se resalta en el libro de Koller y Friedman (2009):

i) *Representación*. Se provee de un modelo declarativo del conocimiento basado en grafos. Por un lado, estos modelos son compactos espacialmente, otorgando una representación eficiente y computacionalmente tratable de las independencias condicionales de una distribución de probabilidades. La clave de esta representación se encuentra en la explotación de las independencias, ya que comúnmente las variables aleatorias de una distribución sólo interactúan directamente con algunas de las demás variables. Por otro lado, por ser gráficos estos modelos son declarativos, lo que los hace prácticos para que un humano experto pueda entender y evaluar su semántica y sus propiedades.

ii) *Inferencia*. Dado un modelo gráfico, la tarea más fundamental es computar distribuciones marginales de un subconjunto del dominio. Esta tarea, usualmente llamada inferencia, se utiliza para llevar a cabo predicciones. La inferencia es la sub-rutina más elemental de los modelos gráficos. Sin embargo, la inferencia exacta es NP-completa en general (Cooper, 1990). Existen varios métodos aproximados para hacer inferencia. En Koller y Friedman (2009) se describen los métodos más populares, como *eliminación de variables*, *métodos de Monte Carlo*, y la *propagación de creencias iterativa*. Otros trabajos recientes son el *pase de mensajes en árboles ponderados* (Wainwright et al., 2003), *Power EP* (Minka, 2004), *propagación de creencias generalizada* (Yedidia et al., 2005), y *Pase de mensajes variacional* (Winn y Bishop, 2005). Interesantemente, existe una librería de código

abierto que provee implementaciones de varios de estos métodos (Mooij, 2010).

iii) *Aprendizaje*. La construcción de un modelo gráfico puede llevarse a cabo manualmente por un conjunto de humanos expertos, o puede llevarse a cabo aprendiendo este modelo automáticamente desde los datos. Existe gran cantidad de algoritmos que modelan la distribución de probabilidad de los datos, devolviendo un modelo gráfico como solución. Esto resulta muy poderoso, ya que el conocimiento de los expertos tiene sus límites, y no siempre resulta suficiente para diseñar un modelo propicio. Por esto, algunos autores consideran a estos algoritmos como una herramienta para *descubrimiento de conocimiento* (KDD, del inglés, *knowledge discovery in data bases*). Más aún, cuando se construyen modelos para un problema específico es posible incluso utilizar un enfoque híbrido, utilizando algunas partes del modelo diseñadas manualmente por expertos, y completando los detalles automáticamente, según lo aprendido desde los datos.

1.1. Motivación

Dadas las amplias capacidades de los modelos probabilísticos gráficos, y la vasta variedad de aplicaciones posibles de este tipo de modelos, el presente trabajo pretende contribuir específicamente sobre el área del aprendizaje de redes de Markov. Más específicamente, la contribución es un nuevo enfoque de aprendizaje de estructuras de independencia de dichos modelos. Su desarrollo se llevó a cabo continuando la línea de uno de los enfoques específicos que se ha desarrollado recientemente: los algoritmos *basados en independencia*.

El enfoque basado en independencia ha sido diseñado bajo sólidos lineamientos teóricos, permitiendo un aprendizaje eficiente y robusto a partir de la ejecución de un conjunto de *tests estadísticos de independencia* sobre los datos. Los resultados de dichos tests son utilizados como restricciones que guían una búsqueda en el espacio de las estructuras de independencia posibles, convergiendo a una estructura que satisface los resultados de todos los tests. Un aspecto interesante de estos algoritmos es que garantizan que la estructura aprendida es correcta,

bajo la suposición de que los tests estadísticos son correctos. Sin embargo, para computar estadísticas confiables los tests de independencia requieren poseer una cantidad de datos que crece exponencialmente con el número de variables que involucran. Por este motivo, en muchos casos de la práctica los datos no son suficientes para computar tests estadísticamente confiables. En estos casos, los algoritmos basados en independencia descartan la estructura correcta cada vez que ejecutan un test erróneo, y cometen errores en cascada durante su ejecución, es decir, los errores se acumulan y propagan, aprendiendo estructuras con gran cantidad de errores estructurales. Por esto, utilizar los resultados de los tests estadísticos como simples restricciones resulta ser una limitación importante del enfoque.

1.2. Tesis

Este trabajo propone un nuevo enfoque para aprendizaje de estructuras de independencias de redes de Markov. La motivación del enfoque presentado se debe a la necesidad de mejorar la calidad de las estructuras aprendidas por los algoritmos basados en independencia, que confían ciegamente en los resultados de los tests estadísticos. El objetivo es proveer un enfoque de aprendizaje más robusto, que aún estando basado en el uso de tests estadísticos, evita la propagación y acumulación de los errores estructurales generados por tests estadísticos incorrectos.

Es oportuno destacar que esta tesis se centra específicamente en el aprendizaje de estructuras correctas (es decir, sin errores estructurales), del mismo modo que los algoritmos basados en independencia. Si bien el aprendizaje de la red de Markov completa requiere de aprender la estructura de independencias y también los parámetros numéricos del modelo, los algoritmos estudiados en este trabajo centran su atención en el mejoramiento del aprendizaje estructural. Esto se debe a que el aprendizaje de la estructura cumple un rol más importante, ya que cuando se lleva a cabo el aprendizaje de los parámetros sobre estructuras incorrectas, las distribuciones aprendidas son muy diferentes a la distribución correcta.

La hipótesis central de esta investigación es que es posible mejorar la calidad de las estructuras aprendidas por los algoritmos basados en independencia me-

diante el uso de un enfoque probabilístico que permita combinar los resultados de diversos tests, para tomar decisiones más acertadas en el aprendizaje de la estructura de independencias, relajando la suposición de que los tests estadísticos son correctos. Más aún, el uso de un enfoque probabilístico otorga un modo de diseñar diversos algoritmos para mejorar significativamente la calidad del aprendizaje de estructuras. Como hipótesis secundaria, se sostiene que además este enfoque puede permitir el diseño de algoritmos eficientes computacionalmente.

A modo de tesis derivada de dichas hipótesis, la contribución del presente trabajo es un enfoque probabilístico basado en independencias llamado *IBMAP* (por sus siglas en inglés, *independence-based maximum-a-posteriori*). IBMAP propone el uso de una función de puntaje de estructuras que utiliza tests estadísticos de independencia, llamada *IB-score* (también del inglés, puntaje basado en independencias). Esta función permite computar de un modo aproximado la probabilidad a posteriori de una estructura dados los datos $Pr(G|D)$, combinando los resultados de un conjunto de tests estadísticos. De este modo, el aprendizaje de estructuras consiste en maximizar el IB-score sobre el espacio de todas las estructuras posibles, asignando a cada estructura un puntaje más alto o más bajo según las probabilidades de las independencias que codifican. A modo de instanciación del enfoque IBMAP se presenta una serie de algoritmos que realizan la maximización con diversos mecanismos de optimización. Entre ellos, la instanciación más competitiva en términos de calidad y eficiencia computacional es un algoritmo que realiza una búsqueda local heurística diseñada para obtener eficientemente estructuras de alta calidad.

1.3. Organización

El resto de este documento está organizado como se describe a continuación. El Capítulo 2 provee información conceptual y teórica de respaldo sobre la representación de las redes de Markov y el aprendizaje de su estructura de independencias. Se explican en detalle los aspectos referentes a la representación de estos modelos. Luego se explica el problema de aprendizaje de redes de Markov desde datos, sus aspectos técnicos, los objetivos que suelen tener los usuarios del aprendizaje de estos modelos, se revisa el problema de aprendizaje paramétrico y el

problema de aprendizaje de la estructura de independencias. Se explican también algunos aspectos teóricos específicos del problema de aprendizaje de estructuras, y se revisan los enfoques que muestra la literatura para resolver el problema.

El Capítulo 3 revisa el estado del arte de los algoritmos basados en independencia, describiendo cómo funciona cada uno de los algoritmos que aparecen en la literatura, explicando el alcance de cada uno en términos de calidad. Finalmente, se hace un análisis del estado del arte, y se destaca cómo los distintos algoritmos recaen en el problema que se ataca en la presente tesis. Los contenidos de los Capítulos 2 y 3 son, en su mayoría, derivados de Schlüter (2012).

El Capítulo 4 describe la tesis de este trabajo: el enfoque IBMAP para aprendizaje robusto de estructuras de independencia de redes de Markov. Se describen los aspectos teóricos del enfoque y las características generales del mismo. Se explica la función IB-score, una función de puntaje basada en independencias para el cómputo aproximado de $\Pr(G \mid D)$. Se describe también el *conjunto de cierre basado en mantas de Markov*, un mecanismo lógico para computar el IB-score eficientemente. Además, se explican diversos métodos de optimización para instanciar el enfoque IBMAP, es decir, una serie de algoritmos que utilizan diferentes técnicas para maximizar el IB-score.

El Capítulo 5 muestra los resultados experimentales obtenidos con el enfoque IBMAP. Se valida el enfoque mediante la evaluación del desempeño de sus distintas instanciaciones, evaluando la calidad de las estructuras aprendidas con respecto a algoritmos basados en independencia que pertenecen al estado del arte. Se presentan resultados sistemáticos sobre una gran variedad de dimensiones del problema, demostrando que maximizar la función IB-score permite mejorar significativamente la calidad de las estructuras aprendidas. Se demuestra también que existe un algoritmo que puede mejorar contundentemente la calidad de los algoritmos del estado del arte a un costo computacional altamente competitivo. Adicionalmente, se muestran experimentos similares sobre conjuntos de datos del mundo real, y se muestran también resultados tras aplicar el enfoque en una aplicación real, donde el aprendizaje de estructuras se utiliza como paso intermedio de algoritmos evolutivos. Los contenidos de los Capítulos 4 y 5 son derivados mayormente de Schlüter et al. (2014).

Finalmente, el Capítulo 6 resume las contribuciones y conclusiones de la tesis, detallando además la investigación complementaria realizada durante el transcurso de esta investigación, y finalizando con una enumeración de las líneas de investigación futura que se han reconocido como prioritarias para extender este trabajo.

Capítulo 2

Marco teórico

Este capítulo provee información conceptual y teórica de respaldo sobre la representación de las redes de Markov y el aprendizaje de su estructura de independencias. Para comenzar, se describen los detalles técnicos de la representación de las redes de Markov. Luego, se explica el aprendizaje de redes de Markov desde datos, sus aspectos técnicos, los objetivos que suelen tener los usuarios del aprendizaje de estos modelos, el problema de aprendizaje de parámetros numéricos, y se describen los enfoques que muestra la literatura para resolver el problema.

2.1. Redes de Markov

En general, los modelos gráficos consisten en dos componentes: uno cualitativo, y otro cuantitativo. Ambos componentes permiten representar una distribución de probabilidades P sobre un dominio de n variables aleatorias, denotado $\mathbf{V} = \{X_0, ..., X_{n-1}\}$. De aquí en adelante, n hará referencia a la cantidad de variables del dominio (el tamaño del problema). El componente cualitativo es la *estructura de independencias* G (también conocida como la *red*, o el *grafo*) del modelo, que representa las independencias condicionales existentes entre las variables del dominio, y define una familia de distribuciones de probabilidad. El componente cuantitativo es un conjunto de parámetros numéricos θ que define una distribución única entre la familia denotada por la estructura, y cuantifica las relaciones existentes en la misma.

2.1.1. La estructura de independencias

La estructura de independencias es una representación compacta de las *independencias condicionales* presentes en la distribución subyacente P. En dicha distribución, dos variables X e Y son independientes condicionalmente dado un conjunto de variables $\mathbf{Z}$ cuando sabiendo el valor de las variables en $\mathbf{Z}$, los valores de Y no son informativos para predecir los valores de X. En este trabajo, denotamos las independencias condicionales como $(X \perp\!\!\!\perp Y|\mathbf{Z})$, y $(X \not\perp\!\!\!\perp Y|\mathbf{Z})$ a las dependencias condicionales.

Comúnmente, la representación de la estructura de independencias G de una red de Markov es un grafo no dirigido con n nodos, cada uno representando a una variable aleatoria del dominio $\mathbf{V}$. Las aristas en el grafo codifican independencias condicionales entre las variables. Cada arista representa una influencia probabilística directa entre dos variables, es decir, ninguna otra variable puede mediar esta influencia. La Figura 2.1 muestra dos ejemplos de estructuras no dirigidas, ambas representando dominios de $n = 12$ variables aleatorias $\mathbf{V} = \{X_0, \ldots, X_{11}\}$. En el ejemplo (a) se muestra un grafo irregular con diferentes grados de conectividad para los distintos nodos. En el ejemplo (b) se muestra una rejilla regular donde las variables pertenecen al dominio de un problema espacial, como se usa típicamente para representar imágenes 2D, o para modelos Ising (que son modelos matemáticos de ferromagnetismo muy utilizados en mecánica estadística).

La estructura es un mapa de las independencias de la distribución que se modela. Dichas independencias pueden leerse del grafo a través de la técnica de *separación de vértices*. La lectura de independencias condicionales se realiza sobre un grafo dado, considerando que cada variable es independiente de todas las variables que no son sus vecinas en el grafo (es decir, adyacentes), condicionando en todas sus variables vecinas. Esto es a lo que se llama la *propiedad local de Markov*. Por ejemplo, en la Figura 2.1 (a), se codifica una independencia condicional entre las variables X_0 y X_3, dado el conjunto de variables $\{X_1, X_2\}$. Otro ejemplo puede verse en la rejilla de la Figura 2.1 (b), donde X_5 es independiente condicionalmente de todas sus variables no vecinas, si condicionamos en sus variables vecinas $\{X_1, X_4, X_6, X_9\}$.

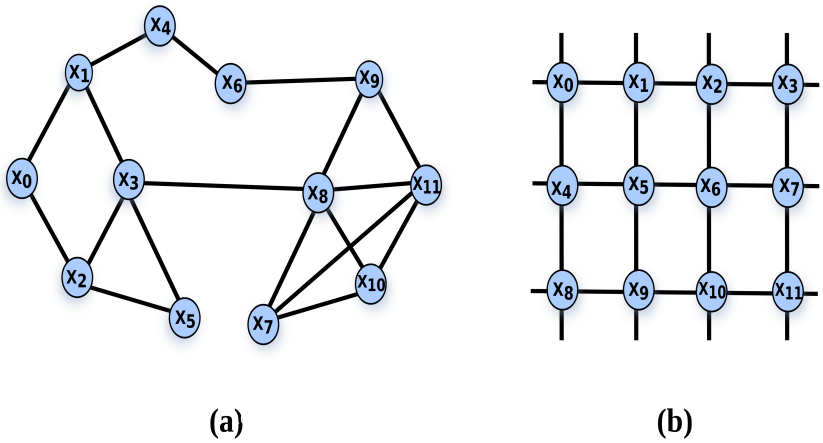

Figura 2.1: Ejemplo de dos estructuras de independencia. (a) Un grafo irregular con diferentes grados de conectividad sobre sus nodos, y (b) una rejilla regular donde las variables pertenecen al dominio de un problema espacial.

2.1.2. Parametrización del modelo

Esta sección explica cómo cuantificar las relaciones codificadas en una estructura de independencias. Aunque este trabajo sólo ataca el problema del aprendizaje de estructuras, los aspectos cuantitativos de las redes de Markov se explican aquí brevemente a fin de proveer un panorama claro y completo a los lectores sobre cómo la estructura influye en la construcción de un modelo completo. Por esto, a continuación se describe un método estándar extraído de Pearl (1988) para factorizar distribuciones a partir de un grafo arbitrario G:

- Identificar los máximos sub-grafos cuyos nodos están todos interconectados entre sí, llamados los *cliques maximales* de G. Por ejemplo, en el grafo en la Figura 2.1 (a) hay un clique maximal de tamaño 4 entre los nodos correspondientes a las variables $\{X_7, X_8, X_{10}, X_{11}\}$, dos cliques maximales de tamaño 3 entre los nodos $\{X_2, X_3, X_5\}$ y $\{X_8, X_9, X_{11}\}$, y el resto de las aristas son cliques maximales de tamaño 2. En la Figura 2.1 (b) todos los cliques son de tamaño 2.

- Para cada clique $\mathbf{c}$ en el conjunto $\mathcal{C}$ de todos los cliques de G, asignar una

función potencial no negativa $g_{\mathbf{c}}(\mathbf{V_c})$ (donde $\mathbf{V_c}$ es el conjunto de variables que corresponden al clique $\mathbf{c}$) midiendo el grado relativo de compatibilidad asociada con cada configuración posible de $\mathbf{V_c}$. Usualmente cada función potencial se representa mediante una tabla con un parámetro numérico asignado a cada asignación completa posible de las variables que componen el clique $\mathbf{c}$. Los valores de estos parámetros no están normalizados.

- Forme el producto $\prod_{\mathbf{c}\in\mathcal{C}} g_{\mathbf{c}}(\mathbf{V_c})$ de las funciones potenciales sobre todos los cliques.

- Construya la distribución conjunta normalizando el producto sobre todos los valores de las posibles combinaciones de las variables del sistema.

$$P(X_0, .., X_{n-1}) = \frac{1}{Z} \prod_{\mathbf{c}\in\mathcal{C}} g_{\mathbf{c}}(\mathbf{V_c}), \tag{2.1}$$

donde Z es aquí la *función de partición*, ó también conocida como *constante de normalización*, computada como:

$$Z = \sum_{X_0,..,X_{n-1}} \prod_{\mathbf{c}\in\mathcal{C}} g_{\mathbf{c}}(\mathbf{V_c}). \tag{2.2}$$

Mediante el conocido teorema de Hammersley-Clifford (Hammersley y Clifford, 1971) puede probarse que la forma general de la distribución de la Ecuación (2.1) incluye todas las independencias condicionales codificadas en el grafo G. Esta forma de las distribuciones presenta algunas dificultades. Primeramente, es complejo discernir el significado de las funciones potenciales. Además, el costo computacional de calcular la función de partición Z requiere una suma exponencial sobre todas las posibles configuraciones de las asignaciones completas de las variables.

2.1.3. Correctitud de la estructura

Para representar correctamente una distribución de probabilidades P mediante una red de Markov, la estructura G debe ser un mapa de las independencias que están presentes en P. Como se prueba formalmente en Pearl (1988), un grafo G

se llama un *mapa de independencias* (ó simplemente *I-map*) de una distribución P cuando todas las independencias codificadas en G existen en P.

Definición 1. *I-map (Pearl, 1988)[p.92]. Un grafo G es un* I-map *de una distribución P si para todos los subconjuntos disjuntos de variables $\mathbf{X}$, $\mathbf{Y}$ y $\mathbf{Z}$, se satisface la siguiente condición:*

$$(\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z})_G \Rightarrow \langle \mathbf{X}, \mathbf{Y}, \mathbf{Z} \rangle_P, \tag{2.3}$$

donde $(\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z})_G$ son las independencias codificadas en G, y $\langle \mathbf{X}, \mathbf{Y}, \mathbf{Z} \rangle_P$ son las independencias existentes en la distribución subyacente P.

Similarmente, G es un mapa de dependencias (D-map) cuando:

$$(\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z})_G \Leftarrow \langle \mathbf{X}, \mathbf{Y}, \mathbf{Z} \rangle_P. \tag{2.4}$$

Cuando se utiliza un grafo G que es un I-map de P se garantiza que los nodos que están separados corresponden a variables independientes, pero no se garantiza que todos los nodos conectados sean dependientes. Contrariamente, cuando G es un D-map se garantiza que todos los nodos conectados en G son dependientes en la distribución P. De esto se desprende que los grafos totalmente conectados (todas las aristas posibles) son I-maps triviales, y los grafos vacíos (sin ninguna arista) son D-maps triviales. Se dice que una distribución P es un *mapa perfecto* de P si es al mismo tiempo un I-map y un D-map.

Otro concepto importante es el de *isomorfismo a grafos*, íntimamente relacionado a la separación de vértices. Básicamente, una distribución es isomorfa a un grafo cuando todas las independencias entre sus variables pueden ser codificadas mediante un grafo no dirigido.

Definición 2. *Isomorfismo a grafos (Pearl, 1988)[p.93].*

Se dice que una distribución es isomorfa a grafos *cuando existe un grafo no dirigido G que es un* mapa perfecto *de P, es decir., para todos los subconjuntos disjuntos $\mathbf{X}$, $\mathbf{Y}$ y $\mathbf{Z}$, se cumple:*

$$(\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z})_G \Longleftrightarrow \langle \mathbf{X}, \mathbf{Y}, \mathbf{Z} \rangle_P. \tag{2.5}$$

Una condición necesaria y suficiente para que una distribución P sea isomorfa a grafos es que $\langle \mathbf{X}, \mathbf{Y}, \mathbf{Z} \rangle_P$ satisfaga los siguientes axiomas de independencia, introducidos por Pearl y Paz (1985):

$$
\begin{aligned}
&\textbf{Simetría} && (\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z}) \Leftrightarrow (\mathbf{Y} \perp\!\!\!\perp \mathbf{X}|\mathbf{Z}) \\
&\textbf{Descomposición} && (\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \cup \mathbf{W}|\mathbf{Z}) \Rightarrow (\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z}) \ \& \ (\mathbf{X} \perp\!\!\!\perp \mathbf{W}|\mathbf{Z}) \\
&\textbf{Transitividad} && (\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z}) \Rightarrow (\mathbf{X} \perp\!\!\!\perp \lambda|\mathbf{Z}) \text{ or } (\lambda \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z}) \\
&\textbf{Unión Fuerte} && (\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z}) \Rightarrow (\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z} \cup \mathbf{W}) \\
&\textbf{Intersección} && (\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z} \cup \mathbf{W}) \ \& \ (\mathbf{X} \perp\!\!\!\perp \mathbf{W}|\mathbf{Z} \cup \mathbf{Y}) \Rightarrow (\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \cup \mathbf{W}|\mathbf{Z}),
\end{aligned}
\tag{2.6}
$$

donde $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$ y $\mathbf{W}$ son todos subconjuntos disjuntos del conjunto de todas las variables del dominio $\mathbf{V}$, y λ representa una variable simple, que no pertenece a $\mathbf{X} \cup \mathbf{Y} \cup \mathbf{Z} \cup \mathbf{W}$. El axioma de intersección es válido sólo para distribuciones de probabilidad estrictamente positivas (lo que significa que todas las configuraciones de la distribución tienen una probabilidad mayor a cero). Esta lista de axiomas representa las relaciones que se satisfacen entre las independencias que están codificadas por la estructura. Hay otro conjunto de axiomas que se cumple para redes de Bayes, pero están fuera del interés de este trabajo.

En resumen, cuando una distribución P es isomorfa a un grafo, existe un grafo no dirigido G que es un mapa perfecto para la distribución P. Para representar una distribución P, puede utilizarse cualquier grafo G que sea un I-map de P, ya que todas las independencias codificadas pertenecen a la distribución, y contribuyen a que el modelo de la distribución sea más compacto. Por esto, mientras más independencias de la distribución sean codificadas por el grafo, mejor será el modelo en términos de su precisión y complejidad. Por último, nótese que asumir isomorfismo a grafos es una decisión realmente importante, ya que no todas las distribuciones reales pueden representarse mediante un grafo no dirigido. Por ejemplo, hay distribuciones que pueden representarse naturalmente por grafos acíclicos dirigidos, y en este caso, las redes de Bayes son el modelo correcto a utilizar.

2.1.4. La manta de Markov de una variable

En esta sección se describe el concepto de *la manta de Markov* de una variable, un concepto teórico central en la representación de distribuciones de probabilidad (Pearl, 1988). La manta de Markov de una variable es el único conocimiento necesario para predecir el comportamiento de dicha variable. Por esto, este concepto tiene una gran relevancia para una amplia variedad de aplicaciones donde las relaciones locales entre algunas variables son realmente significativas.

Definición 3. *Manta de Markov. La manta de Markov de una variable X es un conjunto mínimo de variables, denotado aquí $\mathbf{MB}^X$, condicionado en el cual todos los nodos del dominio $\mathbf{V}$ que no pertenecen a dicho conjunto son independientes de X, es decir:*

$$\forall Y \in \mathbf{V} - \{\mathbf{MB}^X\},\ (X \perp\!\!\!\perp Y | \mathbf{MB}^X). \tag{2.7}$$

Esto significa que la manta de Markov de una variable X es el conjunto más pequeño de variables que protege a X de la influencia probabilística de todas las variables que no pertenecen a dicho conjunto. Desde un punto de vista gráfico, la manta de Markov de X es básicamente el conjunto de todos los nodos vecinos en el grafo (es decir, unidos por una arista).

En el libro de Pearl (1988) se prueba formalmente que para distribuciones estrictamente positivas, la estructura de independencias puede construirse si poseemos la manta de Markov de cada una de las variables del dominio, conectando con una arista todas las variables X e Y cuando X pertenezca a la manta de Markov de Y. También se prueba que cada variable $X \in \mathbf{V}$ en una distribución isomorfa a grafos, tiene una única manta de Markov.

2.2. El aprendizaje de redes de Markov

Esta sección discute las dificultades del problema de aprendizaje de redes de Markov a partir de datos. Esta tarea requiere idealmente que el tamaño del conjunto de datos de entrenamiento D sea suficiente, y que los datos sean, en efecto, un muestreo representativo de la distribución subyacente P. Cuando estas condiciones se satisfacen, es posible aprender un modelo de representación de P explorando y analizando D. El conjunto de datos D utilizado como entrada

contiene información histórica, comúnmente estructurada en *formato tabular*, lo que es estándar en aprendizaje de máquinas. Esto significa que los datos poseen un formato de tabla, con una columna por cada una de las variables aleatorias de la distribución P, y una fila por cada entrada de datos, siendo cada fila una asignación completa de todas las variables. Por ejemplo, una fila de un conjunto de datos con $n = 4$ variables aleatorias binarias $\mathbf{V} = \{X_0, X_1, X_2, X_3\}$ podría ser $(X_0 = 0, X_1 = 1, X_2 = 1, X_3 = 0)$. Generalmente, en la literatura de aprendizaje de redes de Markov se ignora el problema de datos incompletos, ya que esto suele resolverse mediante algoritmos estándar que completan datos, que se utilizan previamente al proceso de aprendizaje de la red de Markov a modo de pre-procesamiento.

2.2.1. Objetivos del aprendizaje

Para evaluar el mérito de un método de aprendizaje de redes de Markov es importante considerar cuál ha sido el objetivo por el cual se está llevando a cabo el aprendizaje. Claramente, el aprendizaje de un modelo completo (estructura y parámetros numéricos) es el método ideal, pero dadas las limitaciones computacionales, espaciales o de requerimiento de datos, puede que esto no sea posible en la práctica. Por esto, usualmente suelen considerarse otros objetivos menos ambiciosos, como los tres objetivos de aprendizaje discutidos en el libro de Koller y Friedman (2009):

- *Estimación de densidad.* Una razón común para aprender una red de Markov es usar esta misma para alguna tarea de inferencia. Cuando se formula el objetivo de aprendizaje como de estimación de densidad, se pretende construir un modelo M de manera que la distribución definida *se acerque* a la distribución solución P. Una métrica común para evaluar la calidad de dicha aproximación es el uso de la *verosimilitud* de los datos dado el modelo, $Pr(D \mid M)$. Como este objetivo asume que se requiere la distribución completa P, resulta intratable para dominios de gran dimensionalidad.

- *Tareas de predicción específicas.* El objetivo es predecir la distribución de un conjunto particular de variables $\mathbf{Y}$, dado otro conjunto disjunto $\mathbf{X}$. Cuando

el modelo se usa sólo para una tarea específica, si nunca se evalúa el modelo para predecir los valores de las variables $\mathbf{X}$, es mejor opción optimizar la tarea de aprendizaje para mejorar la calidad respecto de la predicción de los valores de las variables en el conjunto $\mathbf{Y}$. Éste ha sido el objetivo de una gran parte del trabajo en aprendizaje de máquinas. Por ejemplo, considere el problema de clasificación de documentos para un conjunto dado de palabras, y una variable que etiqueta si el tópico del documento es relevante a las palabras. Otro ejemplo bien conocido es la tarea de segmentación de imágenes, donde el objetivo de la tarea es predecir las etiquetas de clase para todos los píxeles en la imagen, dadas sus características.

- *Descubrimiento de conocimiento.* Cuando se pretende aprender la distribución con este objetivo se intenta aprender la estructura correcta de la distribución solución. Este objetivo es muy común en la práctica, ya que la estructura aprendida puede revelar importantes propiedades del dominio de estudio. Ésta es una motivación muy diferente al aprendizaje de la distribución completa. Un análisis detenido de la estructura aprendida puede mostrar dependencias entre variables, como correlaciones positivas o negativas. Por ejemplo, en el dominio de diagnóstico médico, aprender la estructura del modelo puede ser útil para aprender qué factores son los causantes de ciertas enfermedades, y qué síntomas se asocian con las diferentes enfermedades.

Debido a que la contribución del presente trabajo está orientada al aprendizaje de la estructura de independencias correcta, podría decirse que el objetivo de aprendizaje del enfoque propuesto es descubrimiento de conocimiento. Sin embargo, el enfoque podría utilizarse de igual manera con los demás objetivos de aprendizaje.

2.2.2. Estimación paramétrica

La estimación de parámetros para redes de Markov usualmente se lleva a cabo ajustando el valor de los parámetros a los datos (es decir, aprendiéndolos automáticamente) ya que esta tarea es casi imposible de realizar manualmente, y

los modelos aprendidos desde los datos utilizando validación cruzada usualmente muestran un buen rendimiento. Sin embargo, se ha demostrado que esta tarea es NP-completa (Barahona, 1982).

Para estimar los parámetros, el método más común que se ha propuesto es la *estimación de la máxima verosimilitud* (más conocido en inglés como *maximum-likelihood estimation*), potencialmente utilizando algún método de regularización, como un parámetro a priori adicional. Desafortunadamente, evaluar la verosimilitud de un modelo completo requiere del cómputo de la función de partición Z para cada conjunto de parámetros propuesto durante el proceso de estimación. La función Z se utiliza para normalizar el producto sobre todas las posibles combinaciones de valores de las variables del dominio, como se muestra en la Ecuación (2.2). Aunque no es posible optimizar la máxima verosimilitud de forma exacta, sí se garantiza que el óptimo global puede encontrarse, ya que esta función es cóncava. Por esto, se introducen algunas aproximaciones y heurísticas en Minka (2001); Vishwanathan et al. (2006), para reducir el costo de la estimación de parámetros mediante métodos iterativos, como ascenso por simple gradiente, u otros algoritmos sofisticados de optimización. Desafortunadamente, este problema aún resulta intratable para dominios de gran dimensionalidad, ya que el uso de la función de partición acopla todos los parámetros del modelo, requiriendo métodos iterativos que utilizan inferencia en cada paso.

Se han propuesto varias soluciones aproximadas para disminuir el costo de la estimación de los parámetros. Algunas alternativas tratables son *PL* (del inglés, Pseudo-likelihood) (Besag, 1977), y *SM* (también del inglés, Score Matching) (Hyvärinen y Dayan, 2005). El método de *propagación de creencias iterativa* (más conocido en inglés como loopy belief propagation) propuesto en Pearl (1988) y luego en Yedidia et al. (2005), con algunas variantes introducidas en Wainwright y Jordan (2008), utiliza una técnica de inferencia para aproximar el gradiente de la función de verosimilitud. En Ganapathi et al. (2008) también se presenta otra solución para mejorar el rendimiento del método de propagación de creencias.

Adicionalmente, para evitar el sobre-ajuste, muchos métodos de puntuación requieren del uso de un término de regularización, agregando un hiper-parámetro extra, cuyo mejor valor debe encontrarse empíricamente. Por ejemplo, utilizan-

do validaciones cruzadas puede encontrarse el mejor valor corriendo el paso de entrenamiento para diferentes valores del hiper-parámetro.

2.2.3. Aprendizaje de la estructura de independencias

Los dos enfoques más utilizados para aprender la estructura de independencias de redes de Markov a partir de datos son los enfoques *basados en puntaje* y *basados en independencia* (también conocido como *basado en restricciones*). Ambos enfoques han sido motivados por distintos objetivos de aprendizaje (los descriptos en la Sección 2.2.1). Generalmente, los enfoques basados en puntaje pueden funcionar mejor para tareas donde se requiere inferencia o predicción, es decir, cuando el objetivo de aprendizaje es estimación de densidad. Como se explica luego, en la Sección 2.2.3.1, los métodos basados en puntaje aprenden una red de Markov completa (es decir, aprenden la estructura, y sus parámetros). Existen diversos usos de las redes de Markov para este propósito, incluyendo segmentación de imágenes y otros, donde se requiere ejecutar una tarea de inferencia. En cambio, los métodos basados en independencia son más propicios para los otros objetivos de aprendizaje: tareas de predicción específicas y descubrimiento de conocimiento. Por un lado, los algoritmos basados en independencia comúnmente se utilizan para tareas como selección de categorías para clasificación (más conocido por su nombre en inglés: *feature selection*), ya que es posible hacer un aprendizaje local para un conjunto particular de variables de interés (ver más detalles en la Sección 2.2.3.2). Por otro lado, los algoritmos basados en independencia suelen ser utilizados para tareas de descubrimiento de conocimiento en tareas donde es importante el entendimiento de las interacciones entre las variables del dominio, como por ejemplo bio-informática, genómica, o diagnóstico médico.

2.2.3.1. Enfoque basado en puntaje

Los algoritmos basados en puntaje fueron propuestos para aprender la estructura de independencias de redes de Bayes en los trabajos de Lam y Bacchus (1994) y Heckerman et al. (1995), y luego se propusieron para redes de Markov en los trabajos de Della Pietra et al. (1997) y McCallum (2003). Estos algoritmos enfocan el problema como una optimización sobre el espacio de modelos completos,

haciendo una búsqueda para maximizar una función de puntaje. Los algoritmos tradicionales hacen una búsqueda global sobre el espacio de las estructuras y los parámetros, para aprender un conjunto de "*características*" (en inglés: *features*) que capturan precisamente regiones de alta probabilidad. Una característica es básicamente un conjunto de asignaciones a un subconjunto de las variables del dominio.

El primer algoritmo propuesto para aprendizaje de estructuras de redes de Markov usando el enfoque basado en puntaje es el de Della Pietra et al. (1997). Este algoritmo aprende la estructura generando un conjunto de características desde los datos. Su estrategia se basa en una búsqueda "de arriba hacia abajo", es decir, una búsqueda que va desde lo general hacia lo específico. Este algoritmo comienza con un conjunto de características atómicas (una función por cada una de las variables del dominio). Luego, se crea un conjunto de características candidatas de dos modos. Primero, cada característica que se halla actualmente en el modelo se asocia con cada una de las otras características del modelo. Segundo, cada característica en el modelo se asocia con cada una de las características atómicas. Luego, por razones de eficiencia, los parámetros se aprenden para cada característica candidata, asumiendo que los parámetros de todas las otras características permanecen sin cambios. Cuando se aprenden los parámetros, se utiliza un mecanismo de inferencia basado en muestreo de Gibbs. Luego, por cada característica candidata, el algoritmo evalúa qué tanto se incrementa la verosimilitud al agregar cada característica. Luego de evaluar para todas las características, se agrega aquella que maximiza esta medida. El procedimiento finaliza cuando ninguna característica candidata mejora el puntaje del modelo. Otro algoritmo que utiliza el mismo enfoque se propone en McCallum (2003). Este algoritmo es similar, pero utiliza una función heurística más eficiente para buscar sobre el espacio de estructuras candidatas, induciendo automáticamente cuáles son las características que mejoran la verosimilitud. Sin embargo, como se reporta en Davis y Domingos (2010), estos métodos que van desde lo general hacia lo específico son ineficientes, debido a que buscan diversas variaciones sobre el espacio de búsqueda que no tienen soporte en los datos, y porque además son muy propensos a caer en óptimos locales.

Recientemente, otros enfoques alternativos se han considerado (Lee et al., 2006; Höfling y Tibshirani, 2009; Ravikumar et al., 2010). En éstos enfoques se propone acoplar el aprendizaje paramétrico con el aprendizaje de las características en un solo paso, utilizando regularización-L_1, que fuerza la mayoría de los parámetros numéricos a cero. Estos algoritmos también enfocan el problema como una optimización, otorgando un gran conjunto inicial de características, que contiene a todas las posibles características de interés. Luego, después del aprendizaje, la selección de modelos se lleva a cabo seleccionando aquellas características con parámetros no nulos. Por razones de eficiencia, los enfoques de Höfling y Tibshirani (2009), y Ravikumar et al. (2010) sólo construyen estructuras "pairwise" (redes que para factorizar sólo involucran cliques de tamaño 2 ó 1). En cambio, el algoritmo de Lee et al. (2006), puede aprender arbitrariamente largas funciones potenciales (pese a que solamente muestra experimentación sobre potenciales de tamaño 2).

Un enfoque alternativo reciente se propuso en Davis y Domingos (2010); Lowd y Davis (2014), donde se presenta el algoritmo de aprendizaje "de abajo hacia arriba" para redes de Markov (BLM, del inglés *Bottom-up Learning of Markov Networks*). BLM comienza con características grandes, una por cada punto de datos presente en el conjunto de datos de entrenamiento, generalizando cada una de éstas para que coincidan con sus k-vecinas más cercanas, removiendo variables. Cuando una característica nueva generalizada mejora el puntaje del modelo, se incorpora al mismo. El bucle finaliza cuando ninguna generalización puede mejorar el puntaje.

Uno de los enfoques presentado más recientemente en Van Haaren y Davis (2012) es el algoritmo de aprendizaje de estructuras mediante generación y selección (GSSL, del inglés *Generate Select Structure Learning*) , que combina aspectos de ambos enfoques. Básicamente, en la fase de generación de características, el algoritmo procede con un enfoque "de abajo hacia arriba" para explorar el espacio de las características candidatas, generando las características iniciales desde los datos (al igual que BLM). Luego, en una fase de selección de características, se lleva a cabo el aprendizaje paramétrico sólo una vez, a fin de seleccionar las mejores características, y luego se sigue la filosofía de los enfoques basados en

modelos locales, intentando minimizar el costo computacional del aprendizaje de parámetros.

2.2.3.2. Enfoque basado en independencia

En líneas generales, los algoritmos basados en independencia consisten en una búsqueda llevada a cabo mediante la evaluación iterada de *tests estadísticos de independencia condicional* entre distintas consultas de independencia sobre los datos. Cada nueva independencia encontrada en los datos puede ser inconsistente con una o más estructuras, las cuales directamente se descartan como candidatas. El algoritmo prosigue hasta que queda una sola estructura consistente con los tests realizados. La existencia de una única estructura consistente puede demostrarse bajo suposiciones (Spirtes et al., 2000). También suele llamarse a estos algoritmos como *basados en restricciones*, ya que la búsqueda que realizan puede ser vista como un problema de satisfacción de restricciones, donde las independencias aprendidas por los tests estadísticos desde el conjunto de datos de entrada son las restricciones, y el objetivo es encontrar una estructura que codifique todas las independencias presentes en los datos.

Cada test de independencia consulta los datos para responder una consulta respecto de la independencia condicional entre un par de variables aleatorias X e Y, resultando en una aserción de independencia condicional ($X \perp\!\!\!\perp Y|\mathbf{Z}$), ó en una aserción de dependencia condicional ($X \not\perp\!\!\!\perp Y|\mathbf{Z}$). El costo computacional de cada test estadístico es proporcional al número de filas en el conjunto de datos de entrada D, y al número de variables involucradas en el test. Algunos ejemplos de tests estadísticos utilizados comúnmente en la práctica son el test de Información Mutua (Cover y Thomas, 1991), los tests de Pearson χ^2 y G^2 (Agresti, 2002), el test Bayesiano (Margaritis, 2005), y para datos Gaussianos continuos el *test de correlación parcial* (Spirtes et al., 2000). Dichos tests de independencia computan un valor estadístico para un triplete de variables $\langle X, Y, \mathbf{Z}\rangle$, dado un conjunto de datos de entrada D, y deciden independencia o dependencia comparando este valor con un umbral. Por ejemplo, χ^2 y G^2 usan el *p-value*, que se computa como la probabilidad de que la hipótesis nula sea cierta (es decir, que las variables sean independientes). La hipótesis nula se rechaza cuando el p-value es menor que

$1 - \alpha$, que usualmente es 0.05 ó 0.01. Cuando el resultado es estadísticamente significativo, la hipótesis nula se acepta.

La mayoría de los algoritmos que usan estos tests para aprender estructuras de independencias utilizan la estrategia *LGL* (local-to-global). Esta estrategia es presentada como un mecanismo elegante, eficiente y escalable en Aliferis et al. (2010b). Este trabajo consiste en una generalización de varios algoritmos previos que utilizan esta misma estrategia. El Algoritmo 1 muestra el pseudo-código de este procedimiento, que se destaca por ser sencillo y teóricamente correcto. Este pseudo-código omite un último paso de orientación de aristas, utilizado en el aprendizaje de redes de Bayes.

Algoritmo 1 Estrategia LGL para redes de Markov

1: Aprender $\mathbf{MB}^{X_i}$ para toda variable $X_i \in \mathbf{V}$.
2: Reconstruir la estructura global usando "regla OR".

Dicha estrategia sugiere aprender la estructura de independencias dividiendo el problema en n problemas de aprendizaje de manta de Markov independientes, es decir, se aprende la manta de Markov para cada variable del dominio $\mathbf{V}$. El aprendizaje de la manta de Markov está generalizado por Aliferis et al. (2010a) para el aprendizaje de redes de Bayes, en el marco de trabajo de aprendizaje local generalizado (GLL, del inglés *Generalized Local Learning*). Los algoritmos que utilizan la estrategia GLL aprenden localmente la manta de Markov de todas las variables del dominio, y luego construyen la estructura global enlazando cada una de las variables con cada miembro de su manta de Markov, utilizando una "regla OR" (que dice que una arista existe entre dos variables X e Y cuando $X \in \mathbf{MB}^Y$ ó cuando $Y \in \mathbf{MB}^X$).

Para el aprendizaje de la estructura de redes de Bayes, los algoritmos basados en independencia aparecieron en 1993, cuando Spirtes et al. (2000) publicó los conocidos algoritmos SGS y PC, en la primer edición de su libro. Luego, otros algoritmos basados en independencia aparecieron en trabajos de selección de características mediante el aprendizaje de la manta de Markov, y también en trabajos sobre aprendizaje de redes de Bayes y redes de Markov. Por esto, surgieron una serie de algoritmos basados en aprendizaje de la manta de Markov para aprender

redes de Bayes, como el algoritmo KS (Koller-Sahami) (Koller y Sahami, 1996), el algoritmo GS (Grow-Shrink) (Margaritis y Thrun, 2000), el algoritmo IAMB (Incremental Association Markov Blanket) (Tsamardinos et al., 2003), el algoritmo MMPC/MB (Max-Min Parents and Children Markov Blanket) (Tsamardinos et al., 2006), el algoritmo HITON-PC (Aliferis et al., 2003), el algoritmo Fast-IAMB (Yaramakala y Margaritis, 2005), el algoritmo PCMB (Parent-Children Markov Blanket) (Peña et al., 2007) y el algoritmo IPC-MB (Iterative Parent and Children Markov Blanket) (Fu y Desmarais, 2008).

Para aprendizaje de la estructura de redes de Markov, los algoritmos basados en independencia aparecieron luego, en 2006, cuando Bromberg et al. (2006, 2009) publicó el algoritmo GSMN (Grow-Shrink Markov Network) y el algoritmo GSIMN (Grow-Shrink Inference-based Markov Network). Posteriormente se propusieron PFMN (Particle Filter Markov Network) (Bromberg y Margaritis, 2007; Margaritis y Bromberg, 2009), y DGSIMN (Dynamic Grow Shrink Inference-based Markov Network) (Gandhi et al., 2008). Otro enfoque que se propone aparece en Bromberg y Margaritis (2009), como un marco de trabajo basado en argumentación para mejorar la credibilidad de los tests de independencia. Estos algoritmos se explican en detalle en el Capítulo 3.

En general, los algoritmos basados en independencia poseen varias ventajas. Primero, pueden aprender la estructura sin intercalar la costosa tarea de aprender los parámetros numéricos (contrariamente a la mayoría de los algoritmos basados en puntaje explicados en la Sección 2.2.3.1), obteniendo en algunos casos complejidades polinomiales en el número de tests estadísticos requeridos. Si se requiere el modelo completo, los parámetros sólo deben estimarse una única vez para la estructura aprendida. Otra ventaja importante es que estos algoritmos son teóricamente correctos, es decir, cuando los tests estadísticos son correctos, la estructura aprendida representa correctamente la distribución subyacente. Sin embargo, estos algoritmos sólo son correctos cuando se cumplen las siguientes suposiciones:

i) la distribución de los datos es *isomorfa a grafos*;

ii) la distribución subyacente es estrictamente positiva;

iii) los tests estadísticos son correctos.

La tercer condición es realmente un problema importante de los algoritmos basados en independencia. Cuando un conjunto de datos utilizado para aprendizaje no es lo suficientemente grande, o sino en el caso de que éste no es un muestreo representativo de la distribución, las respuestas de los tests estadísticos es incorrecta, y por lo tanto las estructuras aprendidas contienen gran cantidad de errores. Este problema de que los tests no son confiables se empeora exponencialmente con el número de variables involucradas (para un tamaño fijo del conjunto de datos). Para obtener una calidad aceptable, los tests estadísticos requieren cantidades suficientes de datos en sus tablas de contingencias. Por ejemplo, Cochran (1954) recomienda que el test χ^2 no debe considerarse confiable cuando menos del 20 % de sus celdas tienen una cantidad esperada menor a 5 puntos de datos. Ésta es una de las causas de que la escasez de datos sea un problema frecuente en los algoritmos basados en independencia, ya que las tablas de contingencias poseen una cantidad de celdas exponencial en la cantidad de variables que se involucran en el test estadístico, y en el tamaño del dominio de dichas variables.

Capítulo 3

Análisis del estado del arte

Este capítulo revisa los algoritmos basados en independencia para aprendizaje de estructuras de redes de Markov que han aparecido en la literatura. En esta revisión se ordena cronológicamente los algoritmos según fueron publicados, y sobre el final se presenta un análisis global sobre cómo los mismos atacan el problema de interés en esta tesis: la calidad de las estructuras aprendidas.

3.1. El algoritmo GSMN

GSMN (del inglés, *Grow-Shrink Markov Network*) se presenta en Bromberg et al. (2009), como el primer algoritmo basado en independencia para aprendizaje de redes de Markov. Este algoritmo es una adaptación para redes de Markov del algoritmo GS (Margaritis y Thrun, 2000), diseñado para el aprendizaje de la manta de Markov de redes de Bayes. GSMN aprende la estructura global utilizando la estrategia LGL (ver el Algoritmo 1), aprendiendo la manta de Markov de cada variable con GS (ver el Algoritmo 2).

GS mantiene un conjunto llamado **S**, inicializado como vacío en la línea 1. En la línea 2 se ordenan todas las variables del dominio por asociación con X, de mayor a menor. Esto se lleva a cabo utilizando un test incondicional entre X y cada variable $Y \in \mathbf{V} - \{X\}$. Luego, el algoritmo prosigue en dos fases, la fase de *crecimiento*, y la fase de *encogimiento*, utilizando este ordenamiento. Durante la fase de crecimiento (línea 4) el algoritmo va agregando al conjunto **S**

Algoritmo 2 GS($X, \mathbf{V}$).

1: $\mathbf{S} \longleftarrow \emptyset$.
2: ordenar $\mathbf{V} - \{X\}$ según asociación con X
3: /* Fase de crecimiento */
4: **mientras** $\exists Y \in \mathbf{V} - \{X\}$ tal que $(Y \not\perp\!\!\!\perp X | \mathbf{S})$, **hacer** $\mathbf{S} \leftarrow \mathbf{S} \cup \{Y\}$.
5: /* Fase de encogimiento */
6: **mientras** $\exists Y \in \mathbf{S}$ tal que $(Y \perp\!\!\!\perp X | \mathbf{S} - \{Y\})$, **hacer** $\mathbf{S} \leftarrow \mathbf{S} - \{Y\}$.
7: **retornar S**

cada variable Y que sea encontrada dependiente de X, condicionando en el estado actual del conjunto $\mathbf{S}$. Para cuando esta fase finaliza, el conjunto $\mathbf{S}$ contiene todos los miembros de la manta de Markov, pero incluye potencialmente algunos falsos positivos que no son miembros de la manta de Markov, debido a la heurística de ordenamiento. Posteriormente estos falsos positivos son removidos en la fase de encogimiento (línea 6), donde las variables Y que se encuentran independientes de X condicionando en $\mathbf{S} - \{Y\}$ se remueven de $\mathbf{S}$. Hacia el final, GS retorna el conjunto $\mathbf{S}$, que contiene la manta de Markov de X. Luego, la estructura que retorna GSMN se construye agregando una arista entre cada variable, y cada una de las variables de su manta de Markov (regla "OR").

Las ventajas principales de GSMN son *i)* este algoritmo es sólido, y *ii)* es eficiente. La solidez de GSMN se prueba teóricamente por sus autores, garantizando que se encuentra la estructura correcta cuando los tests estadísticos no cometen ningún error. Además, este algoritmo es eficiente porque ejecuta un número de tests estadísticos que es polinomial en el tamaño del dominio $\mathbf{V}$, siendo que cada test estadístico además tiene un costo polinomial en la cantidad de variables involucradas. Una desventaja de utilizar GS es que cuando los tests estadísticos no son confiables, se producen errores en cascada, no sólo generándose decisiones de independencias erróneas, sino que también se acumulan estos errores en las fases de crecimiento y encogimiento (Spirtes et al., 2000).

Existe un algoritmo que se diseñó posteriormente a GS, introduciendo algunas modificaciones simples para mejorar el aprendizaje de la manta de Markov: el algoritmo HITON-PC (Aliferis et al., 2003). Se ha probado empíricamente que

este algoritmo es más robusto que GS a errores en los tests, introduciendo dos variantes simples. Por un lado se intercala el paso de ordenamiento con la fase de crecimiento, es decir, se intercala las líneas 2 y 4 del Algoritmo 2. Con esta modificación se logra maximizar la precisión, reduciendo el número de falsos positivos que se cometen en la fase de crecimiento. Por otro lado, se introduce una modificación en el criterio utilizado para testear independencias, en la fase de encogimiento. Cuando se testea independencia, en la fase de encogimiento, en vez de sólo condicionar en su manta de Markov tentativa $\mathbf{S}$, HITON-PC testea independencia condicionando con cada uno de los subconjuntos de $\mathbf{S}$ (es decir, cada conjunto $\mathbf{Z} \subseteq \mathbf{S} - \{Y\}$). Como los tests estadísticos son más confiables mientras menos variables poseen, esta modificación mejora la confiabilidad de los tests estadísticos. Una desventaja del enfoque propuesto por HITON-PC es su costo exponencial en el tamaño de $\mathbf{S}$. En el Apéndice C se presenta un algoritmo llamado en este trabajo HHC-MN, presentado como una adaptación de HITON-PC para aprendizaje de redes de Markov, y que se utiliza luego en los experimentos del Capítulo 5 como algoritmo competidor.

3.2. El algoritmo GSIMN

El algoritmo GSIMN (del inglés, *Grow Shrink Inference Markov Network*) se presentó junto a GSMN en Bromberg et al. (2009). Este algoritmo trabaja de un modo similar a GSMN, utilizando la estrategia LGL del Algoritmo 1, y aprendiendo la manta de Markov de todas las variables con el algoritmo GS, pero se intercala un paso de inferencia para reducir la cantidad de tests estadísticos requeridos para aprender la manta de Markov. Utilizando un teorema llamado por sus autores "el teorema del triángulo", GSIMN reduce la cantidad de tests estadísticos ejecutados sobre los datos, sin afectar adversamente la calidad de las estructuras aprendidas. Los autores dicen que esto resulta útil para problemas donde los tests estadísticos son muy costosos, como cuando existe gran cantidad de datos, o cuando se ejecutan en dominios distribuidos.

El teorema del triángulo está basado en los axiomas de Pearl descriptos en la Sección 2.1.3. Este teorema permite inferir independencias desconocidas a partir de otras aserciones de independencia que ya se han aprendido.

Teorema del triángulo. Dadas las propiedades que se muestran en la Ecuación (2.6), para todas las variables X, Y, W y los conjuntos $\mathbf{Z}_1$ y $\mathbf{Z}_2$ de modo que $\{X, Y, W\} \cap \mathbf{Z}_1 = \{X, Y, W\} \cap \mathbf{Z}_2 = \varnothing$, se cumple que:

$$\begin{aligned} (X \not\perp\!\!\!\perp W|\mathbf{Z}_1) \wedge (W \not\perp\!\!\!\perp Y|\mathbf{Z}_2) &\Rightarrow (X \not\perp\!\!\!\perp Y|\mathbf{Z}_1 \cap \mathbf{Z}_2) \\ (X \perp\!\!\!\perp W|\mathbf{Z}_1) \wedge (W \not\perp\!\!\!\perp Y|\mathbf{Z}_1 \cup \mathbf{Z}_2) &\Rightarrow (X \perp\!\!\!\perp Y|\mathbf{Z}_1). \end{aligned}$$

Se llama "regla D-triángulo" a la primer relación, y "regla I-triángulo" a la segunda relación.

Cuando GSIMN consulta independencias sobre los datos, primero aplica este teorema sobre una base de conocimiento que contiene los tests que se han computado previamente, a fin de chequear si esta aserción de independencia puede inferirse lógicamente y la ejecución del test puede evitarse. Si el resultado del test no puede inferirse, éste se ejecuta normalmente sobre los datos, y luego su resultado es almacenado en la base de conocimiento. Por conveniencia, el algoritmo determina el orden de visita (el orden del aprendizaje de la manta de Markov) de modo de maximizar la eficacia de la inferencia. Los resultados obtenidos con GSIMN muestran mejoras de un 40 % en los tiempos de corrida de GSMN, obteniendo calidades comparables a las de GSMN.

3.3. El algoritmo PFMN

El algoritmo PFMN (del inglés, *Particle Filter Markov networks*) se presentó en Margaritis y Bromberg (2009), como un enfoque basado en independencia novedoso para aprendizaje de estructura de redes de Markov. Los algoritmos basados en independencia presentados previamente, como GSMN y GSIMN, usan la estrategia LGL. En cambio, este algoritmo directamente aprende una estructura global como la solución.

PFMN se diseñó para mejorar la eficiencia del algoritmo GSIMN. Este algoritmo trabaja ejecutando iterativamente tests estadísticos de independencia, seleccionando vorazmente en cada iteración qué test estadístico elimina el mayor número de estructuras inconsistentes. Esta decisión se toma modelando primeramente el problema de aprendizaje con un enfoque Bayesiano, seleccionando como

solución una estructura G que maximiza su probabilidad a posteriori $\Pr(G \mid D)$. Como el cómputo directo de esta probabilidad es intratable, PFMN propone un modelo generativo con tests de independencia que es una aproximación de la probabilidad a posteriori. Según los autores, es posible demostrar que bajo la suposición de que los tests estadísticos son correctos, la distribución de $\Pr(G \mid D)$ converge a la estructura correcta.

Al igual que GSIMN, este algoritmo es útil en dominios donde los tests son muy costosos, ya que los resultados reportados para PFMN muestran mejoras en los tiempos de corrida de hasta un 90 % respecto de GSIMN, con calidades estructurales comparables a las obtenidas por GSIMN y GSMN.

3.4. El algoritmo DGSIMN

DGSIMN (del inglés, *Dynamic Grow Shrink Inference-based Markov Network*) se presenta en Gandhi et al. (2008). Se trata de una extensión de GSIMN que también utiliza el teorema del triángulo para evitar la ejecución de tests innecesarios. Adicionalmente, se propone un método de ordenamiento de las variables alternativo. El esquema de DGSIMN es similar al de GSMN y GSIMN, utilizando la estrategia LGL del Algoritmo 1, y utilizando también el algoritmo GS mostrado en el Algoritmo 2 para aprender la manta de Markov de las variables. La diferencia está en que intercala un paso de inferencia diferente al de GSIMN, con el fin de reducir aún más la cantidad de tests a ejecutar. DGSIMN mejora a GSIMN seleccionando dinámicamente los tests que mejoran el estado de conocimiento sobre la estructura, estimando el número de independencias inferidas que se obtendrían luego de ejecutar un test, y seleccionando aquél que maximiza el número de inferencias.

Los resultados de los experimentos con DGSIMN muestran que se mejora el ordenamiento de las variables, y que los tiempos de corrida de GSIMN son mejorados hasta en un 85 %, obteniendo calidades comparables a GSMN.

3.5. El test argumentativo de independencia

Los algoritmos presentados previamente son algoritmos basados en independencia que hacen foco en mejorar la eficiencia, ignorando los importantes problemas de calidad que aparecen cuando los tests estadísticos no son confiables.

Un enfoque basado en independencia para atacar este problema se presenta en Bromberg y Margaritis (2009), modelando el problema de la baja credibilidad en los tests estadísticos como una base de conocimientos con aserciones de independencia que pueden contener errores, y con los axiomas de Pearl (descriptos en la Sección 2.1.3). La ventaja de este enfoque es su poder para corregir errores en los tests, explotando lógicamente los axiomas de independencia de Pearl. Cuando existen aserciones de independencia que están en conflicto en la base de conocimientos, este enfoque propone resolver los conflictos a través del uso de *argumentación*, que es una lógica rebatible propuesta por Amgoud y Cayrol (2002) para razonar en condiciones de incertidumbre.

Este enfoque se presentó como un test estadístico más robusto llamado *test de independencia argumentativo*, tanto para aprendizaje de redes de Bayes como para redes de Markov. La evaluación experimental muestra mejoras significativas en la precisión del test de independencia argumentativo, respecto de otros tests estadísticos (hasta un 13 %), y mejoras en la precisión del aprendizaje de la manta de Markov (hasta un 20 %). Una desventaja de este enfoque es que, como se trata de un formalismo proposicional, se requiere proposicionalizar el conjunto de las reglas de Pearl, que están en primer orden. Como estas reglas son superconjuntos y subconjuntos de variables, esta proposicionalización requiere la generación de un número de proposiciones de tamaño exponencial. En este trabajo también se presenta una solución aproximada con tiempo de ejecución polinomial, que mejora la calidad en la evaluación experimental (hasta un 9 %), pero haciendo una aproximación drástica que no provee garantías teóricas.

3.6. Conclusiones

Este capítulo revisa los algoritmos basados en independencia para aprendizaje de redes de Markov. Estos algoritmos permiten aprender la estructura de

independencias eficientemente, teniendo la importante ventaja de que permiten aprender la estructura correcta de la distribución subyacente cuando los datos son un muestreo representativo de una red de Markov, los tests estadísticos son confiables, y la distribución subyacente es estrictamente positiva. Esta ventaja teórica es realmente importante, ya que la calidad del aprendizaje de la estructura tiene gran impacto en la calidad de la distribución aprendida tras aprender los parámetros numéricos. Sin embargo, una fuente importante de error en los algoritmos basados en independencia se produce cuando los tests estadísticos arrojan salidas incorrectas, produciendo errores en cascada. Esto suele ser un caso muy común en la práctica, ya que los tests estadísticos requieren de cantidades exponenciales de datos en la cantidad de variables que se involucran en los tests estadísticos, y en el tamaño del dominio de dichas variables.

Los algoritmos revisados son GSMN, GSIMN, PFMN y DGSIMN. También aparece relacionado al aprendizaje de estructuras el test de independencia argumentativo, un enfoque para mejorar la calidad de los tests estadísticos. En la Tabla 3.1 se muestra un resumen de los aspectos más importantes de dichas soluciones. Un problema de gran relevancia que poseen los algoritmos GSMN, GSIMN, DGSIMN y PFMN es la inestabilidad ante incorrectitud de los tests estadísticos, ya que todos asumen que éstos son confiables. No obstante, la situación de insuficiencia de datos para los tests estadísticos es un caso muy común en la práctica, ya que éstos requieren de cantidades exponenciales de datos en la cantidad de variables que se involucran en cada consulta de independencia. El único enfoque presente en la literatura que ataca este problema en realidad intenta mejorar la calidad de los tests estadísticos en sí, presentando el test argumentativo de independencia. Sin embargo, aunque los resultados experimentales usando este enfoque muestran mejoras significativas en la eficacia del test de independencia, el algoritmo exacto presentado tiene un costo exponencial en el número de variables involucradas, y el algoritmo aproximado sólo hace una aproximación drástica que no provee ninguna garantía teórica.

Tabla 3.1: Resumen de algoritmos basados en independencia para redes de Markov

Nombre	Referencia	Comentarios
GSMN	(Bromberg et al., 2006)	▪ Sólido teóricamente, bajo suposiciones ▪ Primer algoritmo basado en independencia para redes de Markov ▪ Usa estrategia LGL ▪ Ejecuta un número polinomial de tests en n ▪ La calidad depende de la complejidad muestreal de los tests
GSIMN	(Bromberg et al., 2006).	▪ Sólido teóricamente, bajo suposiciones ▪ Usa estrategia LGL ▪ Usa teorema del triángulo para reducir cantidad de tests a ejecutar ▪ Útil cuando se usan conjuntos de datos con muchas filas, o dominios distribuídos ▪ Mejoras de hasta un 40 % en tiempo de corrida respecto a GSMN ▪ Calidad comparable a GSMN
PFMN	(Bromberg y Margaritis, 2007)	▪ Sólido teóricamente, bajo suposiciones ▪ No usa estrategia LGL ▪ Diseñado para mejorar la eficiencia de GSIMN ▪ Usa un método aproximado para computar la posterior $\Pr(G \mid D)$ usando tests estadísticos ▪ Útil cuando se usan conjuntos de datos con muchas filas, o dominios distribuídos ▪ Mejoras de hasta un 90 % en tiempo de corrida respecto a GSIMN ▪ Calidad comparable a GSMN
DGSIMN	(Gandhi et al., 2008)	▪ Sólido teóricamente, bajo suposiciones ▪ Usa estrategia LGL ▪ Diseñado para mejorar la eficiencia de GSIMN ▪ Usa ordenamiento dinámico para reducir cantidad de tests a ejecutar ▪ Útil cuando se usan conjuntos de datos con muchas filas, o dominios distribuídos ▪ Mejoras de hasta un 85 % en tiempo de corrida respecto a GSIMN ▪ Calidad comparable a GSMN
Test argumentativo	(Bromberg y Margaritis, 2009)	▪ Usa argumentación para corregir errores cuando los tests son incorrectos ▪ Usa una base de conocimientos de independencias. Las inconsistencias y los axiomas se usan para detectar errores en los tests ▪ Diseñado para aprender redes de Bayes y redes de Markov ▪ El algoritmo exacto presentado es exponencial (mejoras de precisión de un 13 %) ▪ El algoritmo aproximado es simplista, y no provee garantías teóricas (mejoras de precisión de un 9 %)

Capítulo 4

El enfoque IBMAP

Este capítulo describe la contribución principal de la presente tesis: el enfoque de máximo a posteriori basado en independencias (IBMAP, del inglés *independence-based maximum a posteriori*) para aprendizaje de estructuras de redes de Markov (Schlüter et al., 2014; Bromberg et al., 2011). La idea central de IBMAP es modelar la tarea del aprendizaje de la estructura de independencias como un problema de maximización de la probabilidad a posteriori (MAP), computando la probabilidad a posteriori de cada estructura posible, dados los datos. Esta contribución se sustenta en la conjetura de que la probabilidad a posteriori de una estructura (es decir, dados los datos) es una medida representativa de su calidad. Formalmente, la intención del enfoque puede resumirse en la siguiente expresión:

$$G^\star = \arg\max_G \Pr(G \mid D). \tag{4.1}$$

Sin embargo, desarrollar esta maximización no es trivial. Para esto, en la siguiente sección se desarrolla una función de puntaje basada en independencias para el cómputo aproximado de $\Pr(G \mid D)$. Luego, en la Sección 4.2 se describe un mecanismo lógico para computar IB-score eficientemente, utilizando un número de tests estadísticos cuadrático en la cantidad de variables del dominio. Posteriormente, la Sección 4.3 enumera una serie de métodos de optimización para realizar la maximización de la Ecuación (4.1), resultando en una serie de algoritmos que utilizan técnicas diferentes para maximizar el IB-score.

4.1. Una función de puntaje de estructuras basada en independencias

En esta sección se describe la función IB-score (del inglés, *independence-based score*), que aproxima el cómputo de $\Pr(G \mid D)$ con el objeto de hallar modos computables para la maximización de la Ecuación (4.1). En este trabajo se propone realizar dicha aproximación mediante la combinación de los resultados de un conjunto de tests estadísticos de independencia que determinan exactamente una estructura G. Llamamos a este conjunto de tests el *cierre* de la estructura. Formalmente:

Definición 4 (Cierre de una estructura)**.** *Sea G una estructura de independencias no dirigida, y sea $P(\mathbf{V})$ una distribución de probabilidad conjunta entre las variables del dominio $\mathbf{V}$. El conjunto de* cierre *de G es un conjunto de aserciones de dependencia ó independencia condicional $\mathcal{C}(G) = \{c_i\}$ que es necesario y suficiente para determinar completamente la estructura G.*

Mediante el uso del conjunto de cierre es posible expresar la posterior de cada estructura como una distribución de probabilidad conjunta sobre el conjunto de aserciones de independencia que determinan a dicha estructura. Por esto, es posible reformular la Ecuación (4.1), reemplazando G por su conjunto de cierre correspondiente $\mathcal{C}(G)$, obteniendo:

$$G^\star = \arg\max_G \Pr(\mathcal{C}(G) \mid D). \tag{4.2}$$

La aplicación de la regla de la cadena sobre las aserciones del cierre $\mathcal{C}(G)$ permite obtener la siguiente expresión:

$$\Pr(\mathcal{C}(G) \mid D) = \prod_{c_i \in \mathcal{C}(G)} \Pr(c_i | c_1, \ldots, c_{i-1}, D). \tag{4.3}$$

La función IB-score se diseñó como una aproximación de la Ecuación (4.3), debido a que actualmente se desconoce la existencia de métodos estadísticos para computar las probabilidades $\Pr(c_i|c_1, \ldots, c_{i-1}, D)$, es decir, la probabilidad de aserciones de independencia condicionadas en otras aserciones de independencia y los datos. Por ello, la aproximación consta de asumir que todas las aserciones

de independencia que pertenecen al conjunto de cierre son *mutuamente independientes*. Esta suposición se lleva a cabo implícitamente por todos los algoritmos de aprendizaje de estructuras de redes de Markov revisados en el Capítulo 3, ya que los tests estadísticos se utilizan como una caja negra, sólo utilizando los datos para decidir independencia por cada aserción de independencias c_i. La aplicación de dicha aproximación resulta en la siguiente expresión:

$$\Pr(\mathcal{C}(G) \mid D) \approx \prod_{c_i \in \mathcal{C}(G)} \Pr(c_i \mid D),$$

que expresada en términos de logaritmos para evitar desbordamiento aritmético, resulta en la siguiente fórmula para el IB-score:

$$\text{IB-score}(G) = \sum_{c_i \in \mathcal{C}(G)} \log \Pr(c_i \mid D). \tag{4.4}$$

En esta ecuación, las probabilidades a posteriori de cada uno de los términos del conjunto de cierre pueden computarse utilizando un test estadístico específicamente diseñado para computar tal probabilidad, el test Bayesiano de independencia condicional (Margaritis, 2005; Margaritis y Bromberg, 2009). En el Apéndice A se explica en detalle el funcionamiento de dicho test, cómo se computan sus estadísticas internas, y se presenta un pseudo-código detallando cómo implementar el mismo.

Concluyendo con la formulación matemática del IB-score, la maximización del enfoque IBMAP mostrada en la Ecuación (4.1) se puede re-expresar del siguiente modo:

$$G^\star \approx \arg\max_G \ \text{IB-score}(G). \tag{4.5}$$

Es importante considerar el hecho de que esta expresión resulta intratable para dominios de gran dimensionalidad, ya que el número de estructuras no dirigidas posibles crece acorde a la cantidad de variables n, es decir, existen $2^{\binom{n}{2}}$ grafos no dirigidos posibles con n nodos. Además, la complejidad total de la maximización del IB-score depende también del tamaño del conjunto de cierre utilizado, que determina la cantidad de tests estadísticos que se requiere ejecutar para computar el puntaje de cada estructura candidata. Por esto, la siguiente sección describe

en detalle el *conjunto de cierre basado en mantas de Markov*, un mecanismo lógico para determinar una estructura utilizando un número reducido de tests estadísticos de independencias.

4.2. El conjunto de cierre basado en mantas de Markov

El *conjunto de cierre basado en mantas de Markov* es un conjunto de aserciones de independencia que sigue directamente de la Definición 4. Este conjunto ha sido diseñado utilizando el concepto de *manta de Markov* (ver la Sección 2.1.4), con la intención de explotar las independencias locales a cada una de las variables. Por esto, el mismo es definido como la unión del conjunto de cierre local a cada variable, que puede computarse independientemente debido a que cada estructura de n variables se descompone en sus n mantas de Markov. Formalmente:

Definición 5 (Conjunto de cierre basado en mantas de Markov)**.** *El* conjunto de cierre basado en mantas de Markov *de una estructura G es un conjunto de aserciones de independencia que está determinado por la unión de n conjuntos disjuntos $\mathcal{C}_X(G)$ de aserciones de independencia ó dependencia, uno por cada variable X del dominio $\mathbf{V}$, es decir,*

$$\mathcal{C}(G) = \bigcup_{X \in \mathbf{V}} \mathcal{C}_X(G), \tag{4.6}$$

donde cada $\mathcal{C}_X(G)$ es la unión de dos conjuntos de aserciones mutuamente exclusivos:

$$\begin{aligned} \mathcal{C}_X(G) = \; & \left\{ \; (X \not\perp\!\!\!\perp Y | \mathbf{MB}^X \setminus \{Y\}) \; : \; Y \in \mathbf{MB}^X \right\} \cup \\ & \left\{ \; (X \perp\!\!\!\perp Y | \mathbf{MB}^X) \; : \; Y \notin \mathbf{MB}^X \right\}. \end{aligned} \tag{4.7}$$

Para cada vecino de X ($Y \in \mathbf{MB}^X$) se agrega a $\mathcal{C}_X(G)$ una aserción de dependencia condicional entre las dos variables, condicionando en $\mathbf{MB}^X \setminus \{Y\}$; y para cada variable no vecina de X ($Y \notin \mathbf{MB}^X$) se agrega a $\mathcal{C}_X(G)$ una aserción de independencia condicional entre las dos variables, condicionando en $\mathbf{MB}^X$.

El siguiente teorema define que un conjunto de cierre basado en mantas de Markov es de hecho un conjunto de cierre, es decir, que determina completamente la estructura G utilizada para construir el mismo.

Teorema 1. *Sea G una estructura de independencias no dirigida de una distribución positiva isomorfa a grafos $P(\mathbf{V})$. El* conjunto de cierre basado en mantas de Markov *de G es un conjunto de aserciones de independencia condicional que es suficiente para determinar completamente la estructura G.*

Demostración. La demostración formal de este teorema se presenta en el Apéndice B. □

Entonces, para computar el IB-score utilizando el cierre basado en mantas de Markov es necesario ejecutar $n \times (n-1)$ tests estadísticos, ya que se requieren $n-1$ aserciones por cada una de las n variables, un número que resulta cuadrático en el tamaño del dominio. De este modo, es posible descomponer el cómputo del IB-score de la Ecuación (4.4) en una sumatoria de *IB-scores locales* distintos, uno por cada variable X del dominio $\mathbf{V}$:

$$\text{IB-score}(G) = \sum_{X \in \mathbf{V}} \text{IB-score}_X(G), \tag{4.8}$$

donde $\text{IB-score}_X(G) = \sum_{c_i \in \mathcal{C}_X(G)} \log \Pr(c_i \mid D)$. Esta descomposición tiene una importante ventaja: permite computar incrementalmente el puntaje de un estructura G', basándose en cómputos previos del puntaje de una estructura similar G, reutilizando tests estadísticos comunes al cierre de ambas estructuras. Por ejemplo, si dos estructuras G y G' difieren sólo por una arista (X, Y), sólo las mantas de Markov correspondientes a X e Y cambian entre las dos estructuras. Esto permite que pueda computarse el IB-score(G') a partir de IB-score(G), sólo re-computando los IB-score locales IB-score_X and IB-score_Y, y reutilizando los $(n-2)$ IB-scores locales restantes. Consecuentemente, para dos estructuras vecinas que sólo difieren en una arista, el costo de computar IB-score(G') a partir de IB-score(G) se reduce de $n \times (n-1)$ tests de independencia a sólo $2 \times (n-1)$ tests, es decir, de $O(n^2)$ a $O(n)$ tests.

Finalmente, para facilitar la lectura de los métodos de optimización explicados en la siguiente sección, detallamos cómo seguir descomponiendo la función

IB-score de la Ecuación (4.8), considerando que cada uno de los IB-score locales IB-score$_X(G)$ está compuesto por $(n-1)$ términos IB-score$_{X,Y}(G)$, llamados *pairwise IB-scores*, como sigue:

$$\text{IB-score}(G) = \sum_{X \in \mathbf{V}} \sum_{Y \in \mathbf{V} \setminus \{X\}} \text{IB-score}_{X,Y}(G). \tag{4.9}$$

De acuerdo a la Ecuación (4.7), cada pairwise IB-score se obtiene mediante el cómputo de la siguiente probabilidad a posteriori, a partir de los datos:

$$\text{IB-score}_{X,Y}(G) = \left\{ \begin{array}{ll} \log \Pr((X \not\perp\!\!\!\perp Y | \mathbf{MB}^X - \{Y\}) \mid D) & \text{si } (X,Y) \text{ son arista en } G, \\ \log \Pr((X \perp\!\!\!\perp Y | \mathbf{MB}^X) \mid D) & \text{de otro modo.} \end{array} \right\}. \tag{4.10}$$

4.3. Optimización de la función de puntaje basada en independencias

Esta sección explica en detalle el problema de optimización planteado por IBMAP, descripto en la fórmula de la Ecuación (4.5). La necesidad de recurrir a métodos de optimización surge de la exponencialidad del problema, ya que la cantidad de posibles estructuras no dirigidas crece exponencialmente con la cantidad de variables del dominio. Para maximizar el IB-score en un problema con n variables aleatorias, el espacio de búsqueda contiene $2^{\binom{n}{2}}$ estructuras diferentes, ya que está dado por todas las posibles estructuras no dirigidas. Por practicidad, llamaremos a dicho espacio el *espacio de estructuras*.

Un modo de representar las estructuras de independencia fácilmente es a través de su matriz de adyacencias. Esta matriz consta de n filas y n columnas, y la matriz triangular superior de la misma (o la inferior) codifica con un 1 ó un 0 según exista arista o no entre dos variables. Como ejemplo, la Figura 4.1 muestra la matriz de adyacencias del grafo que aparece en la Figura 2.1(a). Para facilitar la lectura del ejemplo, se resaltan las celdas que corresponden a las aristas del grafo (celdas con 1), y además se solapa el grafo de ejemplo en la matriz triangular inferior. Un aspecto interesante de esta representación de estructuras es que permite representar a las mismas como una cadena de bits de longitud

	X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}
X_0		1	1	0	0	0	0	0	0	0	0	0
X_1			0	1	1	0	0	0	0	0	0	0
X_2				1	0	1	0	0	0	0	0	0
X_3					0	1	0	0	1	0	0	0
X_4						0	1	0	0	0	0	0
X_5							0	0	0	0	0	0
X_6								0	0	1	0	0
X_7									1	0	1	1
X_8										1	1	1
X_9											0	1
X_{10}												1
X_{11}												

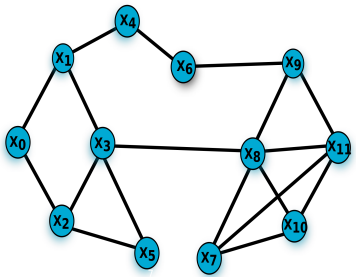

Figura 4.1: Ejemplo de tabla de adyacencias para la Figura 2.1(a)

$\binom{n}{2} = \frac{n\times(n-1)}{2}$. Esta representación se obtiene fácilmente, concatenando las cadenas de bits de las filas de la matriz. Siguiendo el ejemplo de la Figura 4.1, la cadena de bits resultante es `11000000000 0110000000 101000000 01001000 0100000 000000 00100 1011 111 01 1` (sin espacios). Esta codificación resulta útil para implementar algoritmos de búsqueda de estructuras, formulando el espacio de búsqueda como todas las posibles cadenas de bits de longitud $\binom{n}{2} = \frac{n\times(n-1)}{2}$.

A continuación, se explica una serie de estrategias para llevar a cabo dicha maximización, resultando en diferentes algoritmos: IBMAP-BF (búsqueda por fuerza bruta), IBMAP-HC (búsqueda local de ascensión de colinas), IBMAP-HC-RR (búsqueda de ascensión de colinas con reinicios múltiples), IBMAP-GA (búsqueda con algoritmos genéticos), IBMAP-HHC (búsqueda heurística de ascensión de colinas) y IBMAP-HHC-RR (búsqueda heurística de ascensión de colinas con reinicios múltiples). Esta variedad de instanciaciones del enfoque podría extenderse indefinidamente, ya que existe una vasta cantidad de mecanismos de optimización. Sin embargo, con los mecanismos implementados se ha podido obtener información suficiente respecto de la practicidad del enfoque IBMAP. Algunos algoritmos sirven para hacer búsquedas exhaustivas y de gran duración, permitiendo invertir tiempo de cómputo a fin de explorar mejor el espacio de estructuras. Otros algoritmos, en cambio, permiten maximizar el IB-score me-

diante búsqueda heurística, obteniendo también muy buenos resultados pero más eficientemente.

4.3.1. Búsqueda por fuerza bruta

Un algoritmo trivial para maximizar la función IB-score se basa en la sencilla técnica del uso de fuerza bruta. Por simpleza, dicho algoritmo es llamado IBMAP-BF (del inglés, *independence-based maximum a posteriori brute force*). Esta estrategia consiste en enumerar sistemáticamente todas las posibles estructuras candidatas, con el fin de chequear cuál es la estructura cuyo IB-score es más alto.

El Algoritmo 3 muestra el pseudo-código de IBMAP-BF. Se recibe como parámetro de entrada: el conjunto de datos de entrenamiento D y el dominio en cuestión $\mathbf{V}$. Para comenzar, se genera una lista que posee todas las posibles estructuras con $n = |\mathbf{V}|$ variables. Luego, se computa el IB-score de cada una de las estructuras de la lista, almacenando en las variables G^* y $mejorPuntaje$ a la estructura que maximiza IB-score, y su valor de puntaje correspondiente. Finalmente se retorna la variable G^* como la estructura solución.

Algoritmo 3 IBMAP-BF($D, \mathbf{V}$).

```
1: mejorPuntaje ← −∞
2: G* ← null
3: espacioEstructuras ⟵ listar todas las estructuras posibles con n = |V| nodos
4: para cada estructura G en espacioEstructuras hacer
5:     puntaje ← IB-score(G, D)
6:     si puntaje > mejorPuntaje entonces
7:         mejorPuntaje ← puntaje
8:         G* ← G
9: retornar G*
```

Esta estrategia de maximización del IB-score es muy sencilla de implementar. Sin embargo, tiene un costo de ejecución proporcional al número de estructuras posibles, lo que resulta intratable computacionalmente para dominios incluso pequeños. Por ejemplo, para $n = 6$ variables, ya se requiere evaluar el IB-score para $2^{15} = 32768$ estructuras diferentes. Si para todas estas estructuras se computa el

IB-score utilizando el conjunto de cierre basado en mantas de Markov (ver la Sección 4.2), se requiere ejecutar $n \times (n-1) = 6 \times 5 = 30$ tests estadísticos para cada estructura, lo que resultaría en $32768 \times 30 = 983040$ tests necesarios. Afortunadamente, el cómputo de IB-score utilizando el conjunto de cierre basado en mantas de Markov se puede realizar incrementalmente, reutilizando cómputo redundante entre estructuras similares. Aún así, se requerirían 30 tests estadísticos para una primer estructura inicial, y $2 \times (n-1) = 2 \times 5 = 10$ tests estadísticos para cada una de las restantes estructuras, resultando en $30 + (32767 \times 10) = 327700$ tests. Sin embargo, si bien la utilización de este artificio reduce la complejidad en un orden de magnitud, todavía se trata de un costo prácticamente imposible de pagar para dominios de mayor tamaño. La presentación de esta estrategia, lejos de ser práctica, se debe a la necesidad de evaluar sencillamente el potencial de la función de puntaje IB-score.

En el Capítulo 5 se demuestra experimentalmente que el uso de IBMAP-BF mejora significativamente la calidad de las estructuras aprendidas por algoritmos del estados del arte. No obstante, los métodos que se explican a continuación proponen mecanismos de maximización alternativos, con costos computacionales mucho menores a IBMAP-BF, obteniendo estructuras de calidad comparables.

4.3.2. Búsqueda local por ascensión de colinas

En el apartado previo se describe una estrategia de fuerza bruta para maximizar el IB-score. Si bien este mecanismo sirve para mostrar el potencial de dicha función de puntaje de estructuras, se requieren métodos de maximización más prácticos, dada la complejidad del espacio de búsqueda. Por este motivo, en esta sección se propone la utilización de un mecanismo de búsqueda local por ascensión de colinas (Russel y Norvig, 2002). El algoritmo que utiliza esta estrategia de maximización es llamado aquí IBMAP-HC (por sus siglas en inglés *independence-based maximum a posteriori hill-climbing*) Los algoritmos de ascensión de colinas son una técnica de optimización matemática popularmente utilizada debido a su sencillez y facilidad de implementación, como así también por su poca impronta en memoria.

El Algoritmo 4 muestra el pseudo-código de IBMAP-HC. Se recibe como parámetros de entrada: el conjunto de datos de entrenamiento D y el dominio en

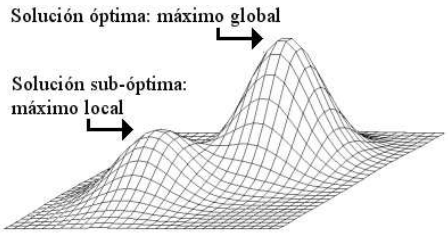

Figura 4.2: Ejemplo de espacio de estados entre posibles estructuras de independencia.

cuestión $\mathbf{V}$. La ascensión de colinas comienza a partir de una estructura con n nodos y sin aristas. Primeramente se computa el IB-score de la estructura inicial, y se pre-establece que ésta es la mejor estructura G hasta el momento. Luego, en la línea 4 se generan todas las estructuras vecinas, es decir, todas las estructuras que se encuentran a sólo una arista de distancia (es decir, resultantes de agregar o quitar una arista). La búsqueda computa el IB-score para cada una de ellas, almacenando en G' la estructura que maximiza esta medida de calidad. Si G' posee mayor puntaje que la mejor estructura actual G, se realiza una ascensión, eligiéndola como la nueva mejor estructura. En este caso se repite el proceso de generación y puntuación de estructuras vecinas para la nueva mejor estructura encontrada. La condición de salida del algoritmo es encontrar un máximo local (ver la Figura 4.2), lo que sucede cuando ninguna de las estructuras vecinas de la mejor estructura superan en puntaje a la misma, retornándose dicha estructura como solución.

Algoritmo 4 IBMAP-HC($D, \mathbf{V}$).

1: $G \leftarrow$ estructura vacía con $n = |\mathbf{V}|$ nodos
2: mejorPuntaje $\leftarrow$ IB-score(G, D)
3: **repetir**
4: $\quad G' \leftarrow \arg\max_{G'' \in \ vecinos(G)}$ IB-score(G'')
5: $\quad$ puntaje $\leftarrow$ IB-score(G', D)
6: $\quad$ **si** puntaje $\leq$ mejorPuntaje **entonces**
7: $\quad\quad$ **retornar** G
8: $\quad$ **sino**
9: $\quad\quad G \leftarrow G'$
10: $\quad\quad$ mejorPuntaje $\leftarrow$ puntaje

Respecto de su costo computacional, el cómputo del puntaje de la estructura inicial del algoritmo requiere de $n \times (n-1)$ tests estadísticos, utilizando el conjunto de cierre basado en mantas de Markov para computar el IB-score. Luego, en el bucle principal del algoritmo se lleva a cabo el procedimiento para seleccionar la estructura a distancia 1 con mejor puntaje, iterando sobre $\binom{n}{2}$ posibles vecinos, y computando el IB-score de cada uno. Utilizando el recurso del cómputo incremental, el IB-score de cada estructuras vecina requiere $2 \times (n-1)$ tests estadísticos, resultando en un costo total de $O(n^3)$ tests por cada ascenso. Luego, si denotamos M al número de ascensos que realiza la búsqueda hasta finalizar, el costo computacional total del algoritmo resulta ser $O(n^2 + Mn^3)$. En el Capítulo 5 se demuestra experimentalmente que M depende tanto de la complejidad del problema (tamaño y densidad de la estructura por aprender), como de la cantidad de datos disponibles. En estos resultados se muestra que el valor de M crece en la mayoría de los casos sub-linearmente con n. En algunos casos particulares donde la estructura por aprender es muy densa (posee muchas aristas), y los datos disponibles son muchos, el valor de M alcanza a crecer linealmente. Por supuesto, el costo de IBMAP-HC es mucho menor que el de IBMAP-BF, pero aún su costo es mayor que el de los algoritmos del estado del arte.

4.3.3. Ascensión de colinas con reinicios múltiples

En el apartado previo se describe una estrategia de escalado simple para maximizar el IB-score. Sin embargo, pese a que este mecanismo es útil para encontrar soluciones que mejoran la calidad estructural frente a algoritmos del estado del arte (ver resultados en el Capítulo 5), es natural que una búsqueda que no se estanque en el óptimo local de la estructura vacía tiene que ser una mejor estrategia de optimización. Por esta razón se presenta adicionalmente un mecanismo de optimización del IB-score que reutiliza IBMAP-HC, pero permitiendo que el mismo reinicie una cantidad configurable de veces, y comenzando desde estructuras generadas aleatoriamente. Dicho algoritmo es llamado IBMAP-HC-RR (del inglés, *independence-based maximum a posteriori hill-climbing with random restarts*).

El Algoritmo 5 muestra el pseudo-código de IBMAP-HC-RR. Se reciben como parámetro de entrada: el conjunto de datos de entrenamiento (D), el dominio en cuestión ($\mathbf{V}$), y adicionalmente la cantidad de reinicios aleatorios que se desea

Algoritmo 5 IBMAP-HC-RR($D, \mathbf{V}, k$).

1: $G^* \leftarrow null$
2: $mejorPuntajeGlobal \leftarrow -\infty$
3: **repetir** k **veces**
4: $G \leftarrow$ estructura aleatoria con $n = |\mathbf{V}|$ nodos y cantidad de aristas entre 0 y $\binom{n}{2}$
5: mejorPuntaje $\leftarrow$ IB-score(G, D)
6: **repetir**
7: $G' \leftarrow \arg\max_{G'' \in \ vecinos(G)}$ IB-score(G'')
8: puntaje $\leftarrow$ IB-score(G', D)
9: **si** puntaje $\leq$ mejorPuntaje **entonces**
10: **si** $mejorPuntaje > mejorPuntajeGlobal$ **entonces**
11: $mejorPuntajeGlobal \leftarrow mejorPuntaje$
12: $G^* \leftarrow G$
13: ir a la línea 3 // siguiente reinicio
14: **sino**
15: $G \leftarrow G'$
16: mejorPuntaje $\leftarrow$ puntaje
17: **retornar** $G*$

que se realicen (k). Luego, se realiza un bucle que realiza k ascensiones de colinas diferentes. Cada ascensión de colinas que se realiza es similar a la de IBMAP-HC, pero comenzando con una estructura inicial aleatoria G', que se genera con $n = |\mathbf{V}|$ nodos y una cantidad de aristas aleatoria entre 0 y $\binom{n}{2}$ (máxima cantidad de aristas que puede poseer una estructura de n nodos). Cuando se encuentra un óptimo local respecto de G', se maximiza el IB-score entre todos los óptimos locales, almacenando la mejor estructura en la variable G^*, y el mejor puntaje en la variable $mejorPuntajeGlobal$. Cuando el algoritmo termina de hacer todos los reinicios aleatorios devuelve G^*, como la mejor estructura obtenida.

El costo computacional de IBMAP-HC-RR es aproximadamente el costo de IBMAP-HC, multiplicado por la constante k, que es la cantidad de reinicios aleatorios. Los resultados experimentales con IBMAP-HC-RR mostrados en el Capítulo 5 muestran que IBMAP-HC-RR permite mejorar la calidad de IBMAP-HC, pero estas mejoras no son significativas.

4.3.4. Búsqueda con un algoritmo genético simple

En esta sección se presenta un algoritmo para maximizar la función IB-score basado en algoritmos genéticos, los cuales dan la posibilidad de buscar en el espacio de estructuras de IB-score sin estancarse en los óptimos locales. Dicho algoritmo es llamado IBMAP-GA (del inglés, *independence-based maximum a posteriori genetic algorithm*). Como ya se explicó en la sección anterior, el desarrollo de IBMAP-HC y IBAMP-HC-RR y su posterior experimentación (ver resultados en el Capítulo 5), dieron las pautas para pensar que la función IB-score es efectiva y útil para hallar estructuras de independencia de alta calidad. Para seguir instanciando nuestro enfoque y desarrollar métodos alternativos de optimización del IB-score se implementó un algoritmo genético para aprendizaje de estructuras utilizando IB-score como función de fitness. Este enfoque tiene algunas ventajas importantes:

- Se utilizan operadores probabilísticos para maximizar el espacio de estructuras, en contraste con los operadores deterministas que se utilizan en las búsquedas locales.
- El algoritmo no se estanca en los óptimos locales del espacio de estructuras.
- Puede resultar sumamente valioso en casos donde la calidad sea una factor preponderante frente al costo computacional (por ejemplo, una aplicación real de aprendizaje de estructuras para descubrimiento de conocimiento). Los algoritmos evolutivos típicamente se utilizan de este modo, corriendo por largos períodos de tiempo para obtener soluciones cada vez más evolucionadas, y obtener desde la última población generada un conjunto de soluciones interesantes de estudiar.

El funcionamiento de IBMAP-GA es idéntico al de un algoritmo genético tradicional, como el que se muestra en la Figura 4.3. Básicamente, las decisiones de diseño de IBMAP-GA (diseño del cromosoma, mecanismos de selección de individuos, etc.) se encuentran desarrolladas a continuación:

Función de fitness: Claramente, la función de fitness a maximizar es IB-score, computado utilizando el conjunto de cierre basado en mantas de Markov sobre cada individuo, según la Ecuación (4.8).

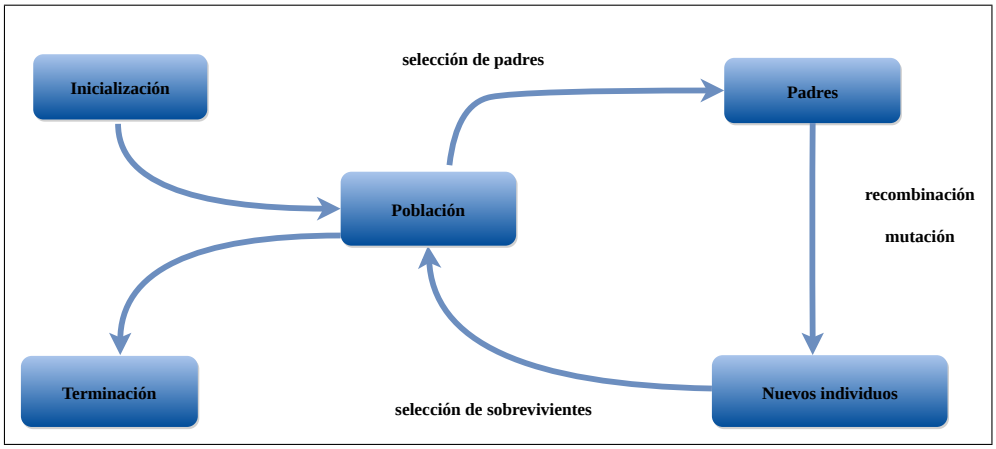

Figura 4.3: Esquema general de un algoritmo genético.

Diseño del cromosoma: El cromosoma en un algoritmo genético representa una solución del problema en forma codificada. Es decir, para maximizar el IB-score las soluciones al problema deben ser cadenas de bits que representen estructuras de independencias, como se explicó al principio de la Sección 4.3 con el ejemplo de la Figura 4.1. En la Figura 4.4 se puede ver un grafo de ejemplo para un problema de 3 variables (a), su correspondiente matriz de adyacencias (b), y el cromosoma que codifica el grafo de ejemplo (c). La acción de decodificación de un cromosoma a un grafo no-dirigido es trivial, deduciendo el grafo desde la cadena de bits.

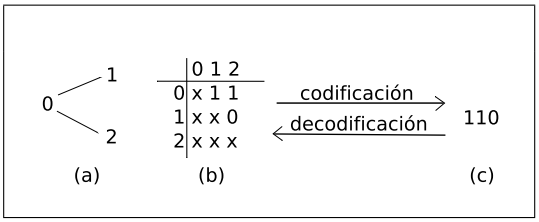

Figura 4.4: Un grafo de ejemplo para un problema de 3 variables, su correspondiente matriz de adyacencias, y el cromosoma que codificaría el grafo de ejemplo.

Población inicial: La población inicial está conformada por cromosomas generados al azar. La cantidad de individuos de la población inicial es un parámetro configurable.

Selección de padres: Para llevar a cabo la selección de padres en una población, se utiliza el método de selección por torneo. Este mecanismo se utiliza para elegir una pareja de padres a partir de la población actual. Para elegir el primer individuo de la pareja se eligen k individuos al azar, y se elige el individuo con fitness más alto. El segundo individuo se elije de la misma manera. Se implementaron también otros mecanismos basados en fitness y basados en edad, presentando convergencias similares del algoritmo. El mecanismo de selección por torneo resulta más indicado debido a que la evaluación del IB-score de cada individuo en relación a los datos requiere de un tiempo computacional considerable. Por simpleza, se escogió la selección por torneo con un $k = 2$ (este valor de k es típico en la literatura).

Cruzamiento: Dada la codificación de un grafo en un cromosoma, cada bit representa la existencia o ausencia de una arista, y no es necesario representar en esta codificación relaciones entre las aristas del grafo, como suele suceder en problemas de permutaciones. Por simpleza, para IBMAP-GA se escogió directamente el cruzamiento uniforme, y éste se lleva a cabo con una probabilidad específica, estipulada como parámetro de entrada (es decir, probabilidad de cruce). El cruzamiento uniforme se lleva a cabo definiendo cada gen de los cromosomas hijos independientemente del resto de los genes. Este cruzamiento consiste en generar un valor aleatorio entre 0 y 1 para cada gen. Si este valor supera un umbral de 0.5, el gen es copiado directamente desde el primer padre, caso contrario, se copia desde el segundo padre. El segundo hijo se crea usando el mapeo inverso. Un ejemplo de este tipo de cruzamiento se muestra en la Figura 4.5.

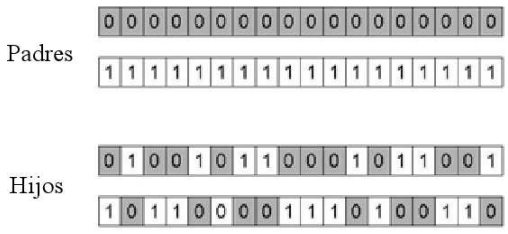

Figura 4.5: Ejemplo de cruce uniforme para dos cromosomas de tamaño 18.

Mutación: Para IMBAP-GA se ha preestablecido el operador de mutación clásico, que consiste en mutar el valor de cada bit de un nuevo individuo, con un valor de probabilidad preestablecido. Este valor se llama *probabilidad de mutación*, y es un parámetro de entrada de IBMAP-GA. La mutación es útil para escapar de los óptimos locales, pero para el problema que intenta resolver IBMAP-GA es incierto el nivel óptimo de mutación a utilizar. Por esto, en nuestras experimentaciones probamos las respuestas del algoritmo para distintos valores del parámetro de probabilidad de mutación.

Selección de sobrevivientes: Se implementaron dos técnicas para selección de supervivientes: steady-state y D-Crowding.

- *steady-state*: este mecanismo resulta práctico, ya que como el espacio a explorar es exponencialmente complejo, es conveniente evitar la pérdida de buenas soluciones halladas. Este mecanismo está basado en el fitness de los individuos de la población. Básicamente, implica reemplazar los k peores individuos de la población actual por los k mejores individuos del conjunto de hijos generados a partir de la población actual. Este valor de k es un parámetro configurable. Para IBMAP-GA se utilizó en los experimentos el 60 % del tamaño de la población, es decir, si la población es de 10 individuos se reemplazan los seis peores individuos de la población actual por los seis mejores hijos generados. Para evitar una convergencia prematura de la población y preservar la diversidad genética, este mecanismo es implementado en IBMAP-GA mediante una política de no permitir hijos duplicados.
- *D-Crowding*: este mecanismo intenta preservar las buenas soluciones halladas, pero manteniendo la diversidad genética de las poblaciones generadas. Este esquema funciona eligiendo parejas de padres en la población, y reemplazando a éstos por sus hijos más cercanos, sólo en caso de que éstos sean mejores. Esto suele favorecer la convergencia hacia óptimos globales.

Condición de terminación: Para resolver este problema no es posible establecer como condición de finalización el hecho de que la función de fitness haya alcanzado algún valor específico (como es natural en algoritmos genéticos),

ya que se desconoce cuál es el valor máximo de IB-score. Por este motivo, el algoritmo está diseñado para generar una cantidad de generaciones máxima N, que es un parámetro de entrada. En los experimentos del Capítulo 5 se eligió un valor de N que garantice la convergencia (Figuras 5.4, 5.5, 5.6, 5.7, 5.8, 5.9).

En el Capítulo 5 se muestra una serie de experimentos donde se muestra que IBMAP-GA tiene capacidad para mejorar la maximización del IB-score respecto de las búsquedas locales. Asimismo, se demuestra también que dicha mejora en la maximización permite en muchos casos mejorar la calidad de las estructuras aprendidas. Si bien las búsquedas locales son más prácticas en términos de costo computacional, esta instanciación con un algoritmo genético puede resultar interesante en situaciones donde se pueda invertir mucho tiempo de cómputo con el objeto de obtener modelos de alta calidad (e.g., descubrimiento de conocimiento).

4.3.5. Búsqueda heurística de ascensión de colinas

En esta sección se presenta un mecanismo de optimización similar a IBMAP-HC, pero que incorpora una función heurística para acelerar la búsqueda. Dicho algoritmo es llamado IBMAP-HHC (por sus siglas en inglés *independence-based maximum a posteriori heuristic hill-climbing*) . El uso de una función heurística permite evitar el alto costo en que incurre IBMAP-HC al maximizar el IB-score entre todas las estructuras vecinas para cada ascenso de la búsqueda local, ya que esto requiere computar el IB-score para las $\binom{n}{2}$ estructuras vecinas a distancia 1. El procedimiento utilizado en IBMAP-HC para este propósito es trivial, por lo que IBMAP-HHC incorpora una función heurística que estima el mejor vecino sin la necesidad de evaluar el IB-score de todas las estructuras vecinas, es decir, un costo de $O(1)$ tests estadísticos. Como consecuencia, el costo de IBMAP-HHC se reduce en un orden de magnitud respecto del costo de IBMAP-HC.

En el Algoritmo 6 se muestra el pseudo-código de IBMAP-HHC, que es idéntico al de IBMAP-HC, con un único cambio en la línea 4, donde la estructura vecina G' es obtenida mediante la llamada a la función *heurística-mejor-vecino*. Esta heurística permite obtener calidades similares reduciendo el costo computacional en un orden de magnitud. De este modo, IBMAP-HHC ha mostrado ser competitivo en costo computacional respecto de los algoritmos del estado del arte,

obteniendo mejoras significativas en la calidad para dominios de gran tamaño.

Algoritmo 6 IBMAP-HHC($D, \mathbf{V}$).

```
1: G ← estructura vacía con n = |V| nodos
2: mejorPuntaje ← IB-score(G, D)
3: repetir
4:    G' ← heurística-mejor-vecino(G, IB-score(G))      // ver el Algoritmo 7
5:    puntaje ← IB-score(G', D)
6:    si puntaje ≤ mejorPuntaje entonces
7:       retornar G
8:    sino
9:       G ← G'
10:      mejorPuntaje ← puntaje
```

El pseudo-código de la función *heurística-mejor-vecino* está descripto en el Algoritmo 7. Como parámetro de entrada se recibe una estructura (la estructura actual G en la ascensión de colinas), y su puntaje correspondiente IB-score(G). La función primero selecciona en la línea 1 el "par óptimo" (X^*, Y^*) como la arista (o no-arista) menos confiable de la estructura actual G. Esto puede efectuarse utilizando una estructura de datos para representar IB-score(G), conteniendo los $n \times (n-1)$ pairwise scores IB-score$_{X,Y}(G)$ de la Ecuación (4.9) (ver la Sección 4.2). Luego, el mejor vecino G' se construye en la línea 2 como una copia de G, pero invirtiendo el par (X^*, Y^*) en la matriz de adyacencias (es decir, se borra la arista cuando ésta existe, o se agrega cuando no existe). La función concluye retornando esta estructura G'.

Algoritmo 7 heurística-mejor-vecino(G, IB-score(G))

```
1: (X*, Y*) ← arg mín_{(X,Y)∈(V×V),X≠Y} IB-score_{X,Y}(G) + IB-score_{Y,X}(G)
2: G' ← G with (X*, Y*) flipped
3: retornar G'
```

La idea central de la heurística se encuentra en la línea 1 del Algoritmo 7. Para comprender la idea, es importante notar que el número de vecinos que difieren sólo en una arista es el mismo número que la cantidad de pares diferentes de variables (X, Y), es decir, $n \times (n-1)/2$ pares. Desde este punto de vista, la Ecuación (4.9) puede verse como una suma de 2 *pairwise IB-scores* por cada uno

de los pares de variables. Esto resulta en la siguiente expresión del IB-score:

$$\text{IB-score}(G) = \sum_{(X,Y)\in\mathbf{V}\times\mathbf{V}, X\neq Y} \text{IB-score}_{X,Y}(G) + \text{IB-score}_{Y,X}(G). \qquad (4.11)$$

Con esta nueva forma del IB-score, está claro que la minimización encuentra el par (X^*, Y^*) cuya contribución al puntaje IB-score(G) es mínima. La suposición que realiza la heurística es que la estructura resultante de invertir el par (X^*, Y^*) es similar a maximizar el IB-score entre todas las estructuras vecinas (que es lo que hace IBMAP-HC). Como se explica en la Sección 4.2, para computar incrementalmente IB-score(G') desde IB-score(G) sólo se requiere re-computar IB-score$_X(G')$ y IB-score$_Y(G')$. La aproximación que se realiza en esta minimización consiste en asumir que IB-score$_X(G') \approx$ IB-score$_{X,Y}(G')$, y que IB-score$_Y(G') \approx$ IB-score$_{Y,X}(G')$, ignorando los términos restantes IB-score$_{X,W}(G')$ y IB-score$_{Y,W}, \forall W \in \mathbf{V} \setminus \{X, Y\}$. Esto se basa en el hecho de que, desde G a G', se espera un cambio fuerte en los términos IB-score$_{X,Y}$ y IB-score$_{Y,X}$, ya que la probabilidad a posteriori de dependencia se utiliza en una estructura, y la probabilidad a posteriori de independencia se utiliza en la otra. En contraste, se asume que los términos ignorados tendrán un cambio débil, porque sólo la manta de Markov de X e Y han cambiado, y por lo tanto las aserciones sólo varían en el conjunto condicionante. Esta aproximación es posible debido a que los *pairwise IB-scores* que corresponden a la arista invertida IB-score$_{X,Y}(G')$ y IB-score$_{X,Y}(G)$ son complementarios en ambas estructuras G y G', es decir, la probabilidad a posteriori de independencia y la de dependencia suman en total 1. Esto permite estimar IB-score$_{X,Y}(G')$ a partir de IB-score$_{X,Y}(G)$, sin la necesidad de efectuar ningún test de independencia. Esta estimación se realiza implícitamente en la minimización. La suposición que realiza la heurística es que los términos ignorados tendrán un impacto mínimo en la búsqueda del vecino óptimo. Por supuesto, al tratarse de una aproximación, sólo los resultados empíricos pueden dar indicios de su efectividad. En el peor caso, la aproximación resultaría en la selección de un vecino sub-óptimo. Esto, sin embargo, no es diferente de muchos otros algoritmos de optimización que siguen caminos sub-óptimos (por ejemplo, los mecanismos de cruzamiento de IBMAP-GA).

El costo computacional de IBMAP-HHC es el siguiente. Para el cómputo del puntaje de la estructura inicial del algoritmo se requieren $n \times (n-1)$ tests estadísticos, al igual que IBMAP-HC. Luego, en el bucle principal del algoritmo

se lleva a cabo el procedimiento para seleccionar la estructura a distancia 1 con mejor puntaje, con costo $O(1)$, a diferencia de IBMAP-HC que incurre en un costo de $O(n^3)$. Luego, denotando nuevamente M al número de ascensos, el costo computacional total del algoritmo resulta ser de $O(n^2+Mn)$ tests estadísticos. En el Capítulo 5 se demuestra experimentalmente que M no representa una fuente de complejidad extra, ya que este valor depende tanto de la complejidad del problema (tamaño y densidad de la estructura por aprender), como de la cantidad de datos disponibles. En estos resultados se muestra que el valor de M crece en la mayoría de los casos sub-linealmente con n. En algunos casos particulares donde la estructura por aprender es muy densa (posee muchas aristas), y los datos disponibles son muchos, el valor de M alcanza a crecer linealmente.

Respecto de la calidad estructural, en el Capítulo 5 se muestra experimentalmente que IBMAP-HHC aprende estructuras que mejoran significativamente las calidades de las estructuras aprendidas por algoritmos del estado del arte. También se corrobora que las calidades de IBMAP-HHC son similares a las de IBMAP-HC, probando la efectividad de la heurística utilizada. Adicionalmente, se demuestra que el costo computacional de IBMAP-HHC es altamente competitivo respecto de los algoritmos del estado del arte. Por último, en la Sección 5.6 se presentan mediciones empíricas de la superficie del IB-score, mostrando que en la mayoría de los casos la estrategia de selección de estructuras de IBMAP-HHC permite realizar optimizaciones realmente efectivas en el espacio de estructuras.

4.3.6. Búsqueda heurística de ascensión de colinas con reinicios múltiples

En esta sección se presenta un un algoritmo de optimización similar a IBMAP-HHC, pero que puede reiniciar una cantidad configurable de veces, y comenzando desde estructuras generadas aleatoriamente. Dicho algoritmo es llamado IBMAP-HHC-RR (por sus siglas en inglés *independence-based maximum a posteriori heuristic hill-climbing with random restarts*). Básicamente, se trata del mismo algoritmo IBMAP-HC-RR, pero en vez de utilizar la selección de estructuras vecinas naïve, utiliza la misma heurística que se implementa en IBMAP-HHC.

El Algoritmo 8 muestra el pseudo-código de IBMAP-HHC-RR. Como puede verse a simple vista, se trata del mismo pseudo-código de IBMAP-HC-RR, pero

Algoritmo 8 IBMAP-HHC-RR($D, \mathbf{V}, k$).

1: $G^* \leftarrow null$
2: $mejorPuntajeGlobal \leftarrow -\infty$
3: **repetir** k **veces**
4: $G \leftarrow$ estructura con $n = |\mathbf{V}|$ nodos y cantidad de aristas aleatoria entre 0 y $\binom{n}{2}$
5: mejorPuntaje $\leftarrow$ IB-score(G, D)
6: **repetir**
7: $G' \leftarrow$ *heurística-mejor-vecino*(IB-score(G)) *// ver el Algoritmo 7*
8: puntaje $\leftarrow$ IB-score(G', D)
9: **si** puntaje $\leq$ mejorPuntaje **entonces**
10: **si** $mejorPuntaje > mejorPuntajeGlobal$ **entonces**
11: $mejorPuntajeGlobal \leftarrow mejorPuntaje$
12: $G^* \leftarrow G$
13: ir a línea // siguiente reinicio
14: **sino**
15: $G \leftarrow G'$
16: mejorPuntaje $\leftarrow$ puntaje
17: **retornar** $G*$

utilizando la heurística del Algoritmo 7 en la línea 7. El costo computacional de IBMAP-HHC-RR es aproximadamente el mismo costo que el de IBMAP-HHC, multiplicado por la constante k, que es la cantidad de reinicios aleatorios. Los resultados experimentales con IBMAP-HHC-RR mostrados en el Capítulo 5 muestran que IBMAP-HHC-RR permite en muchos casos mejorar la calidad de IBMAP-HHC, pero dichas mejoras no son significativas.

Capítulo 5

Evaluación experimental

En este capítulo se describe una serie de experimentos realizados a fin de probar empíricamente qué tan robusto es el enfoque IBMAP, y la eficiencia de los algoritmos que instancian dicho enfoque. Los resultados experimentales se encuentran organizados del siguiente modo:

- Primeramente, se presenta una evaluación sobre datos sintéticos para evaluar el enfoque IBMAP sobre dominios de tamaño pequeño. El objetivo de este experimento es comparar las calidades estructurales obtenidas por las distintas instanciaciones de IBMAP respecto de algoritmos del estado del arte, sin importar su costo computacional.

- Una evaluación sobre datos sintéticos, donde se muestra cómo funciona IBMAP-GA. Este algoritmo sólo se evalúa en esta sección a fin de realizar un análisis más detenido sobre su funcionamiento, ya que éste puede parametrizarse con una gran cantidad de configuraciones diferentes.

- Una evaluación sobre datos sintéticos en dominios de gran tamaño. En estos experimentos sólo presentamos resultados comparando IBMAP-HHC (la instanciación más eficiente de IBMAP) contra los algoritmos del estado del arte.

- Una evaluación sobre datos del mundo real (conjuntos de datos de referencia, en inglés *benchmark*), obtenidos desde repositorios de aprendizaje de máquinas y conjuntos de datos para descubrimiento de conocimiento.

- Una serie de experimentos para confirmar que el costo computacional de IBMAP-HHC es competitivo respecto de los algoritmos del estado del arte. Para esto, se analizan los resultados de tiempo de corrida de los experimentos de calidad estudiados previamente. Además, se reportan resultados de una experimentación realizada a fin de evaluar cómo la complejidad de la búsqueda y la cantidad de datos disponibles afectan en el costo computacional de la búsqueda local.

- Un experimento que analiza empíricamente qué tan eficientes son los mecanismos de búsqueda propuestos para maximizar el IB-score sobre el espacio de estructuras.

- Por último, se muestran resultados de aplicar IBMAP-HHC en un problema real: los algoritmos EDAs (Mühlenbein y Paaß, 1996; Larrañaga y Lozano, 2002). Los EDAs son una variación de los algoritmos evolutivos que reemplazan las etapas de cruzamiento y mutación por el muestreo de poblaciones a partir de una distribución de probabilidades aprendida a partir de la población seleccionada. En este experimento se muestra que la calidad del aprendizaje de estructuras influencia fuertemente en los resultados de la optimización evolutiva.

5.1. Evaluación de calidad estructural en distribuciones pequeñas sobre datos sintéticos

Esta sección muestra un conjunto de experimentos sobre datos sintéticos para distribuciones de pequeño tamaño. El experimento consiste en comparar la calidad de las estructuras aprendidas por los algoritmos que utilizan el enfoque IBMAP (que fueron presentados en el Capítulo 4) respecto de la calidad de las estructuras aprendidas por algoritmos del estado del arte. Como primer competidor se ha seleccionado a GSMN ya que éste es el algoritmo más representativo y simple del estado del arte, que en términos de calidad es similar a los demás algoritmos revisados (ver Capítulo 3, Sección 3.6). Adicionalmente, se ha seleccionado como competidor a un algoritmo basado en independencias que utiliza la variante HITON-PC de GS, y que es un algoritmo del estado del arte para

aprendizaje de redes de Bayes. Dicho algoritmo es más robusto que GS a errores en los tests. Para utilizar HITON-PC a fin de aprender estructuras de redes de Markov, se presenta el algoritmo HHC-MN, como una adaptación del algoritmo HHC de Aliferis et al. (2010b). En el Apéndice C se detalla el funcionamiento de dicho algoritmo.

Con el objeto de evaluar todas las instanciaciones de nuestro enfoque IBMAP, los experimentos de esta sección se muestran sobre dominios de tamaño pequeño. Dado que IBMAP-BF lleva a cabo una búsqueda por fuerza bruta, en este primer experimento se prueba empíricamente que maximizar la función de puntajes IB-score mejora significativamente la calidad de los algoritmos competidores. Adicionalmente, se muestra que las demás instanciaciones del enfoque obtienen calidades comparables a las de IBMAP-BF. Por sencillez, en esta experimentación se omiten resultados de IBMAP-GA, ya que éste se evalúa en forma aislada en la Sección 5.2.

Los conjuntos de datos sintéticos han sido obtenidos muestreando desde distribuciones generadas a partir de redes de Markov aleatorias. Esto permite realizar un estudio sistemático y controlado, otorgando control sobre la complejidad del problema, y permitiendo evaluar la calidad de las estructuras aprendidas por cada algoritmo. Las redes de Markov sintéticas fueron generadas para este experimento para $n \in \{6, 12, 20\}$ variables binarias. Para cada tamaño de dominio, se consideraron distintos tamaños de conectividad creciente: $\tau \in \{1, 2, 4\}$, donde τ indica la cantidad de aristas que contiene cada nodo, en promedio. Por esto, mientras más grande sea el valor de τ, más complejo resulta el aprendizaje de la estructura solución. Para cada caso de τ se generaron aleatoriamente 10 redes diferentes, considerando como aristas a los primeros $n\tau/2$ pares de variables de una permutación aleatoria del conjunto de todos los pares de variables posibles.

Dado que la estructura de independencias determina la factorización de la distribución (ver la Sección 2.1.2), el modelo completo para cada conjunto de datos se obtuvo generando los parámetros numéricos aleatoriamente. Para que los datos se generen desde modelos correctos, y que a su vez las dependencias se representen fuertemente por las aristas de la estructura, se generaron factores de dos variables $\phi(X, Y)$ por cada arista de las estructuras aleatorias generadas, y sus parámetros numéricos correspondientes fueron generados de modo que la correlación entre las variables de las aristas sea fuerte. Para esto, se forzaron los

parámetros para que el log-odds de cada factor $\varepsilon_{X,Y} = \log\left(\frac{\phi(X=0,Y=0)\phi(X=1,Y=1)}{\phi(X=0,Y=1)\phi(X=1,Y=0)}\right)$ sea igual a 1 para todas las aristas. Esto resulta en una ecuación sobre los valores de la función potencial con 4 valores desconocidos. La generación de los parámetros consistió en crear 3 parámetros aleatoriamente en un rango de 0 a 1, y el parámetro restante se resuelve desde la ecuación. Este procedimiento es una forma estándar de generar modelos probabilísticos aleatorios Agresti (2002).

Una vez que se han construido las redes de Markov aleatoriamente, los datos se generan utilizando un muestreador de Gibbs, que es un algoritmo que se utiliza para obtener una secuencia aproximada de puntos de datos a partir de una distribución multivariada. Se generaron 1600 puntos de datos para cada modelo. Posteriormente, se ejecutaron los diferentes algoritmos utilizando porciones crecientes del conjunto de datos generado $D \in \{25, 100, 400, 1600\}$, para cada combinación de (n, τ). Los algoritmos a comparar en este experimento son GSMN, HHC-MN, IBMAP-BF, IBMAP-HC, IBMAP-HC-RR (para 100, 500 y 1000 reinicios aleatorios), IBMAP-HHC, y IBMAP-HHC-RR (para las mismas configuraciones que IBMAP-HC-RR). Para comparar todos los algoritmos al mismo nivel, hemos corrido a todos éstos utilizando el test estadístico Bayesiano (Margaritis, 2005) (ver el Apéndice A).

Para llevar a cabo una medición de los errores que posee cada estructura aprendida se reporta la *distancia de Hamming* entre la estructura que se ha aprendido y la estructura solución, desde la cuál se han generado los datos. La distancia de Hamming se computa como la suma entre los falsos positivos y los falsos negativos. Los falsos positivos son las aristas aprendidas que no existen en la estructura solución, y los falsos negativos son las aristas de la estructura solución que no fueron aprendidas. Otra medida de calidad que comúnmente se tiene en cuenta para medición de los errores estructurales es la F-measure, que es una medida armónica que combina la precisión y el recall de la estructura. En este trabajo se omite la presentación de los resultados de F-measure, ya que presentan en todos los casos tendencias idénticas a las que se ven sobre la distancia de Hamming. Esto último puede corroborarse en los resultados experimentales publicados en Schlüter et al. (2014).

En la Figura 5.1 se muestran los resultados para $n = 6$ variables. La figura organiza los resultados en varias gráficas, mostrando la distancia de Hamming de la estructura aprendida por cada algoritmo, promediando los resultados sobre

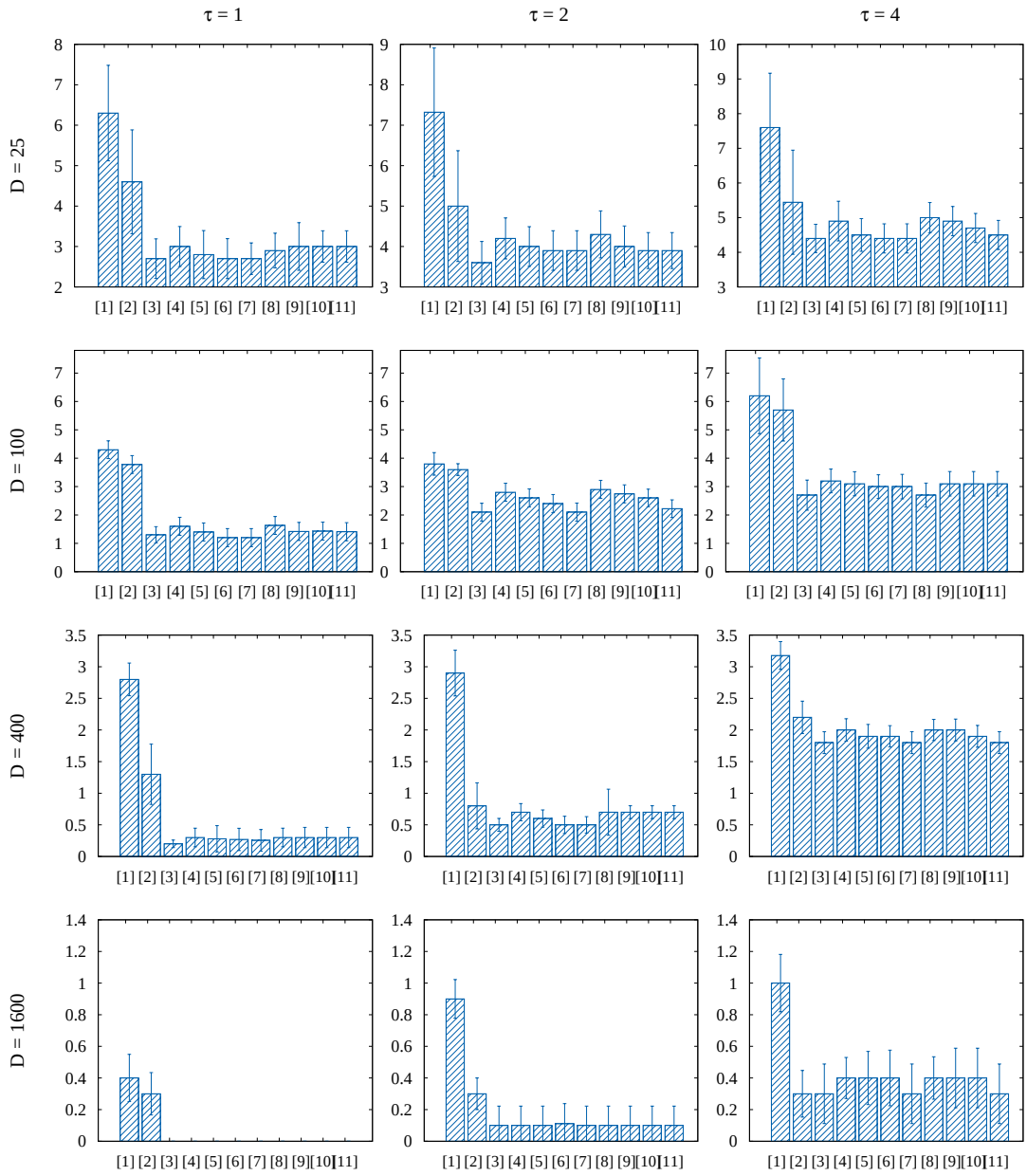

Figura 5.1: Distancia de Hamming en problemas con $n = 6$ variables. Menor distancia de Hamming es mejor. Lista de algoritmos: [1] GSMN, [2] HHC-MN, [3] IBMAP-BF, [4] IBMAP-HC, [5] IBMAP-HC-RR(100), [6] IBMAP-HC-RR(500), [7] IBMAP-HC-RR(1000), [8] IBMAP-HHC, [9] IBMAP-HHC-RR(100), [10] IBMAP-HHC-RR(500), [11] IBMAP-HHC-RR(1000).

diez conjuntos de datos diferentes, y mostrando también la desviación estándar. La figura ordena las gráficas disponiendo en las filas de la grilla los resultados sobre distintos tamaños crecientes del conjunto de datos de entrenamiento $D \in \{25, 100, 400, 1600\}$, y en las columnas de la grilla para los distintos tamaños de conectividad $\tau \in \{1, 2, 4\}$. De este modo, pueden revisarse verticalmente las tendencias de calidad a medida que crece la cantidad de datos disponibles para una misma dificultad de estructuras (τ en la columna). Asimismo, también puede revisarse horizontalmente cómo para una misma cantidad de datos ocurren tendencias en la calidad de las estructuras aprendidas, a medida que se incrementa la dificultad de las estructuras por aprender. Como es esperable, estos resultados muestran que para todos los algoritmos, mientras más compleja es la estructura solución (es decir, mientras crece τ), más grande es la distancia de Hamming de las estructuras aprendidas respecto de la solución. Puede verse también que para cualquier valor fijo de D, la cantidad de errores de cada algoritmo crece con τ. Como GSMN (abreviado como [1]) y HHC-MN (abreviado como [2]) siguen el enfoque tradicional basado en independencias, se espera que obtengan mejores calidades cuando los conjuntos de datos son grandes, (es decir, los casos de valores más grandes de D y valores más chicos de τ). Las gráficas muestran claramente que todos los algoritmos que instancian IBMAP mejoran en todos los casos a las estructuras aprendidas por GSMN y HHC-MN, con calidades significativamente mejores (distancias de Hamming menores).

Para todos los casos, GSMN tiene la convergencia más lenta en D para reducir los errores estructurales. Esto ocurre con GSMN debido a que este algoritmo tiende a agregar muchos falsos positivos en la fase de crecimiento, lo que requiere luego que en la fase de encogimiento se ejecuten tests que involucran muchas variables, volviéndose estos mismos muy poco confiables. Esto produce una cantidad numerosa de errores en cascada. En el caso de HHC-MN, puede verse que los errores estructurales se reducen significativamente respecto de GSMN. Estas mejoras se obtienen gracias al uso de una estrategia de eliminación intercalada con el uso de una función heurística de inclusión (ver el Apéndice C). El resto de los algoritmos son las diversas instanciaciones de IBMAP: IBMAP-BF, IBMAP-HC, IBMAP-HC-RR con 100 , 500, y 1000 reinicios aleatorios, IBMAP-HHC, y IBMAP-HHC-RR con 100, 500 y 1000 reinicios aleatorios; abreviadas respectivamente como [3],[4],[5],[6],[7],[8],[9],[10] y [11]. Como puede verse claramente en

las gráficas, en todos los casos los resultados de IBMAP-BF muestran calidades ligeramente mejores que todos los algoritmos. Este resultado era esperable, ya que IBMAP-BF realiza una búsqueda por fuerza bruta, es decir, obtiene la estructura que maximiza el IB-score. Sin embargo, resulta interesante también observar que el resto de las instanciaciones de IBMAP muestran calidades comparables a IBMAP-BF, pero llevando a cabo búsquedas que escalan mucho más eficientemente con el tamaño del dominio. Además, en estos resultados puede apreciarse que las calidades de IBMAP-HC-RR son comparables a las de IBMAP-HC, obteniendo apenas mejoras leves, no significativas. Estas mejoras son esperables, ya que se trata de 10, 100 y 500 reinicios aleatorios. Similarmente, las calidades obtenidas por IBMAP-HHC-RR son ligeramente mejores las obtenidas por IBMAP-HHC. Esto indica que tanto IBMAP-HC-RR como IBMAP-HC-RR pueden resultar apropiados en casos donde se está dispuesto a invertir tiempo de cómputo a cambio de obtener un modelo de alta calidad (al igual que el algoritmo IBMAP-GA). En cambio, si lo que se desea es un algoritmo que obtenga altas calidades al mejor costo posible, la mejor opción es el algoritmo IBMAP-HHC. En la Sección 5.5 se muestran experimentos de tiempos de corrida y otros experimentos adicionales que confirman que la complejidad temporal de IBMAP-HHC es competitiva con la de los algoritmos del estado del arte.

Asimismo, en las Figuras 5.2 y 5.3 se muestran los resultados para $n = 12$ y $n = 20$, respectivamente. En estas figuras se muestran los resultados para los mismos algoritmos, pero omitiendo IBMAP-BF (abreviado como [3]), porque resulta imposible de correr (en los casos de $n = 12$ se requeriría evaluar IB-score para 2^{66} estructuras, y en los casos de $n = 20$ para 2^{190} estructuras). Estos resultados muestran tendencias similares a los analizados para $n = 6$, pero permiten ver además ciertos patrones de comportamiento en los que los algoritmos GSMN (abreviado como [1]) y HHC-MN (abreviado como [2]) funcionan mejor que los algoritmos del enfoque IBMAP, como puede verse en la columna de $\tau = 4$. Este resultado es esperable, si se tiene en cuenta que los algoritmos IBMAP-HC, IBAMP-HC-RR, IBMAP-HHC y IBMAP-HHC-RR hacen búsquedas locales, y para estos tamaños de dominio, cuando $\tau = 4$ los tests estadísticos utilizados en el conjunto de cierre basado en mantas de Markov poseen una complejidad muestreal muy alta, ya que involucran muchas variables en el conjunto condicionante.

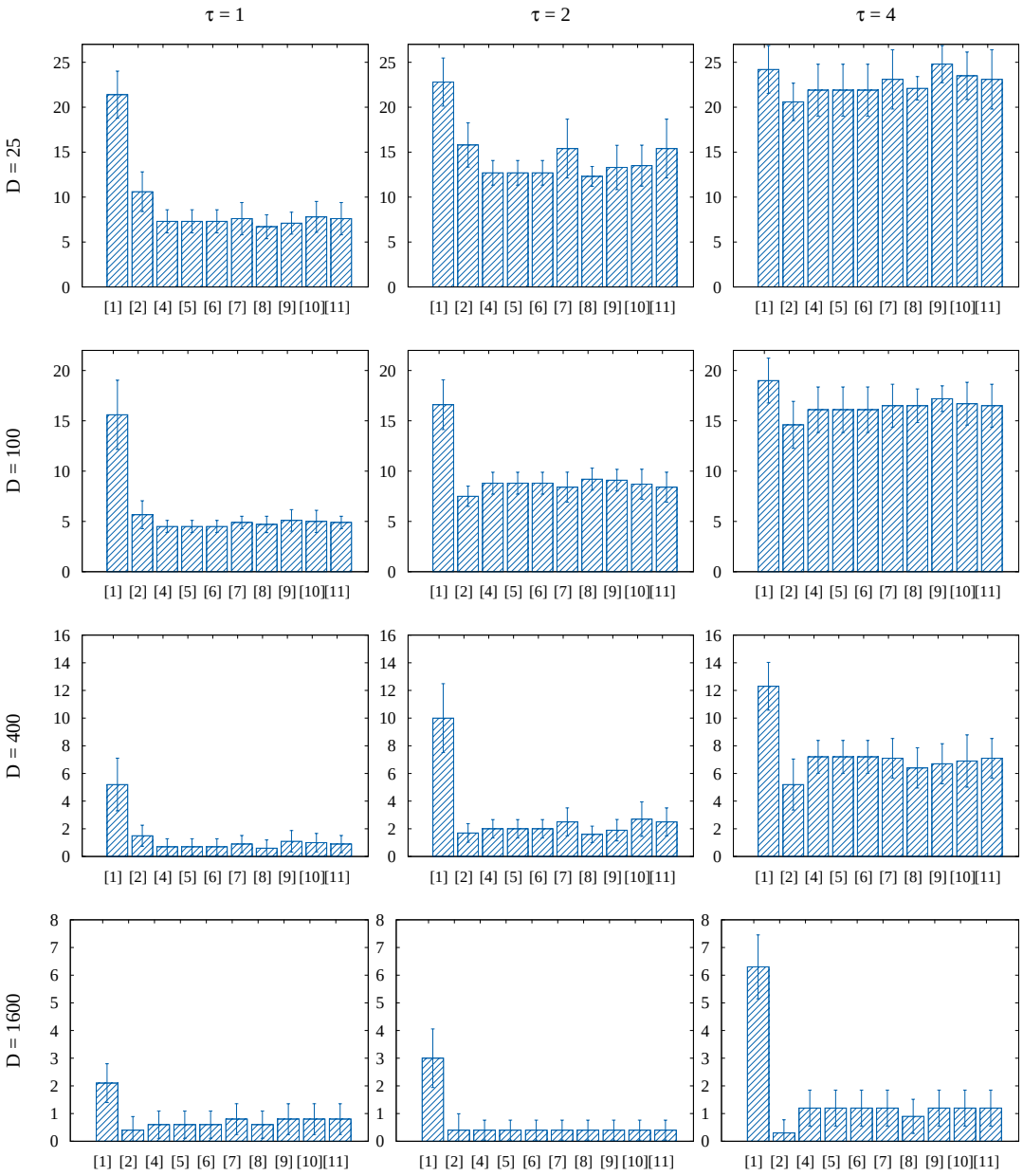

Figura 5.2: Distancia de Hamming en problemas con $n = 12$ variables. Menor distancia de Hamming es mejor. Lista de algoritmos: [1] GSMN, [2] HHC-MN, [4] IBMAP-HC, [5] IBMAP-HC-RR(100), [6] IBMAP-HC-RR(500), [7] IBMAP-HC-RR(1000), [8] IBMAP-HHC, [9] IBMAP-HHC-RR(100), [10] IBMAP-HHC-RR(500), [11] IBMAP-HHC-RR(1000).

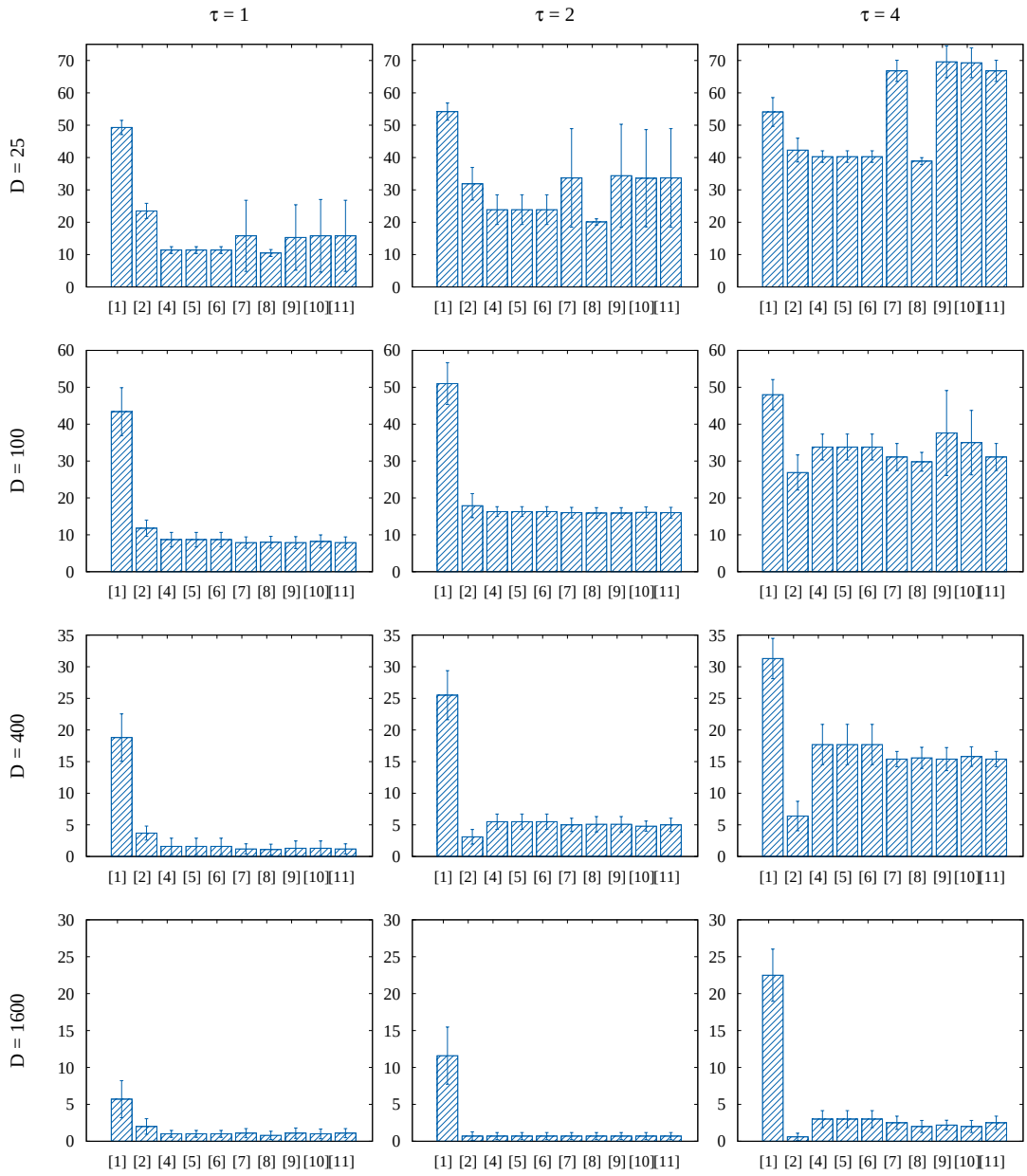

Figura 5.3: Distancia de Hamming en problemas con $n = 20$ variables. Menor distancia de Hamming es mejor. Lista de algoritmos: [1] GSMN, [2] HHC-MN, [4] IBMAP-HC, [5] IBMAP-HC-RR(100), [6] IBMAP-HC-RR(500), [7] IBMAP-HC-RR(1000), [8] IBMAP-HHC, [9] IBMAP-HHC-RR(100), [10] IBMAP-HHC-RR(500), [11] IBMAP-HHC-RR(1000).

5.2. Evaluación de calidad estructural con el enfoque basado en algoritmo genético

Esta sección muestra un conjunto de experimentos con el algoritmo IBMAP-GA sobre datos sintéticos, generados del mismo modo que los utilizados en la sección anterior. Debido a la flexibilidad de la parametrización de este algoritmo, esta sección compara la calidad de IBMAP-GA utilizando diversas configuraciones del mismo, como el tamaño de población, la probabilidad de mutación, la probabilidad de cruzamiento, la técnica de selección de individuos, etc. Para mostrar que dicho algoritmo genético puede hacer una buena maximización del IB-score se muestran dos experimentos distintos. El primero permite analizar la curva de evolución generada por IBMAP-GA, mostrando empíricamente su progreso. Con este experimento se valida que el algoritmo puede realizar una maximización efectiva del IB-score, encontrando en la mayoría de los casos estructuras con IB-score más alto que IBMAP-HHC. En un segundo experimento se muestran además resultados en términos de la calidad de las estructuras aprendidas.

El primer experimento se realizó sobre un conjunto de datos de $n = 12$ variables binarias, con $\tau = 2$, y $D = 100$ puntos de datos. Como condición de corte para IBMAP-GA, se fijaron 1000 iteraciones para cada corrida del algoritmo. En las Figuras 5.4, 5.5 y 5.6 puede apreciarse un conjunto de gráficas en las que se muestra la evolución del valor de fitness de IBMAP-GA utilizando el esquema de selección de supervivientes *steady-state* para una población de 10, 100 y 500 individuos, respectivamente. Del mismo modo, las Figuras 5.7, 5.8 y 5.9 muestran el mismo experimento, pero utilizando el esquema de selección de supervivientes *D-crowding* también para 10, 100 y 500 individuos, respectivamente. Dichas figuras muestran una curva que representa la evolución del IB-score del mejor individuo encontrado en cada iteración, y además se muestran en líneas rectas los valores de IB-score de las estructuras encontradas por IBMAP-HHC y GSMN. Cada figura organiza un conjunto de gráficas en forma de grilla, disponiendo en las distintas columnas los resultados de utilizar probabilidad de mutación de 0.01, 0.1 y 0.9, y en las filas se organizan los resultados según la probabilidad de cruzamiento utilizada (0.3, 0.6, y 0.9). Debido a que en todos los casos del experimento IBMAP-GA convergió al máximo antes de las primeras 200 iteraciones, las figuras sólo muestran este escenario, para facilitar la visualización y el análisis.

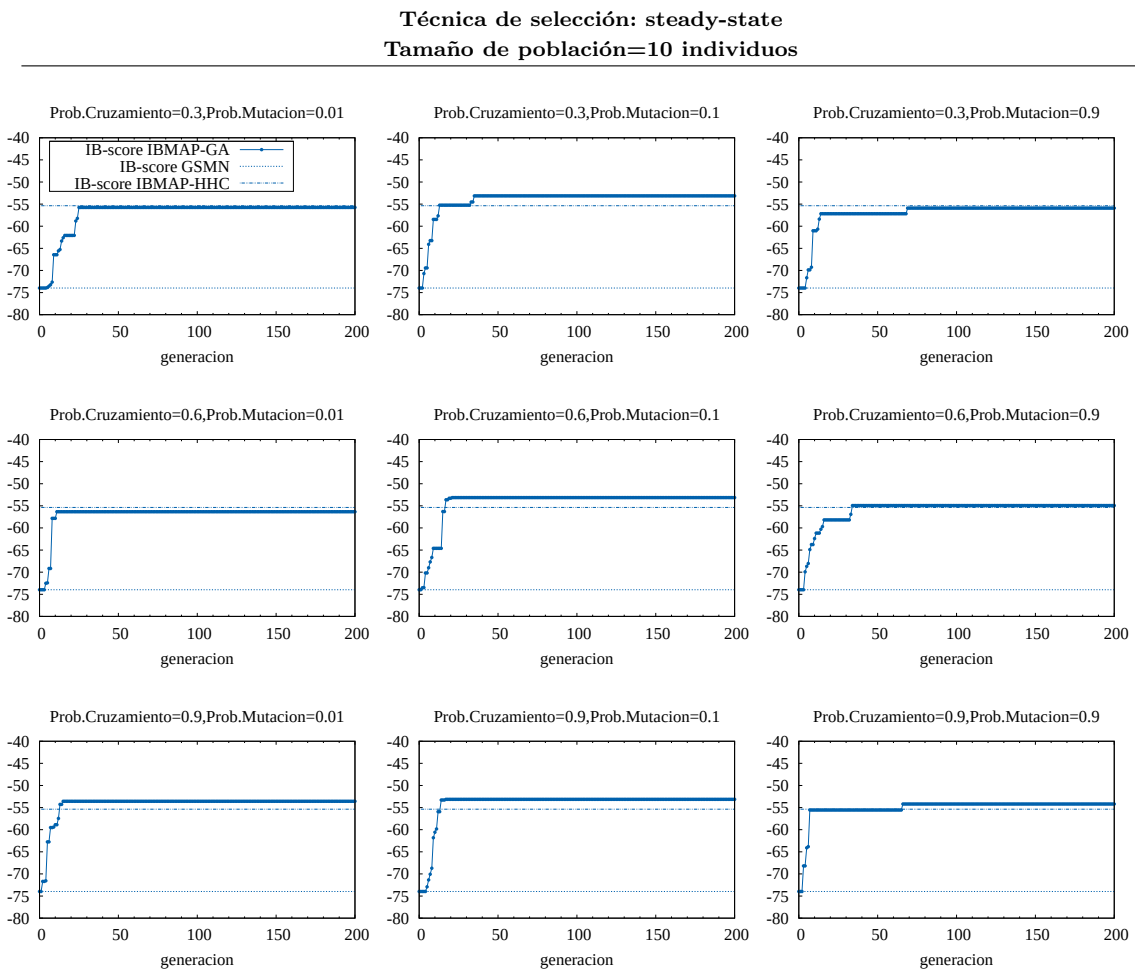

Figura 5.4: Evolución de IBMAP-GA utilizando una población de 10 individuos y selección de supervivientes steady-state. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales.

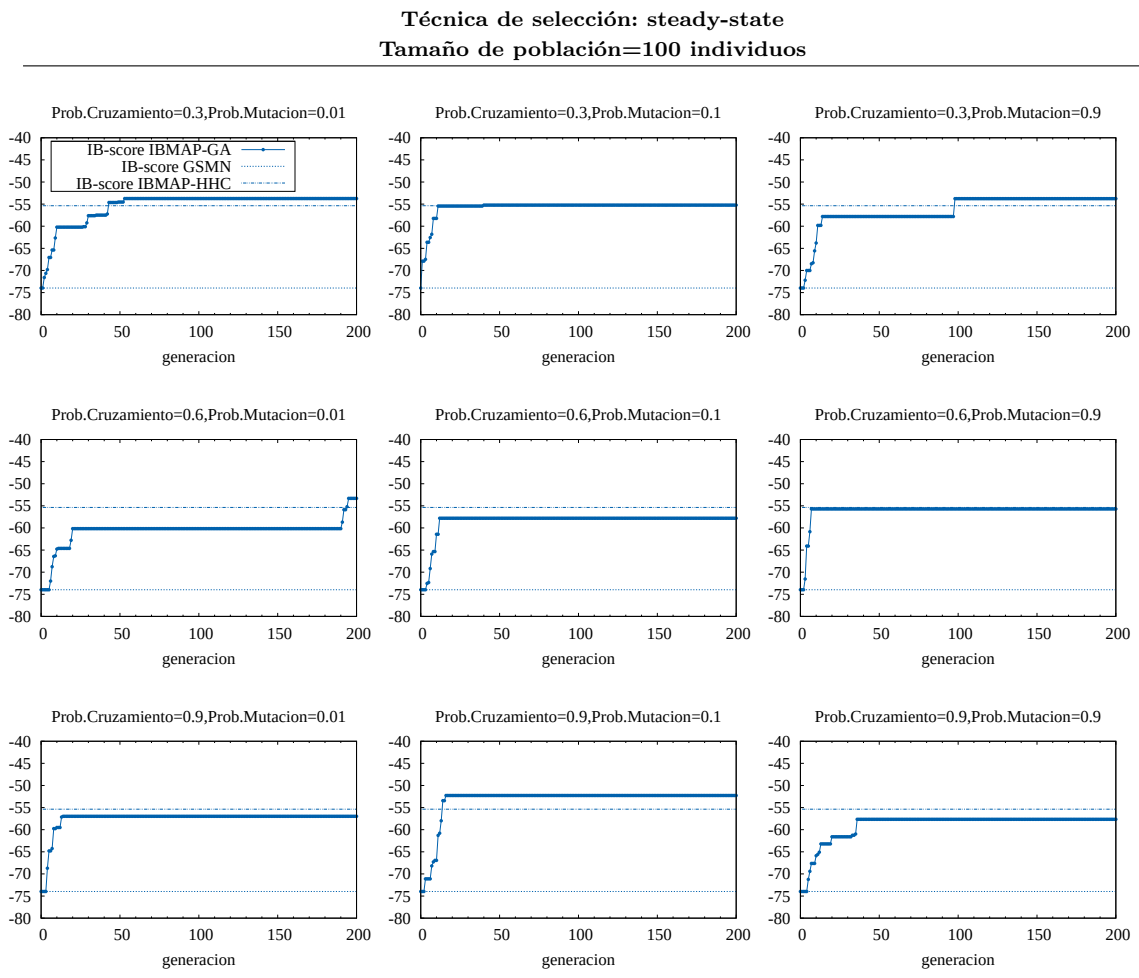

Figura 5.5: Evolución de IBMAP-GA utilizando una población de 100 individuos y selección de supervivientes steady-state. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales.

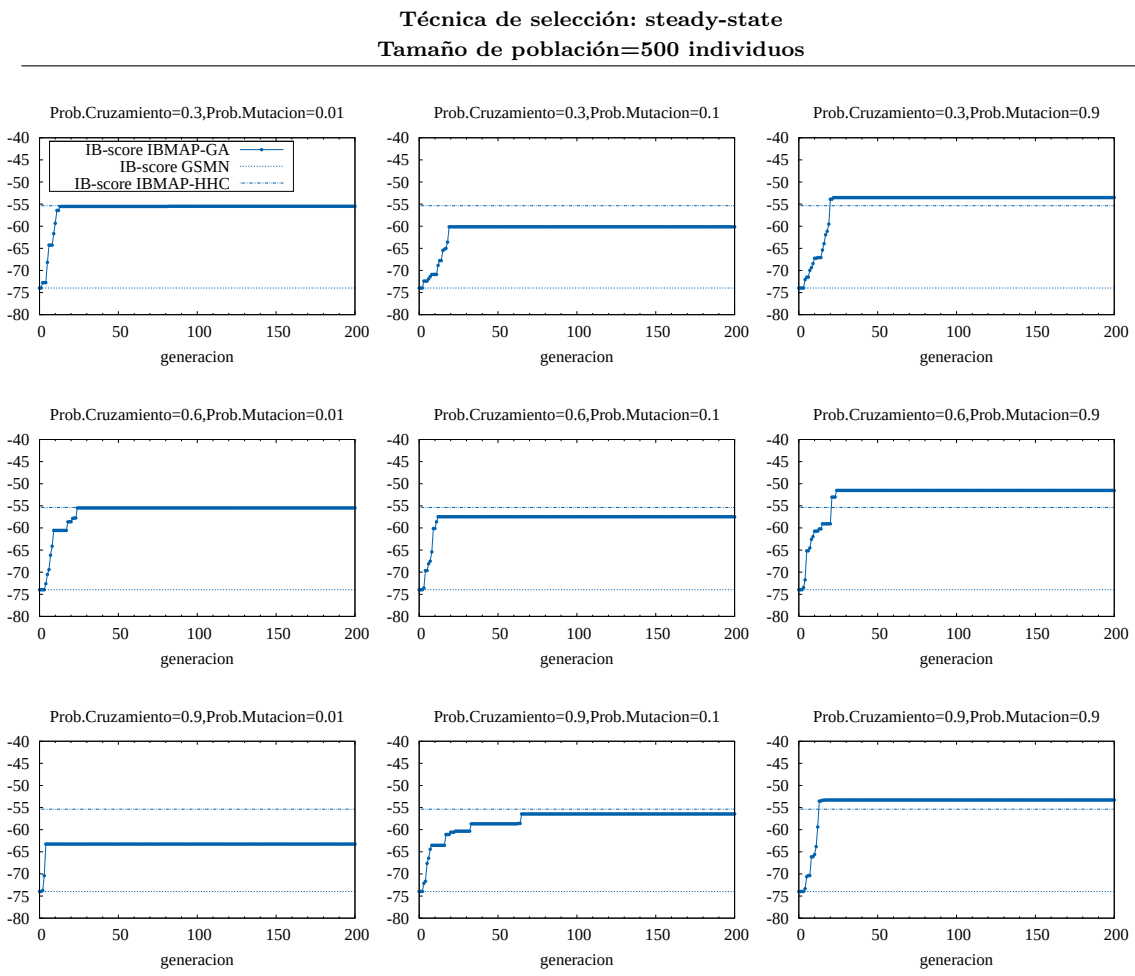

Figura 5.6: Evolución de IBMAP-GA utilizando una población de 500 individuos y selección de supervivientes steady-state. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales.

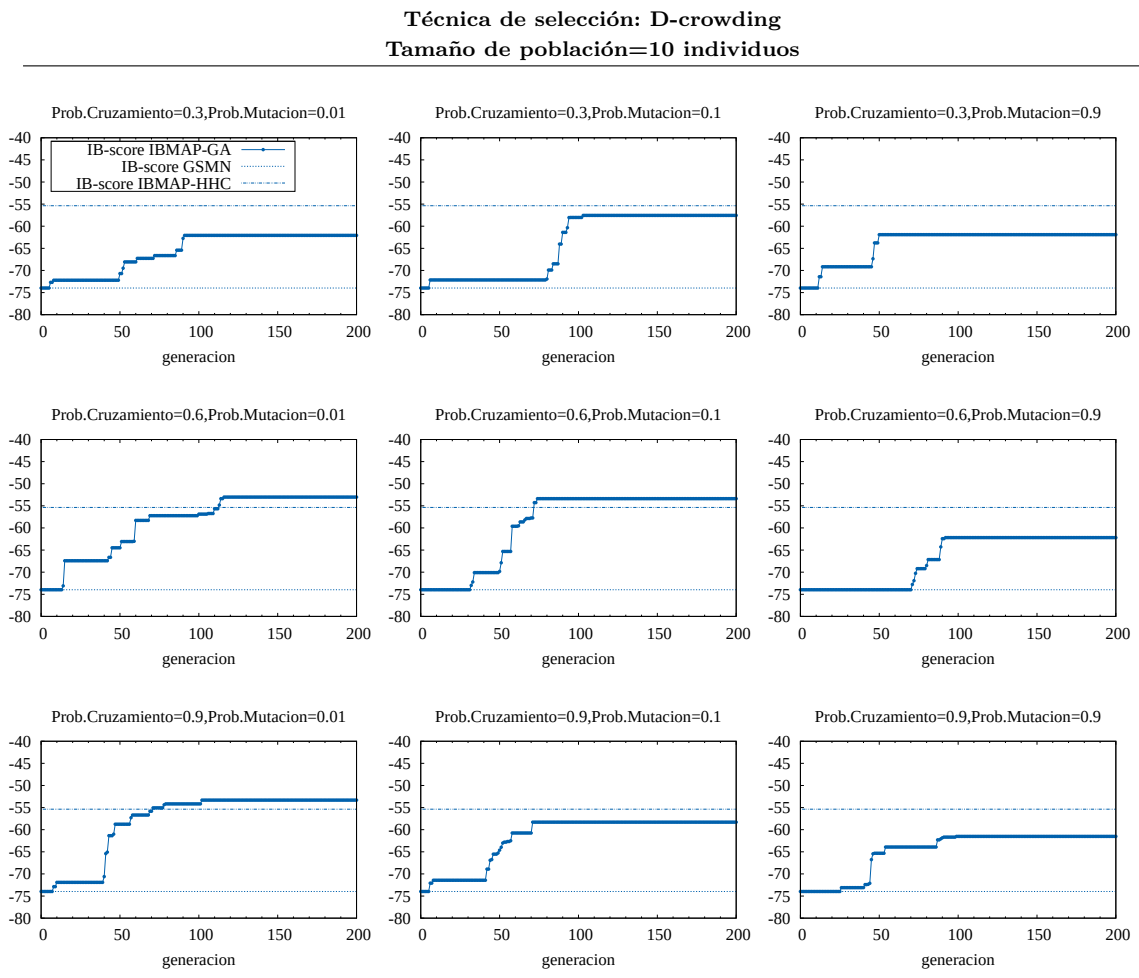

Figura 5.7: Evolución de IBMAP-GA utilizando una población de 10 individuos y selección de supervivientes D-crowding. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales.

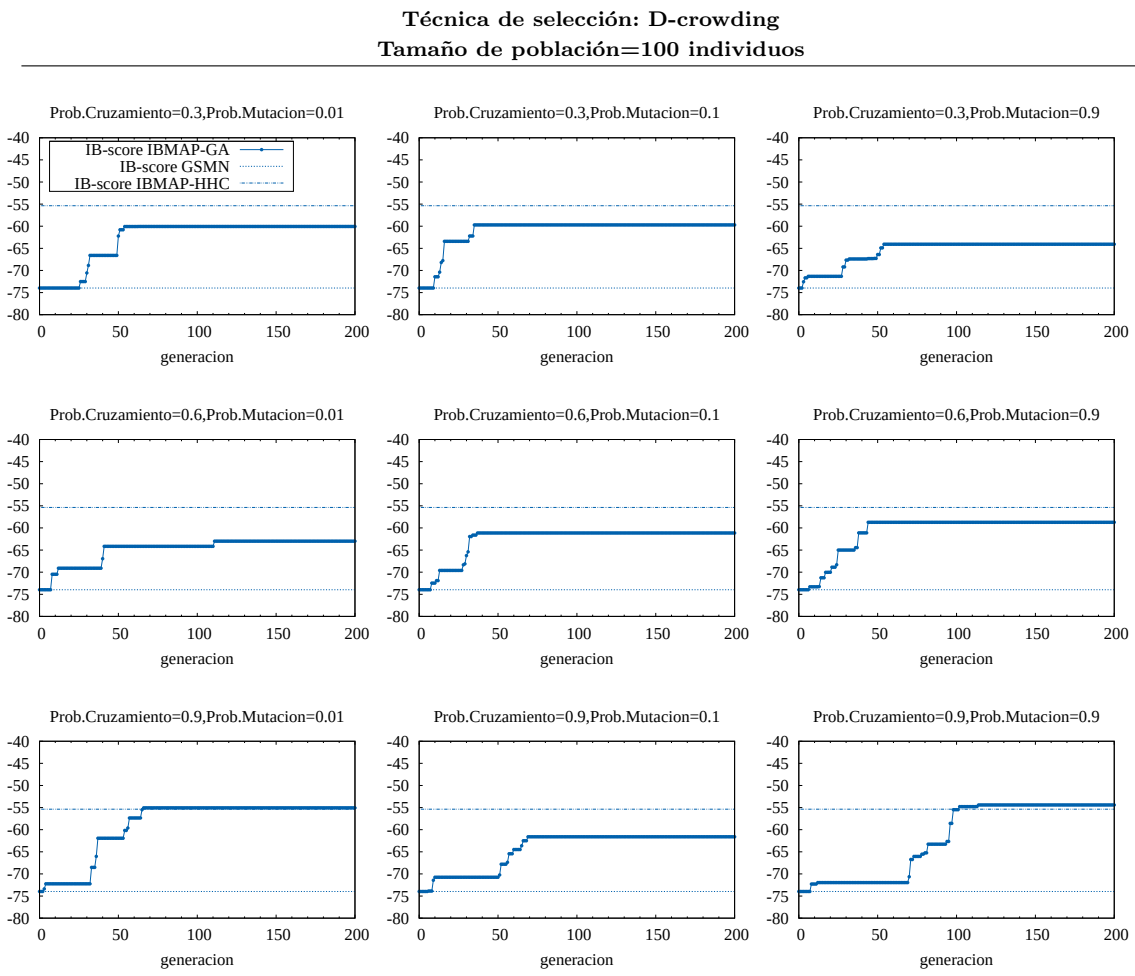

Figura 5.8: Evolución de IBMAP-GA utilizando una población de 100 individuos y selección de supervivientes D-crowding. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales.

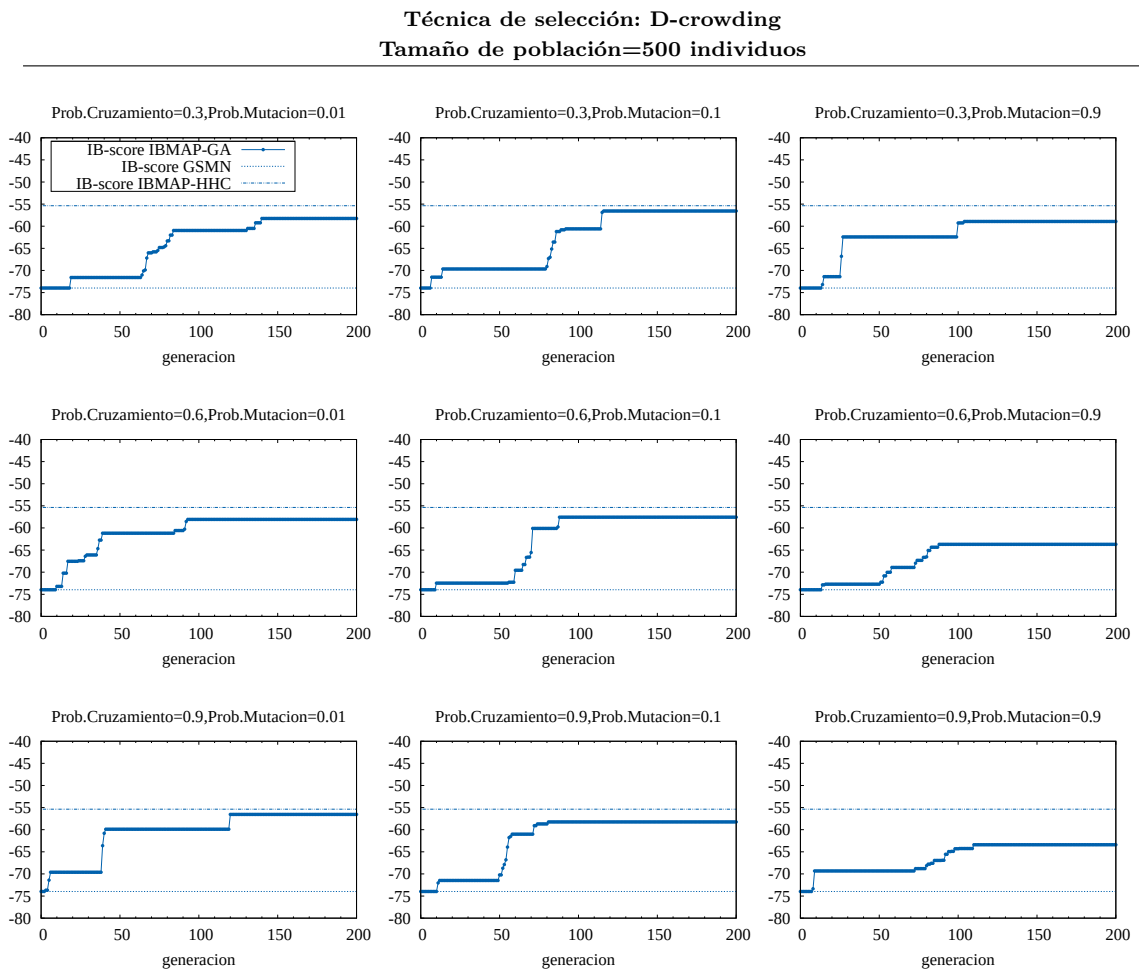

Figura 5.9: Evolución de IBMAP-GA utilizando una población de 500 individuos y selección de supervivientes D-crowding. Se muestra también el IB-score de GSMN y IBMAP-HHC en líneas horizontales.

En dichas figuras puede verse que en todos los casos mostrados IBMAP-GA obtiene estructuras que poseen IB-score más alto que GSMN. Adicionalmente, también se observa que en la gran mayoría de los casos mostrados existe alguna configuración de probabilidad de cruzamiento y de probabilidad de mutación con la que IBMAP-GA supera el IB-score obtenido por IBMAP-HHC. Específicamente, cuando se utiliza steady-state con 10 individuos por población (Figura 5.4), IBMAP-GA converge a un IB-score más alto que IBMAP-HHC en 6 de los 9 casos que muestra la figura. Cuando se utiliza steady-state con 100 individuos por población, IBMAP-GA converge a un IB-score más alto que IBMAP-HHC en 4 de los 9 casos que muestra la Figura 5.5. Cuando se utiliza steady-state con 500 individuos por población, IBMAP-GA converge a un IB-score más alto que IBMAP-HHC en 3 de los 9 casos que muestra la Figura 5.6 (cuando la probabilidad de mutación es 0.9). Cuando se utiliza d-Crowding con 10 individuos por población, IBMAP-GA converge a un IB-score más alto que IBMAP-HHC en 3 de los 9 casos que muestra la Figura 5.7. Cuando se utiliza d-Crowding con 100 individuos por población, IBMAP-GA converge a un IB-score más alto que IBMAP-HHC en sólo 1 caso de los 9 casos que muestra la Figura 5.8, y cuando se utilizan 500 individuos por población, no converge a ningún IB-score más alto que IBMAP-HHC, en la Figura 5.9. Estos resultados permiten validar que IBMAP-GA tiene capacidad de mejorar la maximización del IB-score llevada a cabo por IBMAP-HHC, siempre y cuando se realice un barrido apropiado de sus parámetros de entrada. No se reportan resultados para otros tamaños de dominio debido a que las tendencias son similares.

En un segundo experimento con IBMAP-GA no sólo se analizan los valores de IB-score encontrados, sino también los resultados en términos de calidad de las estructuras a las que IBMAP-GA converge. Para esto, IBMAP-GA fue corrido para cada experimento barriendo sobre distintos valores de probabilidad de mutación (0.01, 0.1 y 0.9), probabilidad de cruzamiento (0.3, 0.6, y 0.9), esquema de selección de supervivientes (steady-state y D-crowding), y tamaño de población (10, 100 y 500). Luego se selecciona la estructura con IB-score más alto, entre aquellas obtenidas utilizando las diferentes parametrizaciones de IBMAP-GA. Para este experimento se utilizaron conjuntos de datos de $n \in \{12, 16, 20\}$ variables binarias, con complejidades crecientes $\tau \in \{1, 2, 4\}$. Se muestran los resultados para IBMAP-GA en comparación con GSMN y con IBMAP-HHC, utilizando tamaños

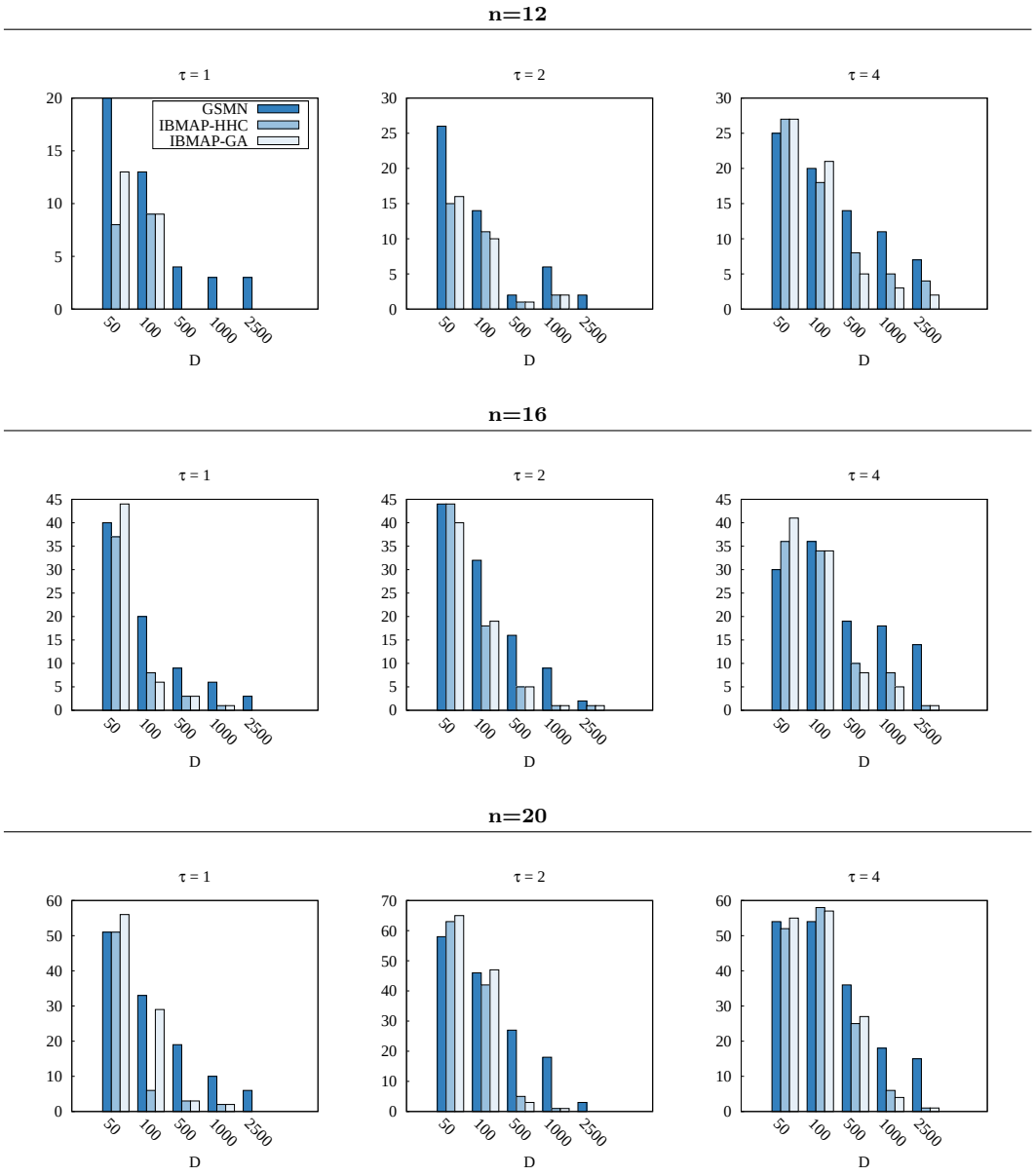

Figura 5.10: Distancia de Hamming en problemas con $n \in \{12, 16, 20\}$ variables para GSMN, IBMAP-HHC y IBMAP-GA. Menor distancia de Hamming es mejor.

de muestra crecientes $D \in \{50, 100, 500, 1000, 2500\}$.

En la Figura 5.10 se muestran los resultados del experimento. Las barras muestran la distancia de Hamming de las estructuras aprendidas por cada algoritmo. A primera vista, puede verse en las gráficas que a medida que crece el tamaño D del conjunto de datos utilizado, tanto IBMAP-HHC como IBMAP-GA presentan mejoras de calidad (menores distancias de Hamming) sobre GSMN. Adicionalmente, también puede observarse que IBMAP-GA mejora la calidad de IBMAP-HHC en los siguientes casos:

- $n = 12, \tau = 2, D = 100$
- $n = 12, \tau = 4, D = 500$
- $n = 12, \tau = 4, D = 1000$
- $n = 12, \tau = 4, D = 2500$
- $n = 16, \tau = 1, D = 100$
- $n = 16, \tau = 2, D = 50$
- $n = 16, \tau = 4, D = 500$
- $n = 16, \tau = 4, D = 1000$
- $n = 20, \tau = 2, D = 500$
- $n = 20, \tau = 4, D = 1000$.

El panorama general de estos resultados indica que IBMAP-GA y IBMAP-HHC mejoran la calidad de GSMN en todos los casos, y en general las calidades estructurales de IBMAP-GA son comparables a las obtenidas por IBMAP-HHC (presentando empates en la mayoría de los casos). Esto permite sustentar la hipótesis de que IBMAP-HHC es una solución práctica en términos de calidad y eficiencia, ya que sin utilizar parametrizaciones complejas y sólo convergiendo a un máximo local, tiene capacidad de obtener soluciones de calidad muy similares a las de IBMAP-GA, que explora mucho más aún el espacio de estados, y que requiere que el algoritmo se corra sobre distintos barridos de sus parámetros de entrada. Sin embargo, como ya se fundamentó previamente, IBMAP-GA resulta una alternativa útil e interesante para los casos donde se está dispuesto a invertir tiempo de cómputo a cambio de la mejor calidad posible.

5.3. Evaluación de escalabilidad de la calidad estructural sobre la dimensionalidad de las distribuciones

Esta sección muestra un conjunto de experimentos sobre datos sintéticos para distribuciones de mayor tamaño que las utilizadas en los experimentos de las secciones anteriores. Por esto, dado que el algoritmo más eficiente en términos de complejidad temporal y en la calidad de las estructuras obtenidas es IBMAP-HHC, en esta sección se muestra una comparación de su calidad respecto de los algoritmos competidores GSMN y HHC-MN, a fin de mostrar cómo evoluciona el potencial de mejora de calidad del enfoque IBMAP a medida que crece el tamaño de las distribuciones.

Los conjuntos de datos sintéticos utilizados se generaron del mismo modo que los utilizados en las secciones anteriores. Para este experimento, se generaron redes de Markov sintéticas de tamaños crecientes con $n \in \{50, 100, 200, 500, 750\}$ variables binarias. Para cada tamaño de dominio, se consideraron distintos niveles de conectividad creciente: $\tau \in \{1, 2, 4, 8\}$ (es decir, se agregó un nivel de complejidad adicional). Para cada conjunto de datos se generaron 3200 puntos de datos, y posteriormente se ejecutaron los algoritmos GSMN, HHC-MN y IBMAP-HHC sobre diferentes subconjuntos de tamaño incremental del conjunto de datos $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$.

Los resultados de este experimento se muestran en las Figuras 5.11, 5.12, 5.13, 5.14, y 5.15, para los dominios de $n \in \{50, 100, 200, 500, 750\}$ variables respectivamente. En dichas figuras se muestran los promedios de las distancias de Hamming de cada algoritmo sobre diez conjuntos de datos diferentes y su desviación estándar. La figura ordena las gráficas disponiendo los resultados para las distintas conectividades crecientes τ en las distintas filas de la figura. De este modo, pueden visualizarse verticalmente las tendencias de calidad a medida que crece la dificultad de las estructuras subyacentes. Asimismo, en cada figura puede observarse sobre el eje X la evolución de la distancia de Hamming de los algoritmos a medida que crece el tamaño D del conjunto de datos de entrenamiento utilizado.

Al analizar los resultados de la Figura 5.11 puede verse que para $n = 50$

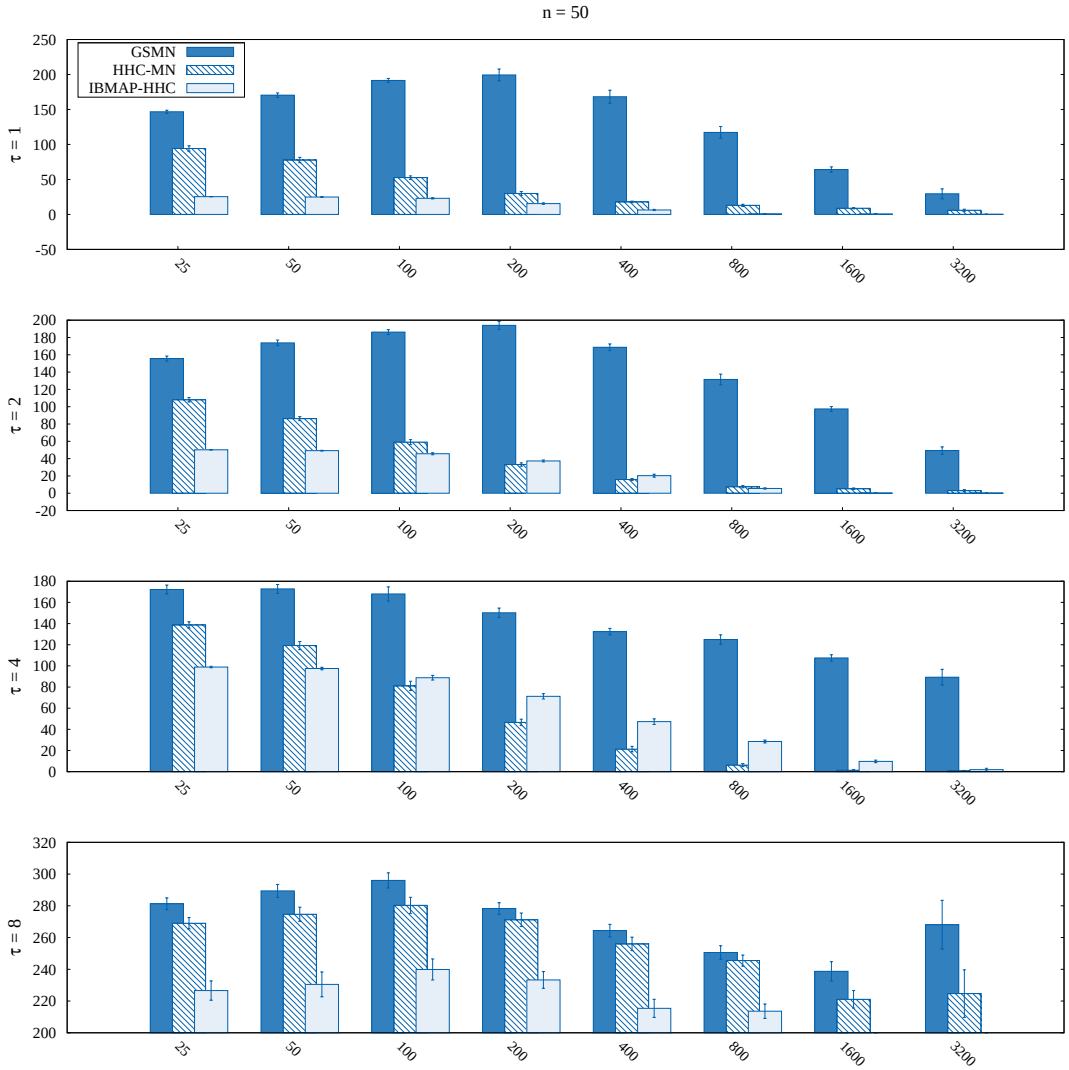

Figura 5.11: Distancia de Hamming de problemas con $n = 50$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). Menor distancia de Hamming es mejor.

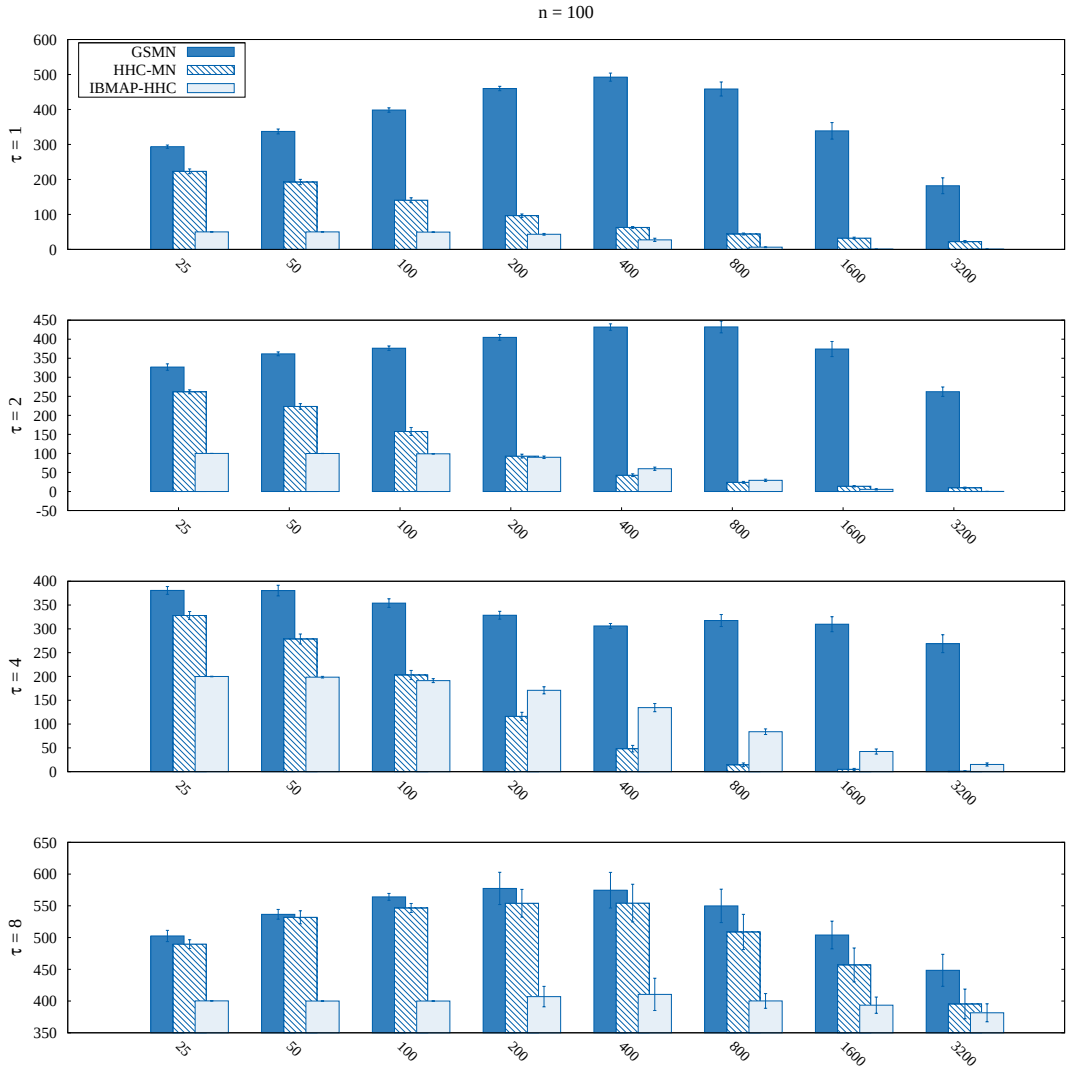

Figura 5.12: Distancia de Hamming de problemas con $n = 100$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). Menor distancia de Hamming es mejor.

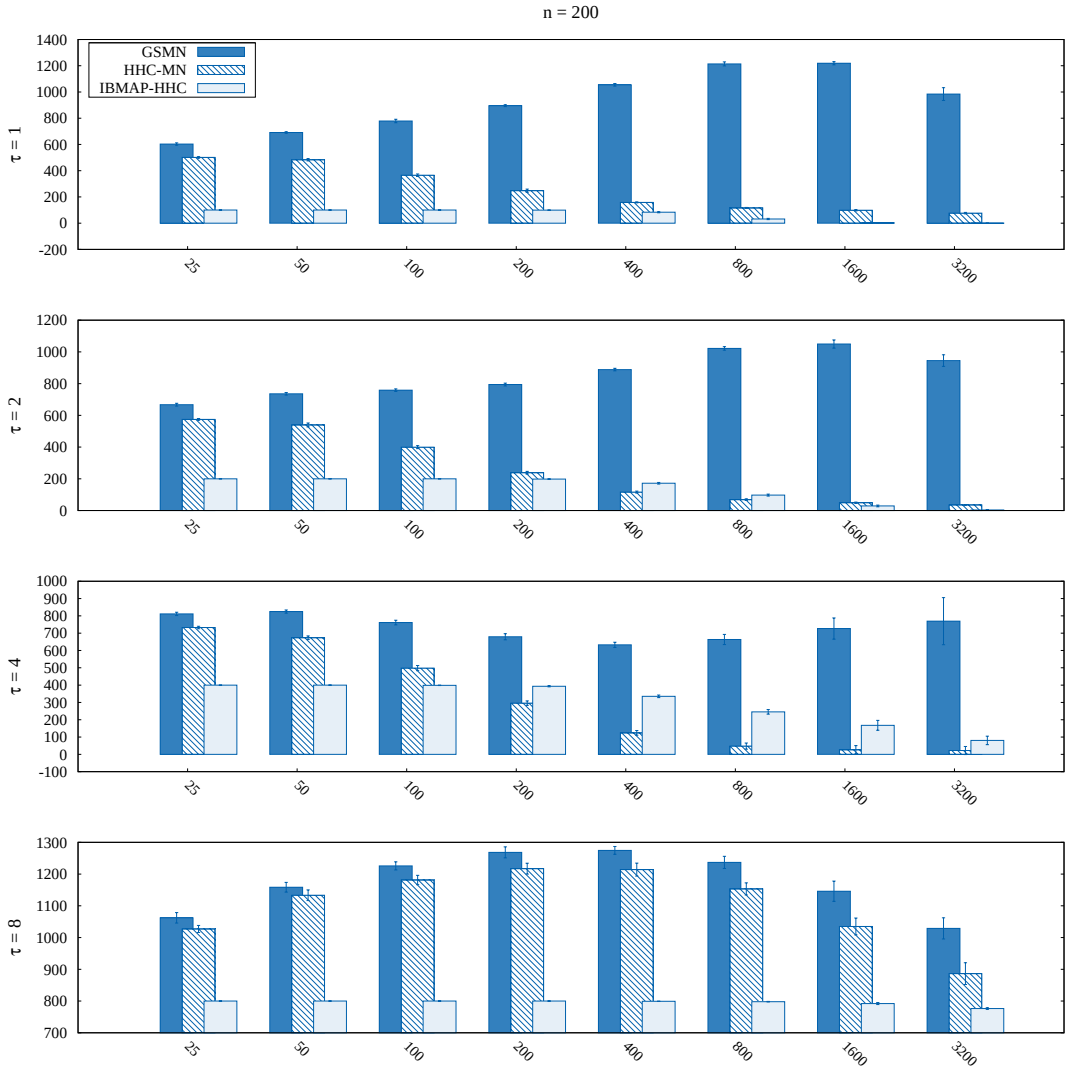

Figura 5.13: Distancia de Hamming de problemas con $n = 200$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). Menor distancia de Hamming es mejor.

variables binarias, en el caso de $\tau = 1$, GSMN muestra la convergencia más lenta en D, cometiendo alrededor de 200 errores estructurales en los peores casos, y aproximadamente 40 errores en el mejor caso. HHC-MN muestra la segunda mejor convergencia, cometiendo alrededor de 90 errores en el peor caso, y convergiendo a una media de 10 errores en el mejor caso. La mejor convergencia en D la muestra IBMAP-HHC, aprendiendo estructuras con aproximadamente 20 errores estructurales en los peores casos ($D \leq 100$), y reduciendo estos errores hasta 0 cuando $D = 3200$. Para los casos de $\tau = 2$ y $\tau = 4$ se observan tendencias similares, mostrando mejoras respecto de GSMN más contundentes aún. Respecto de HHC-MN las mejoras contundentes se presentan en los casos en que los datos son escasos ($D < 100$). Este resultado es esperable, ya que se trata de estructuras no muy densas, y la cantidad de datos es suficiente como para que su estrategia de eliminación intercalada sea realmente efectiva (ver el Apéndice C). Para el caso de $\tau = 8$, la complejidad del problema de aprendizaje es mucho mayor, lo que se ve reflejado en los resultados, ya que como puede verse en el eje Y, se incrementan las cantidades de errores estructurales obtenidas por todos los algoritmos. En este caso, IBMAP-HHC muestra la mejor convergencia para todos los casos de D.

Al analizar los resultados de las Figuras 5.12, 5.13, 5.14, y 5.15 puede verse que para $n \in \{100, 200, 500, 750\}$ variables binarias, los resultados muestran las mismas tendencias, pero el impacto del efecto cascada es mucho más importante a medida que crece n. Por ejemplo, en el caso de $n = 750$, $\tau = 1$ y $D = 25$ en la Figura 5.15 GSMN y HHC-MN cometen alrededor de 2000 errores estructurales, y IBMAP-HHC no llega a los 200 errores. Esto se ve agravado a medida que incrementa τ. Por ejemplo, en el caso de $n = 750$, $\tau = 8$ y $D = 400$, GSMN y HHC-MN cometen alrededor de 4000 errores estructurales, mientras que IBMAP-HHC comete aproximadamente unos 3000 errores. Las mejores ganancias de calidad de IBMAP-HHC respecto de GSMN pueden verse para $n = 750$, $\tau = 1$ y $D = 3200$, donde GSMN comete más de 5000 errores, y IBMAP-HHC comete alrededor de 100 errores. Respecto de HHC-MN, las mejores ganancias de calidad de IBMAP-HHC pueden verse para los casos donde los datos son muy escasos, es decir, $D < 100$.

Para facilitar un análisis a nivel global de todos los resultados mostrados, en la Figura 5.16 se comprimen todos en una única gráfica. Esta figura organiza los resultados disponiendo los resultados para distintos tamaños de distribuciones

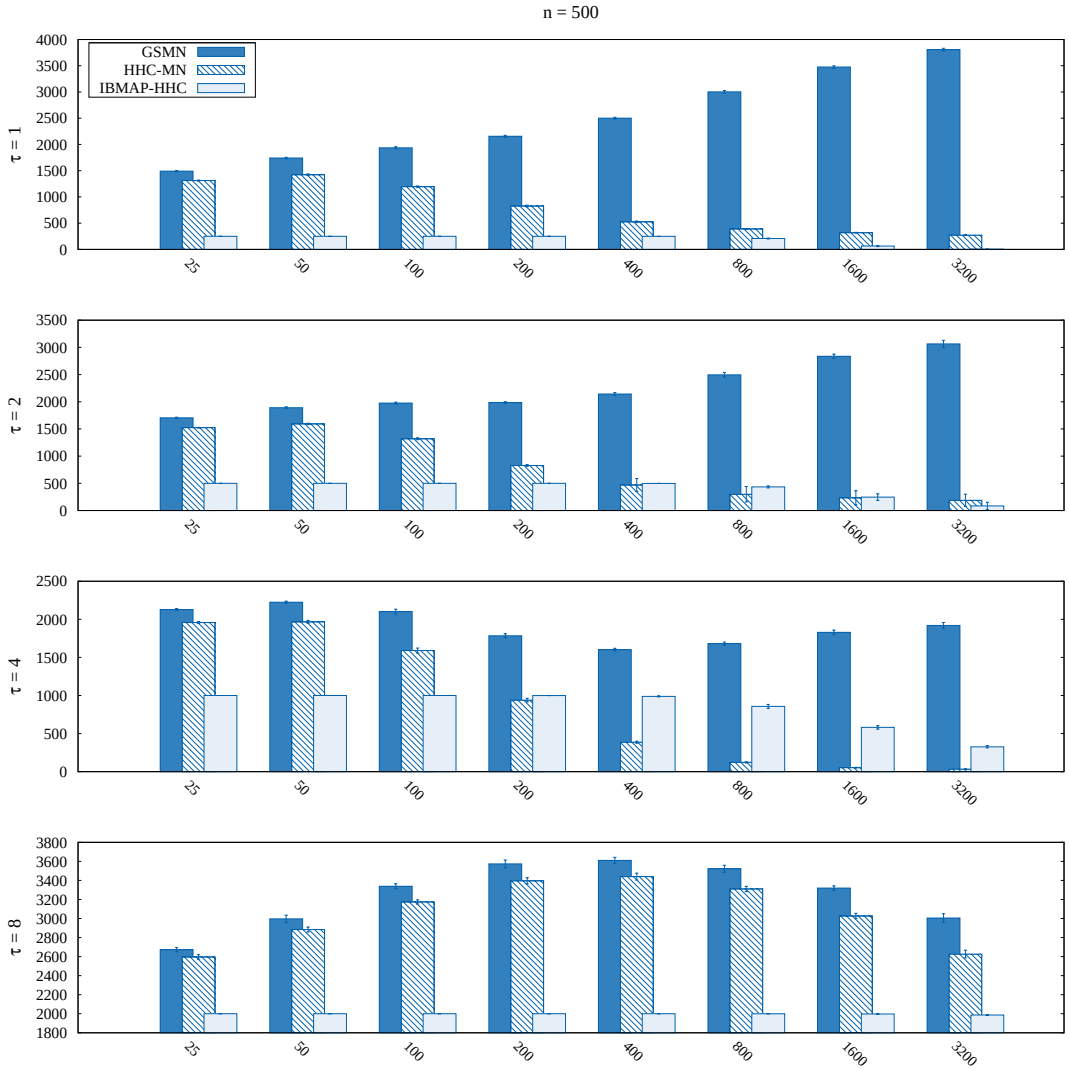

Figura 5.14: Distancia de Hamming de problemas con $n = 500$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). Menor distancia de Hamming es mejor.

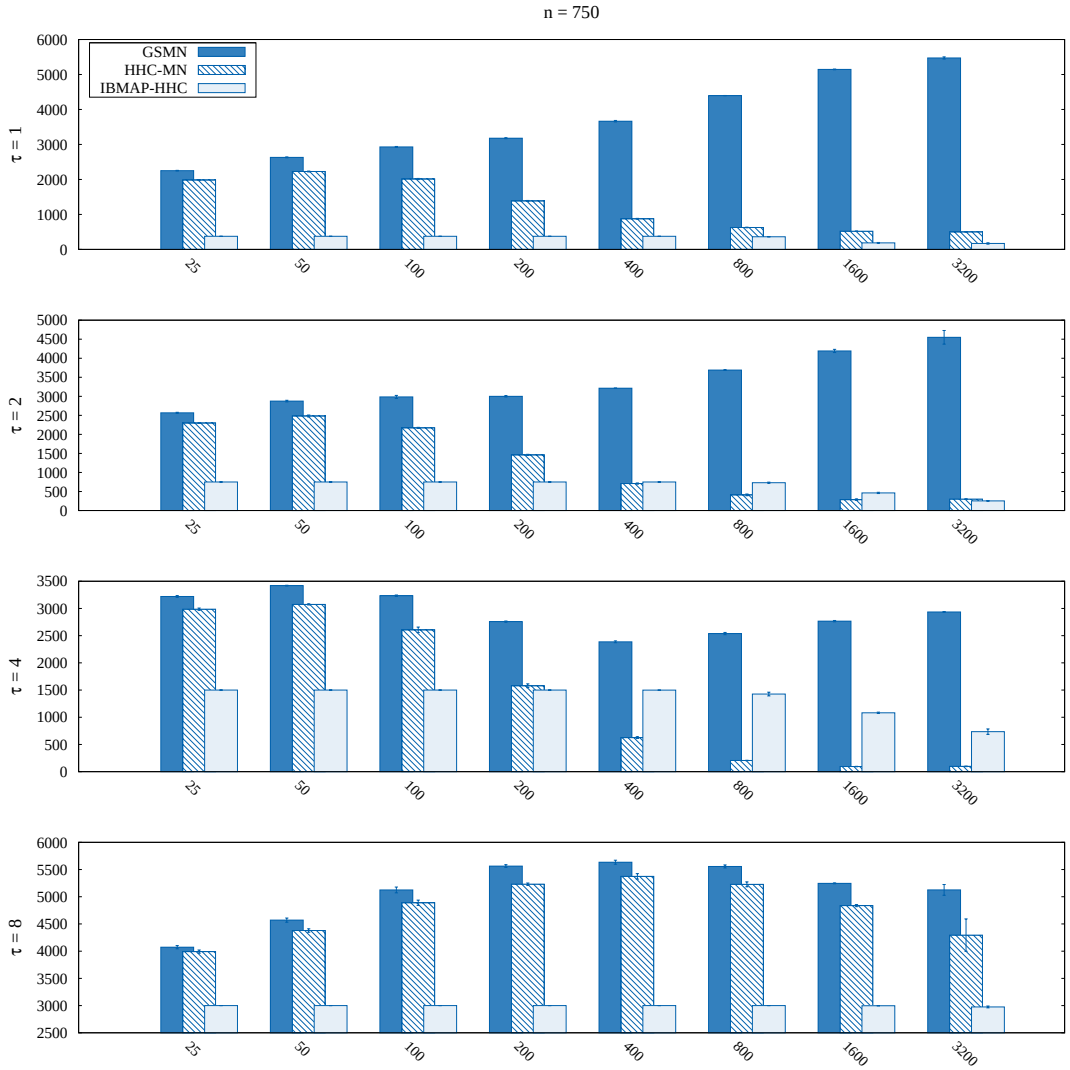

Figura 5.15: Distancia de Hamming de problemas con $n \in \{750\}$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas). Menor distancia de Hamming es mejor.

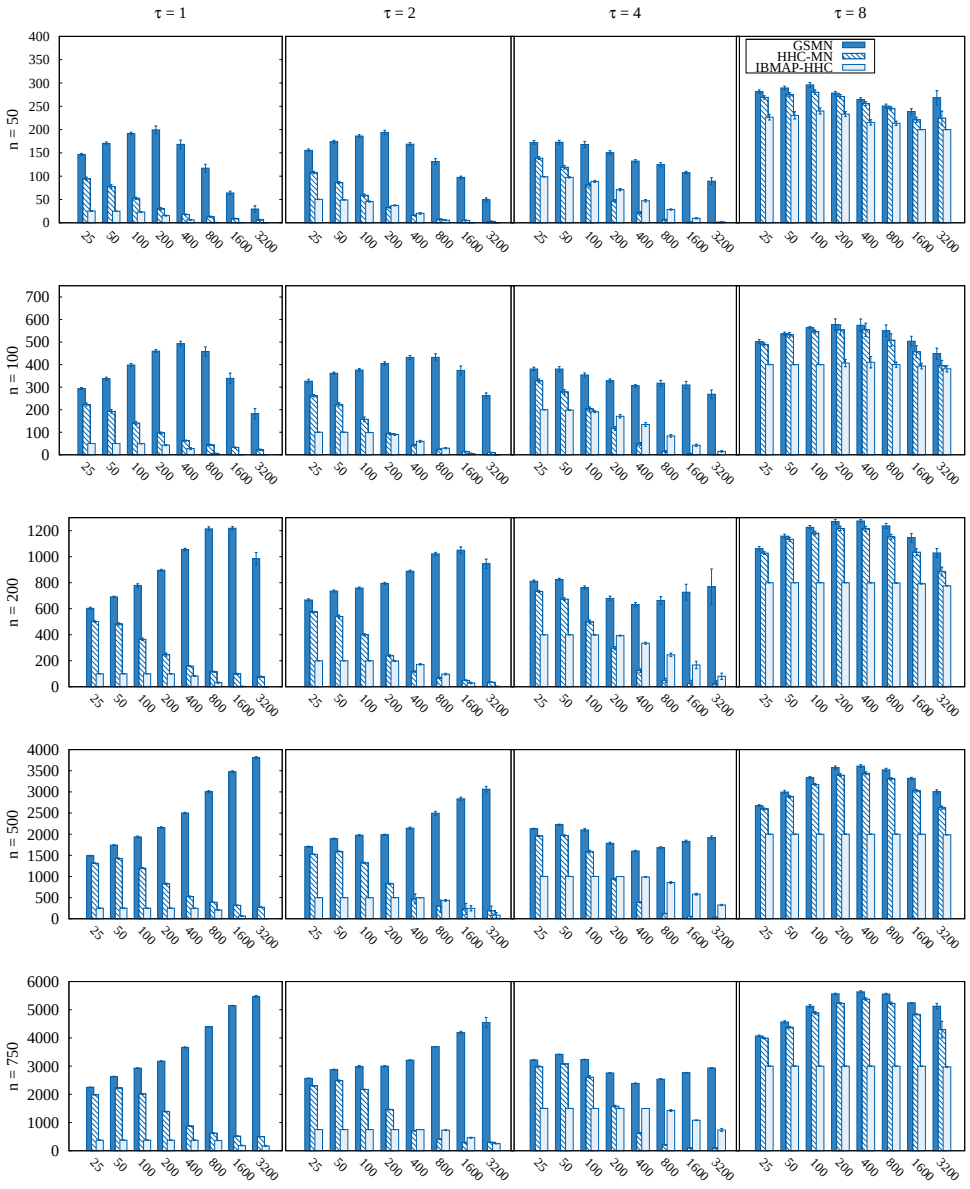

Figura 5.16: Distancia de Hamming de problemas con $n \in \{50, 100, 200, 500, 750\}$ variables (filas) para estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (columnas). Menor distancia de Hamming es mejor.

$n \in \{50, 100, 200, 500, 750\}$ en las filas de una grilla, y para los distintos tamaños de conectividad creciente $\tau \in \{1, 2, 4, 8\}$ en las distintas columnas. De este modo, puede verse verticalmente que hay tendencias a empeorar la calidad a medida que crece n para una misma dificultad de estructuras (τ en la columna). Asimismo, también puede revisarse horizontalmente cómo para un mismo valor de n ocurren tendencias en la calidad de las estructuras aprendidas, a medida que se incrementa la dificultad de las estructuras τ. Al igual que los resultados vistos en la Sección 5.1, estos resultados muestran que para todos los algoritmos, mientras más compleja es la estructura subyacente (es decir, mientras crece τ), más grande es la distancia de Hamming de las estructuras aprendidas respecto de la solución. Puede verse también que para cualquier valor fijo de D, la cantidad de errores de cada algoritmo crece con τ. En la Sección 5.5 se muestran los resultados de tiempo de corrida de este experimento, y junto con otros experimentos que evalúan la complejidad temporal de IBMAP-HHC, se demuestra que la complejidad temporal de IBMAP-HHC es altamente competitiva respecto de GSMN y HHC-MN.

5.4. Evaluación de calidad estructural en datos reales

En esta sección se muestran resultados experimentales sobre conjuntos de datos del mundo real, tomados desde los repositorios UCI de aprendizaje de máquinas (Asuncion y Newman, 2007) y conjuntos de datos para descubrimiento de conocimiento (Hettich y Bay, 1999). Debido a que en estos conjuntos de datos no se conoce la red solución, no es posible analizar la calidad de las estructuras aprendidas mediante la distancia de Hamming. Por esto se utiliza *accuracy*, que es una medida de calidad que contabiliza el número de independencias condicionales que están presentes en los datos y que además se han codificado correctamente en una estructura aprendida. Esta medida de calidad ha sido utilizada con este mismo propósito en otros trabajos relacionados (Bromberg et al., 2009; Margaritis y Bromberg, 2009; Bromberg y Margaritis, 2009). En contraste con otras medidas que evalúan la densidad de la distribución de probabilidades completa (e.g. el Conditional Marginal Log-Likelihood), la accuracy está más orientada a evaluar

la calidad cuando el objetivo del aprendizaje es hallar la estructura de independencias correcta, ya que la misma evalúa específicamente errores estructurales. La accuracy se define como una medida de calidad normalizada para contabilizar las independencias que se cumplen en un conjunto de datos *de testeo*, y que también se cumplen en la estructura aprendida desde el conjunto de datos *de entrenamiento*. Las independencias condicionales se leen desde la estructura aprendida utilizando separación de vértices (ver la Sección 2.1.1).

La accuracy se computa del siguiente modo. Si denotamos $\mathcal{T}$ al conjunto que contiene a todas las posibles preguntas de independencia que se puede hacer sobre el dominio $\mathbf{V}$, la accuracy contabiliza para cuántas preguntas $t \in \mathcal{T}$, se cumple que t es independiente (o dependiente) tanto en el conjunto de datos de testeo como en la estructura aprendida desde el conjunto de datos de entrenamiento. Luego, el número de coincidencias se normaliza por $|\mathcal{T}|$. Desafortunadamente, como el tamaño de $\mathcal{T}$ es exponencial en el tamaño del dominio, la accuracy aproximada se computa sobre un subconjunto $\widehat{\mathcal{T}}$ muestreado aleatoriamente, con una distribución uniforme para cada posible tamaño del conjunto condicionante. En nuestros experimentos utilizamos $|\widehat{\mathcal{T}}| = 100 \times \binom{n}{2}$, es decir, cien preguntas de independencia por cada tamaño posible del conjunto condicionante.

El experimento fue llevado a cabo utilizando 19 conjuntos de datos reales, que están listados en la columna 1 de la Tabla 5.1. Los conjuntos de datos están ordenados por el tamaño del dominio (n) en la segunda columna de la tabla. Para cada conjunto de datos, se realizó una división aleatoria del mismo en un conjunto de datos de entrenamiento para realizar el aprendizaje de la estructura (75 %), y un conjunto de datos de testeo para realizar el cómputo de la accuracy (25 %). La tabla también muestra información sobre la cantidad de puntos de datos disponibles en los conjuntos de datos de entrenamiento y de testeo (tercera y cuarta columna, respectivamente). Para cada conjunto de datos se corrieron los algoritmos GSMN, HHC-MN y IBMAP-HHC, y se computó la accuracy de la estructura obtenida para cada algoritmo. En las columnas 5, 6 y 7 de la Tabla 5.1 puede verse en negrita el mejor resultado entre los tres algoritmos.

Como puede verse, en 10 de los 19 conjuntos de datos utilizados, IBMAP-HHC resultó en una mejor accuracy, en 6 casos se resultó en empate (2 con GSMN, 1 con HHC-MN, y 3 con ambos), y para los casos restantes, los mejores resultados fueron obtenidos por HHC-MN (2 casos) y por GSMN (1 caso). Los casos donde

Conjuntos de datos	n	Datos de entrenamiento	Datos de testeo	accuracy		
				GSMN	HHC-MN	IBMAP-HHC
baloons	5	14	5	**0.950**	0.897	**0.950**
balance-scale	5	468	156	**0.516**	**0.516**	**0.516**
iris	5	112	37	0.695	**0.742**	0.736
lenses	5	17	6	**0.881**	0.875	**0.881**
hayes-roth	6	98	33	**0.516**	**0.516**	**0.516**
car	7	1295	432	0.629	0.641	**0.703**
monks-1	7	416	139	**0.905**	**0.905**	**0.905**
nursery	9	9719	3240	0.392	0.415	**0.649**
ecoli	9	251	84	0.523	0.591	**0.694**
machine	10	156	52	0.590	0.567	**0.679**
cmc	10	1104	368	**0.759**	0.711	0.726
tic-tac-toe	10	718	239	0.671	**0.684**	0.498
echocardiogram	13	45	15	0.696	**0.745**	**0.745**
crx	16	489	163	0.578	0.593	**0.609**
hepatitis	20	59	20	0.496	0.633	**0.796**
imports-85	25	144	28	0.368	0.377	**0.596**
flag	29	145	48	0.446	0.451	**0.803**
dermatology	35	268	53	0.234	0.265	**0.754**
bands	38	207	69	0.399	0.408	**0.546**

Tabla 5.1: Accuracy sobre diversos conjuntos de datos reales. La estructura se aprende utilizando un subconjunto del 75 % llamado conjunto de entrenamiento, y la accuracy se computa utilizando el 25 % restante (datos de testeo). Para cada conjunto de datos, se indica en negrita el mejor resultado.

IBMAP-HHC siempre obtiene mejores calidades que sus competidores son para aquellos donde $n \geq 16$. En estos casos, los datos parecen ser escasos (como puede apreciarse en la tercera columna). Esto es consistente con los resultados mostrados para conjuntos de datos sintéticos, donde IBMAP-HHC siempre mejora sobre sus competidores en los casos donde los datos son escasos, y las mejoras se hacen más contundentes a medida que n es mayor.

5.5. Análisis de complejidad temporal sobre datos sintéticos

Esta sección reporta los resultados de tiempo de corrida del experimento descripto en la Sección 5.3, a fin de evaluar si el costo computacional de IBMAP-HHC es competitivo respecto de los competidores elegidos. Adicionalmente, se describe un conjunto de experimentos que muestran que la cantidad de ascensos M

requeridos por la búsqueda local para converger depende tanto de la complejidad del problema (n y τ), como de la cantidad de datos disponibles (D). En estos resultados se muestra que el valor de M crece en todos los casos linealmente o sub-linealmente, demostrando empíricamente que M no es una fuente de complejidad extra en este tipo de búsquedas.

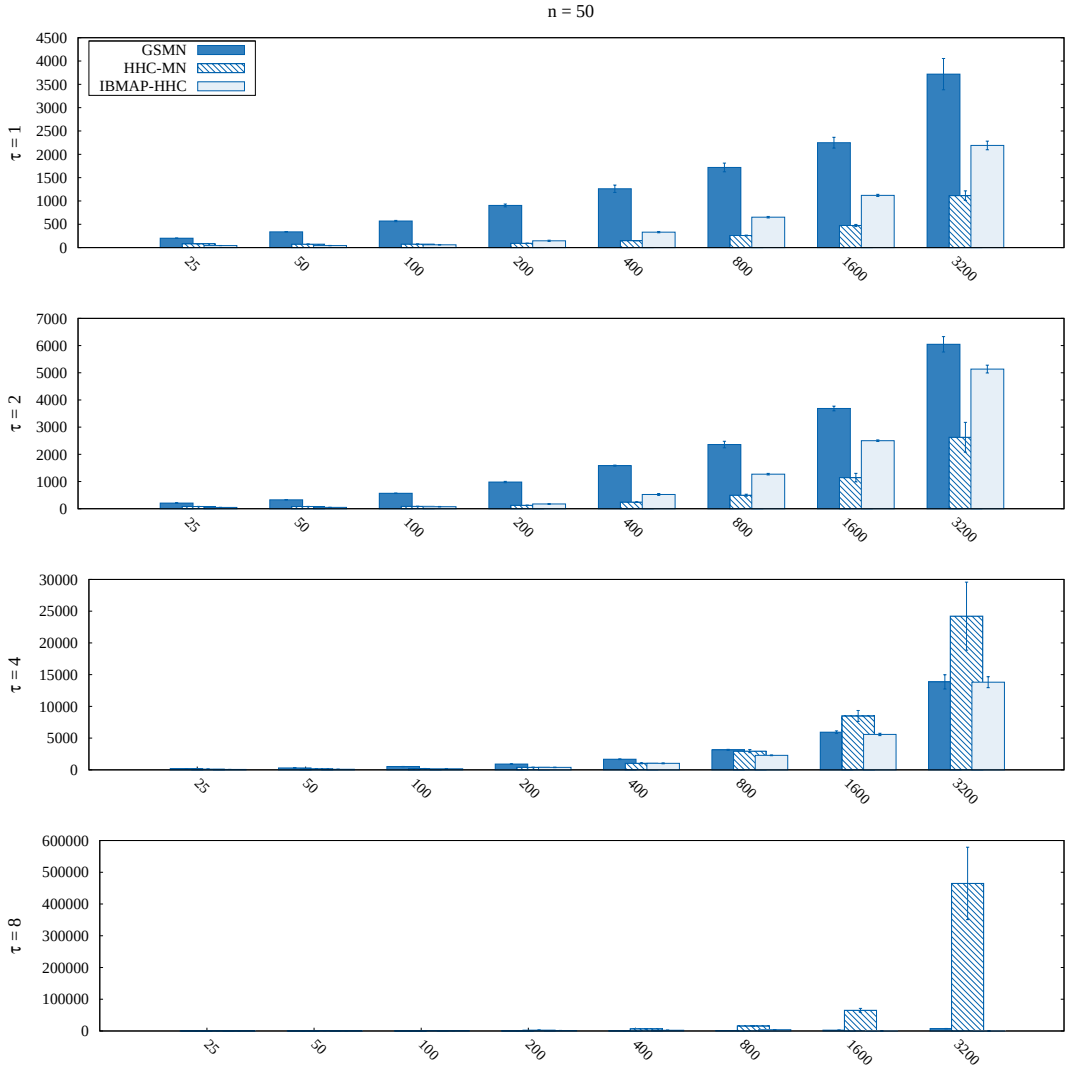

Figura 5.17: Tiempo de corrida para problemas con $n = 50$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas).

En una primera instancia los resultados de tiempo de corrida de IBMAP-HHC,

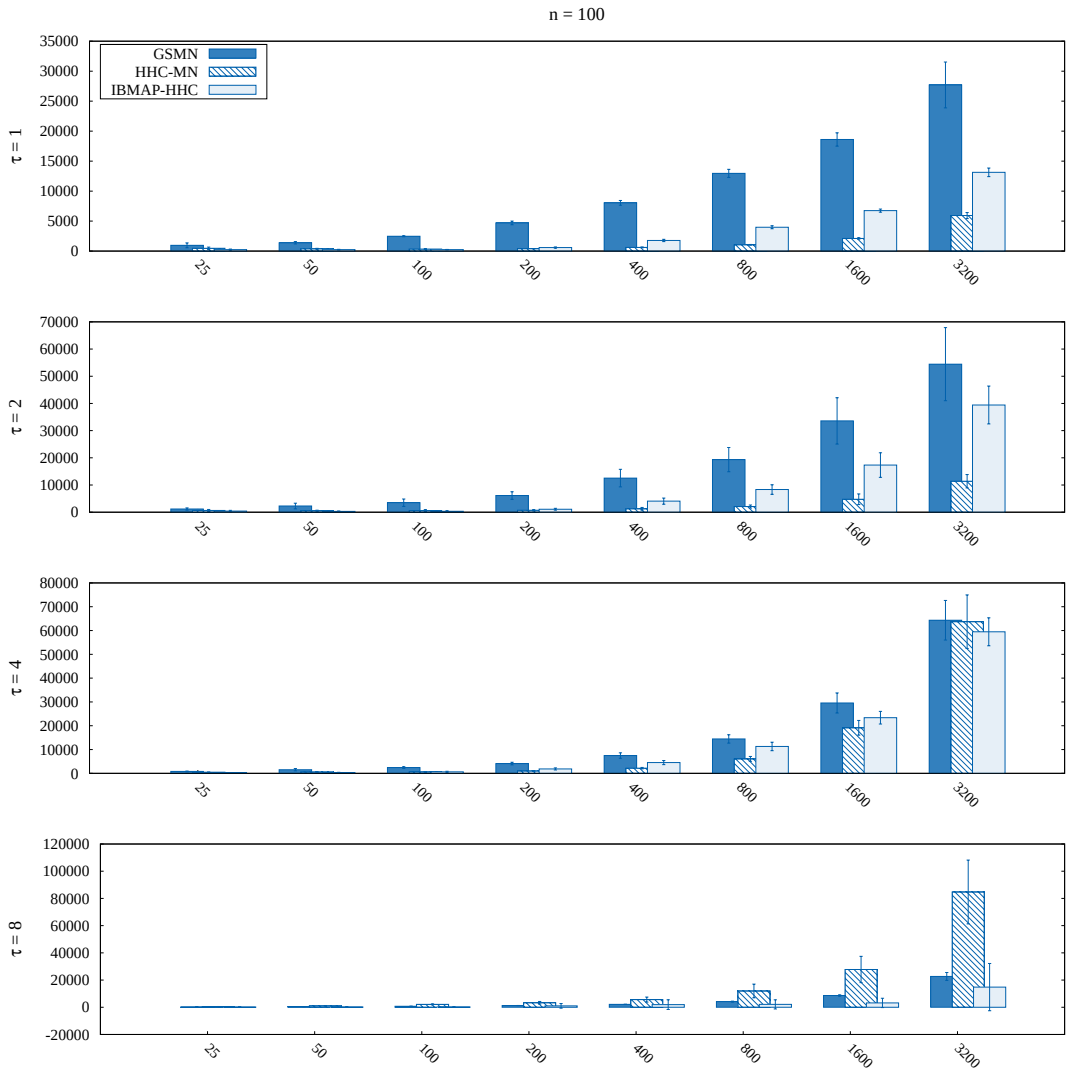

Figura 5.18: Tiempo de corrida para problemas con $n = 100$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas).

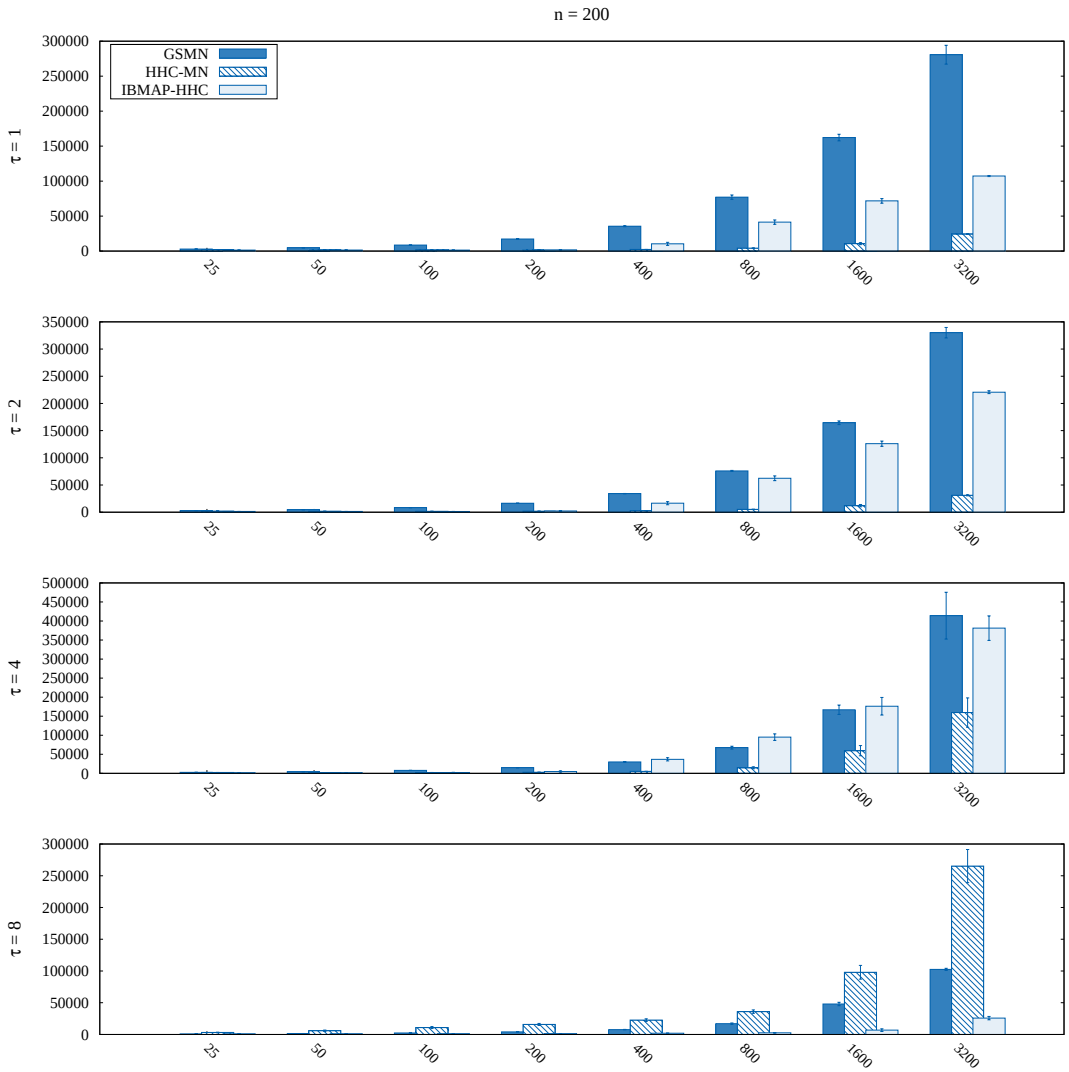

Figura 5.19: Tiempo de corrida para problemas con $n = 200$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas).

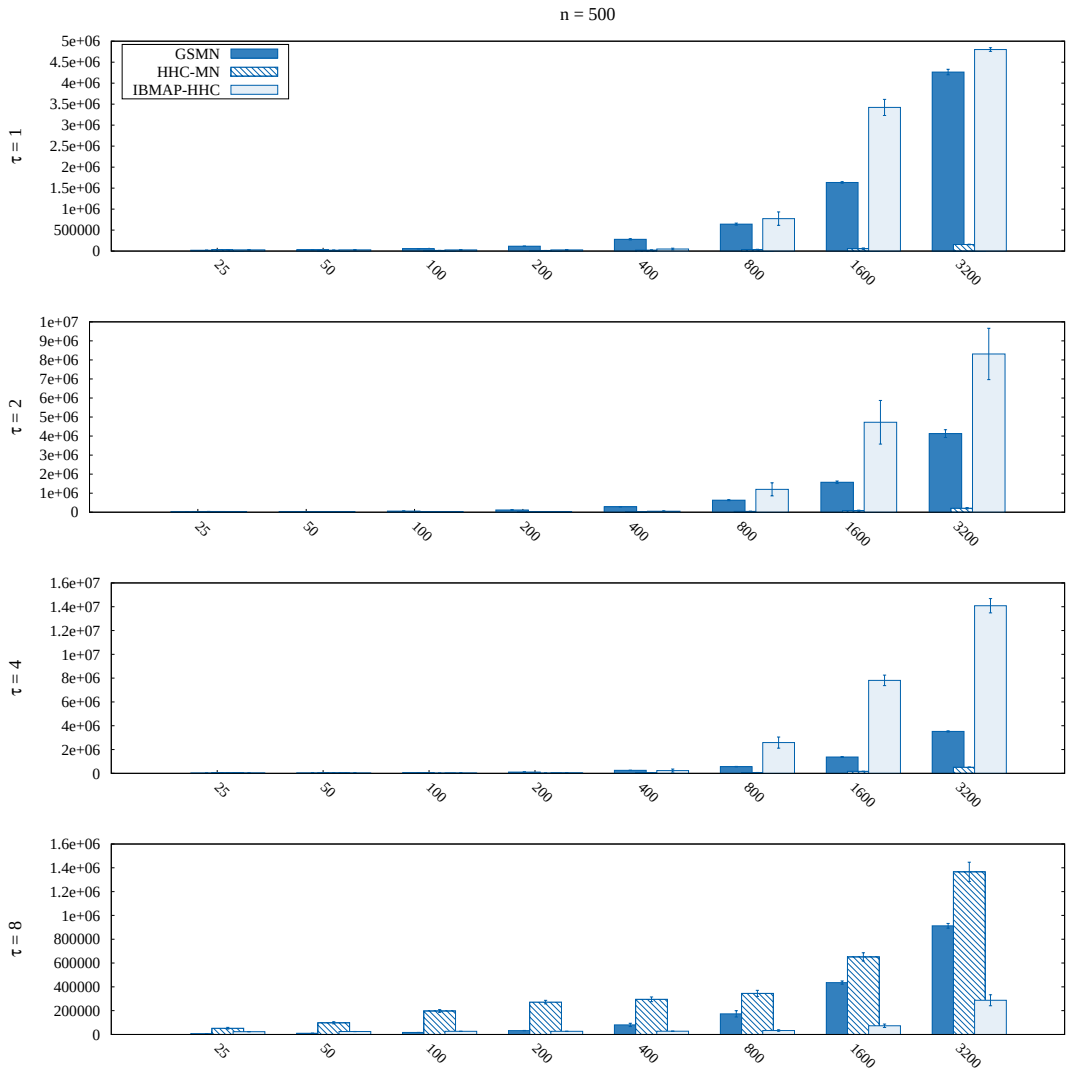

Figura 5.20: Tiempo de corrida para problemas con $n = 500$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas).

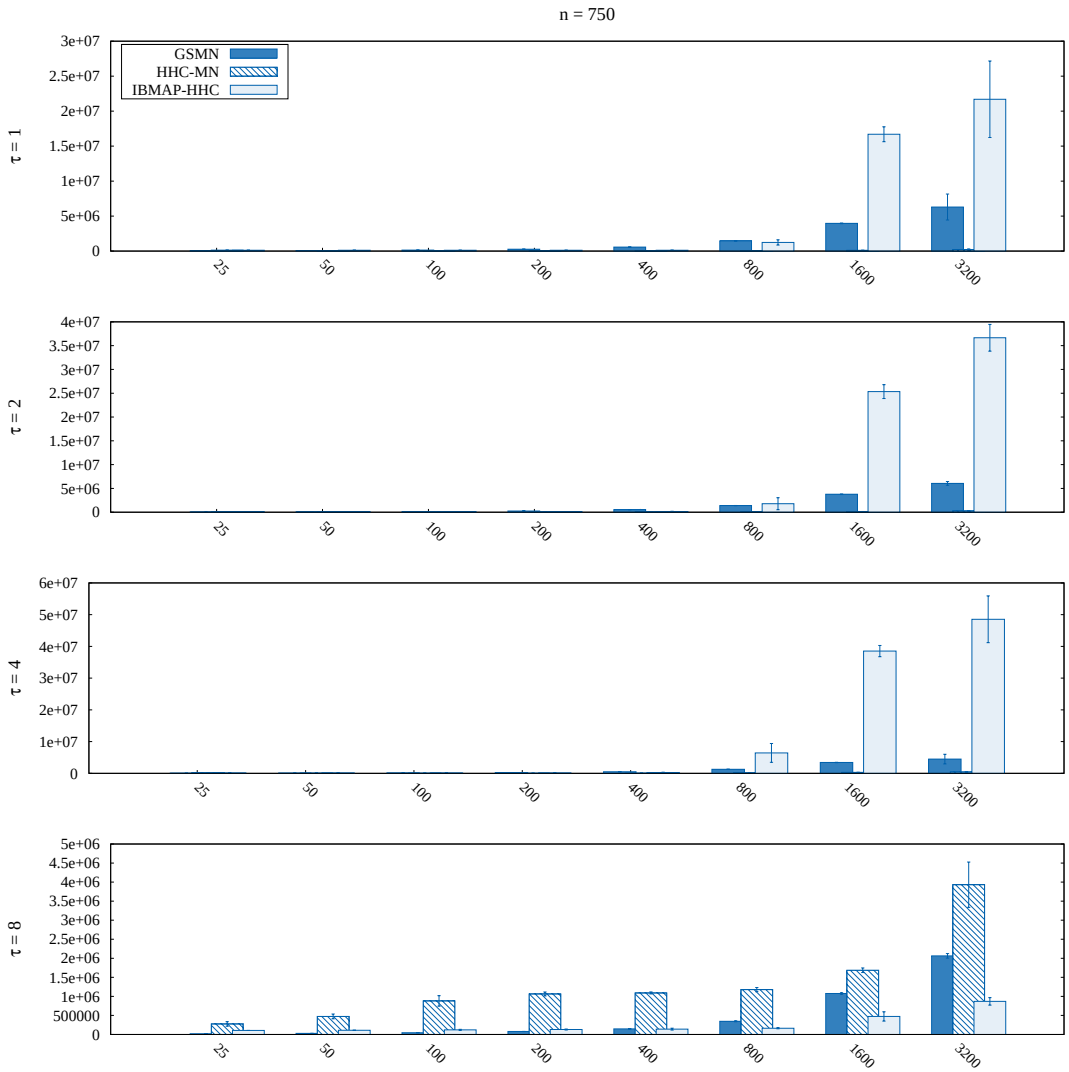

Figura 5.21: Tiempo de corrida para problemas con $n = 750$ variables para tamaños crecientes de $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, sobre estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (filas).

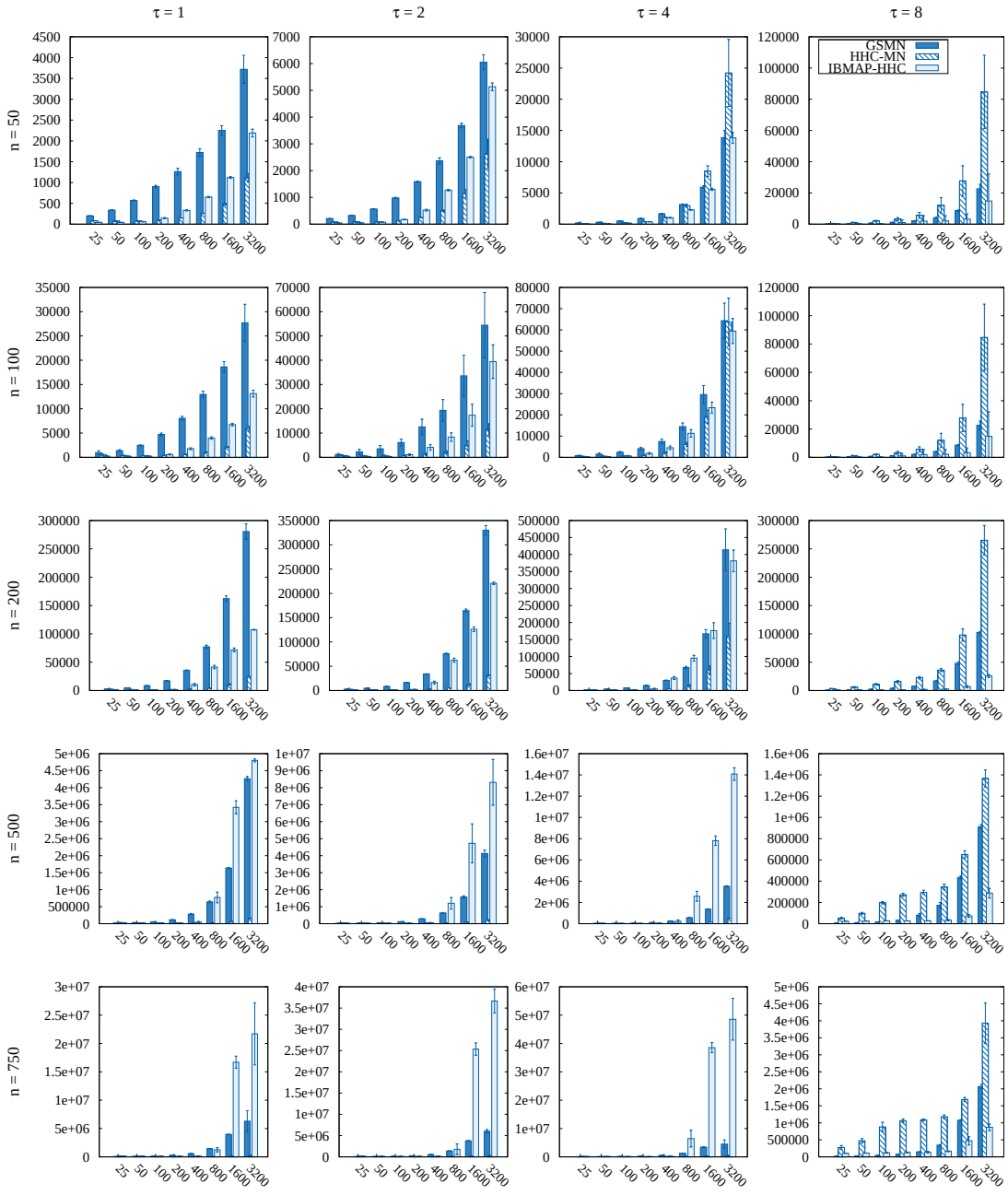

Figura 5.22: Tiempo de corrida para problemas con $n \in \{50, 100, 200, 500, 750\}$ variables (filas), con estructuras subyacentes de complejidad $\tau \in \{1, 2, 4, 8\}$ (columnas).

GSMN y HHC-MN del experimento descripto en la Sección 5.3 se muestran en las Figuras 5.17, 5.18, 5.19, 5.20 y 5.21, reportados en milisegundos. Para interpretar mejor dichos resultados, debe tenerse en cuenta que todos estos experimentos se corrieron en un AMD Athlon(tm), con 3.0 GHz y 4 GB de memoria RAM. En todas las figuras pueden verse similares patrones de crecimiento del tiempo de corrida respecto de D y n. Claramente, GSMN es el algoritmo más caro para la mayoría de los casos de conectividad baja, es decir, $\tau \in \{1, 2\}$. Esto se debe a que este algoritmo tiende a agregar una gran cantidad de falsos positivos en la etapa de crecimiento, y luego en la fase de encogimiento se requiere la ejecución de tests que contienen una gran cantidad de variables en el conjunto condicionante, lo que resulta en una fuente extra de costo computacional. Hay algunos casos donde IBMAP-HHC resulta la alternativa más cara, pero las diferencias de tiempo de corrida no son grandes. Para los casos de $\tau \in \{1, 2, 4\}$ y $D \geq 400$, el algoritmo que requiere el menor tiempo computacional es HHC-MN. Esto se debe a que el intercalado de su heurística de inclusión con su estrategia de eliminación (ver el Apéndice C) es realmente efectivo cuando la estructura subyacente tiene un valor de τ bajo, y la cantidad de datos disponibles D es lo suficientemente grande como para ejecutar tests estadísticos de alta confiabilidad. En estas situaciones, HHC-MN converge hacia su criterio de terminación muy rápidamente, y esa es la razón de que sea el que muestra mejores tiempos de corrida en estos casos. Sin embargo, en los casos de conectividad más alta ($\tau = 8$), el algoritmo HHC-MN incurre en un costo significativamente mayor a los demás algoritmos. Este hecho se debe al costo exponencial en que incurre su estrategia de eliminación, que ejecuta un test estadístico para cada subconjunto posible del conjunto condicionante actual (en promedio, de tamaño τ).

Para facilitar un análisis a nivel global, La Figura 5.22 comprime todos los tiempos de corrida de las figuras anteriores. Esta figura organiza los resultados disponiendo las gráficas para $n \in \{50, 100, 200, 500, 750\}$ en las filas de la grilla, y en las columnas los resultados para $\tau \in \{1, 2, 4, 8\}$. De este modo, puede verse verticalmente que hay tendencias a aumentar el tiempo de corrida a medida que crece n para una misma dificultad de estructuras (τ en la columna). Asimismo, también puede revisarse horizontalmente cómo para un mismo valor de n crece el tiempo de corrida de todos los algoritmos a medida que se incrementa la dificultad de las estructuras τ. Claramente, todos estos algoritmos tienen un costo que

crece con la complejidad del problema de aprendizaje (es decir, mientras crece n y τ). Puede verse también que para cualquier valor fijo de D, el costo computacional crece con τ (horizontalmente), o con n (verticalmente). Estos resultados, en conjunto con los resultados de la Sección 5.3, permiten mostrar empíricamente que IBMAP-HHC tiene un costo computacional altamente competitivo respecto de los algoritmos del estado del arte, arrojando resultados mucho mejores en términos de calidad estructural.

Adicionalmente, se corrió un experimento con el algoritmo IBMAP-HHC a fin de evaluar cómo crece el valor de M respecto de n, τ y D. Para estos experimentos, se utilizaron 10 redes de Markov sintéticas para distintos tamaños crecientes $n \in \{6, 12, 16, 20, 24, 30, 50, 75, 100, 200, 500, 750\}$ para niveles de conectividad creciente $\tau \in \{1, 2, 4, 8\}$. Se corrió IBMAP-HHC sobre todos estos conjuntos de datos, almacenando la cantidad de ascensos M requeridos por la búsqueda local para converger al máximo local. La Figura 5.23 muestra un conjunto de gráficas que reporta los valores promedio de M obtenidos sobre 10 conjuntos de datos de cada n en el eje X. Se muestra una gráfica por cada tamaño del conjunto de datos de entrenamiento utilizado $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$. Los resultados de este experimento muestran claramente que la cantidad de ascensos requeridos por IBMAP-HHC para converger al óptimo crece linealmente o sub-linealmente con n en todos los casos evaluados. Puede observarse que existen tendencias crecientes de M, requiriendo siempre mayores ascensos para las curvas de valores τ más grandes, y menores para las curvas de valores de τ más chicos. Específicamente, para los casos de menor densidad de aristas $\tau \in \{1, 2, 4\}$ el crecimiento de M respecto de n crece sub-linealmente, y en los casos de mayor densidad ($\tau = 8$) M crece linealmente para algunos casos. Puede verse también que a medida que crece D, para todos los valores de τ se converge con cantidades mayores de ascensos M. Este resultado es congruente con los resultados de distancia de Hamming de las secciones anteriores, donde la calidad mejora mientras aumenta el tamaño del conjunto de datos de entrenamiento D. Adicionalmente, estas tendencias muestran también que las curvas tienden a ser más lineales a medida que se incrementa el valor de D. Si se observan las gráficas de los valores más pequeños de D, la cantidad de ascensos M tiende a crecer más detenidamente con n, y para los valores más altos de D las tendencias en M crecen más linealmente. Estas tendencias tienen sentido, si se tiene en cuenta que utilizar

$D = 25$ puntos de datos no tiene el mismo impacto para un problema de $n = 6$ variables que para un problema de $n = 750$ variables. Por otro lado, utilizando $D = 3200$ puntos de datos se ve que la cantidad de ascensos crece más linealmente, pero nunca alcanzando un nivel de crecimiento exponencial. En resumen, dado que en ningún caso se ve un patrón de crecimiento exponencial de M, estos resultados demuestran que esta variable no es una fuente de complejidad extra para el algoritmo IBMAP-HHC.

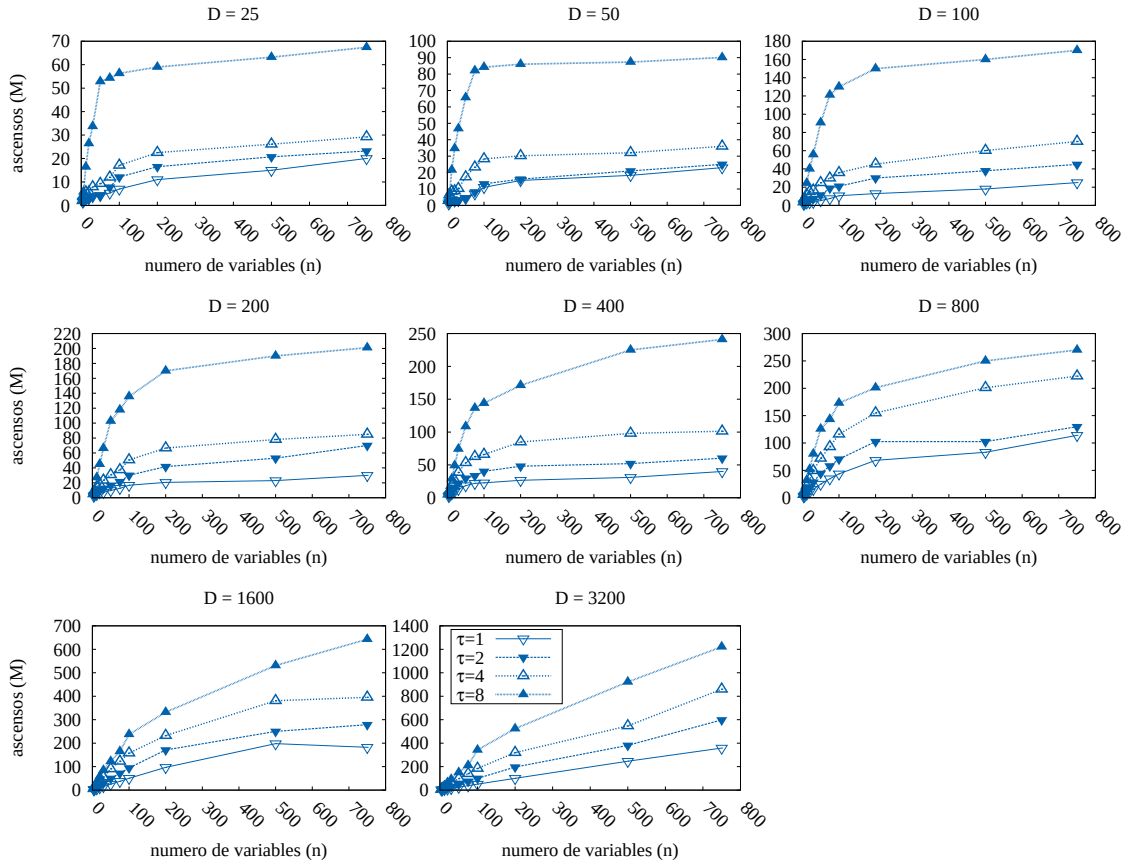

Figura 5.23: Cantidad de ascensos M de IBMAP-HHC (eje Y) para problemas con $n \in \{6, 12, 16, 20, 24, 30, 50, 75, 100, 200, 500, 750\}$ variables (eje X), utilizando conjuntos de datos de tamaños crecientes $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$.

A modo adicional, la Figura 5.23 muestra los resultados de un experimento similar para el algoritmo IBMAP-HC, que evalúa el IB-score para todos los vecinos

a distancia 1, en vez de utilizar la función heurística que utiliza IBMAP-HHC. A diferencia del experimento anterior, en este caso se reportan los resultados para un rango más pequeño de tamaños del dominio $n \in \{6, 12, 16, 20, 24, 30, 50\}$. Estos resultados demuestran que tras maximizar el IB-score sin utilizar la función heurística, el valor de M crece del mismo modo que utilizando la heurística, por lo que también se confirma empíricamente que M tampoco es una fuente de complejidad extra para IBMAP-HC.

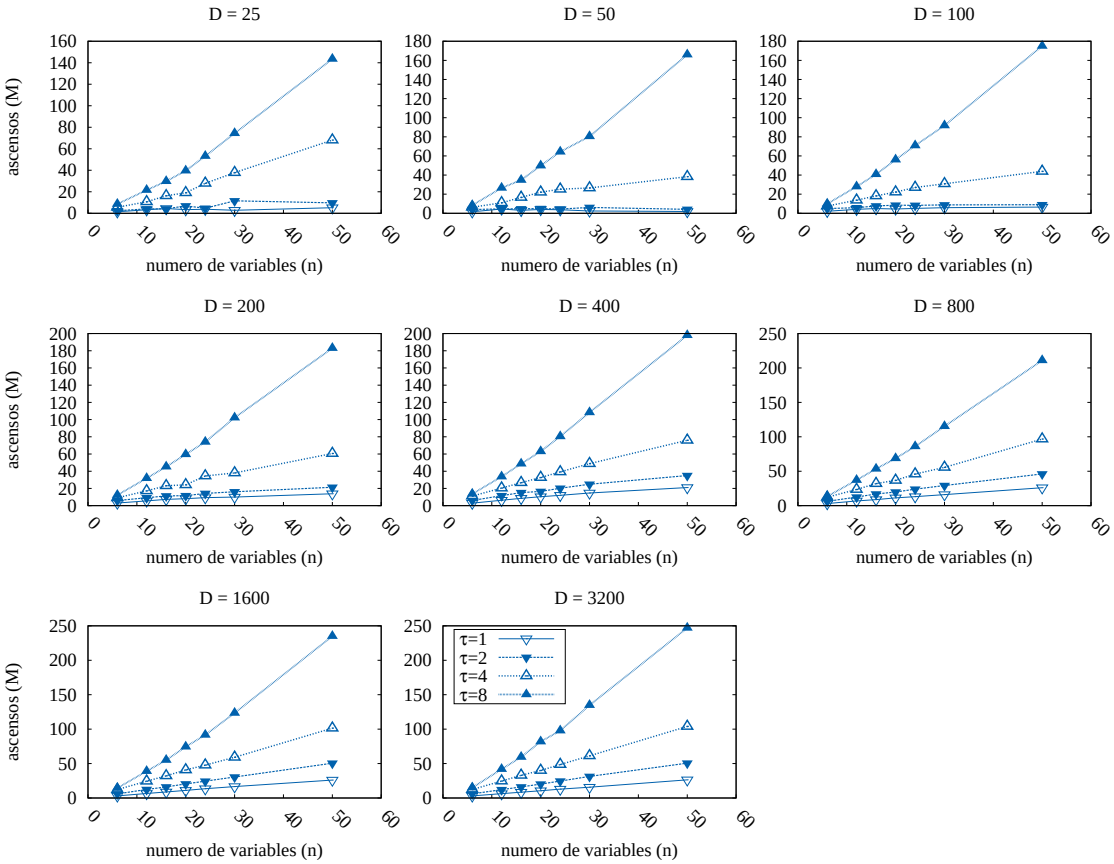

Figura 5.24: Cantidad de ascensos M de IBMAP-HC (eje Y) para problemas con $n \in \{6, 12, 16, 20, 24, 30\}$ variables (eje X), utilizando conjuntos de datos de tamaños crecientes $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$.

5.6. Análisis de superficie de la función IB-score

En esta sección se reporta una serie de experimentos realizados a fin de analizar empíricamente la superficie de la función de IB-score. Con este análisis se pretende evaluar la efectividad de los métodos de optimización propuestos, en términos de la maximización (más que de la calidad de la estructura aprendida). Este experimento consiste en mostrar el IB-score de las estructuras que conforman el espacio de búsqueda, a fin de analizar qué tan efectiva es la maximización realizada por los algoritmos IBMAP-HHC y IBMAP-GA sobre el espacio de búsqueda de estructuras. Estos dos algoritmos son las instancias más representativas del enfoque, una por su eficiencia y la otra por su potencial de exploración del espacio de estructuras.

En una primera instancia, se reportan los resultados para los conjuntos de datos de $n = 6$ reportados previamente en los experimentos de la Figura 5.1. Esto permite además mostrar la estructura encontrada por IBMAP-BF (fuerza bruta, la mejor maximización posible). Los resultados se muestran en la Figura 5.25, cuyas gráficas disponen las estructuras sobre el eje X, ordenadas según su distancia de Hamming hacia la estructura solución del conjunto de datos sintético utilizado para el experimento. Sobre el eje Y se muestra el IB-score de cada estructura. Dada esta disposición, para $n = 6$ los valores del eje X van desde 0 (la estructura solución) hasta $\binom{6}{2} = 15$, que es la máxima cantidad de errores estructurales para este tamaño de dominio. Note que los IB-score de las estructuras aparecen como logaritmos de probabilidades, ya que han sido computados como se muestra en la Ecuación (4.4). Con esta disposición, los puntos que están hacia la izquierda de la gráfica (cercanos a 0 en el eje X) representan las estructuras que tienen menos errores estructurales, y también son aquellas estructuras que se espera tengan mayores valores de la función IB-score. Asimismo, los puntos que están hacia la derecha representan las estructuras con más errores, que se espera tengan menores valores de la función IB-score. Adicionalmente, se muestra con un círculo grande la estructura hallada por IBMAP-BF (es decir, maximización por fuerza bruta), y con triángulos las estructuras que halladas por IBMAP-HHC y IBMAP-GA, a fin de analizar qué tan cercanas al óptimo son las estructuras encontradas por estos algoritmos. Los casos donde se ve una estrella dentro de un

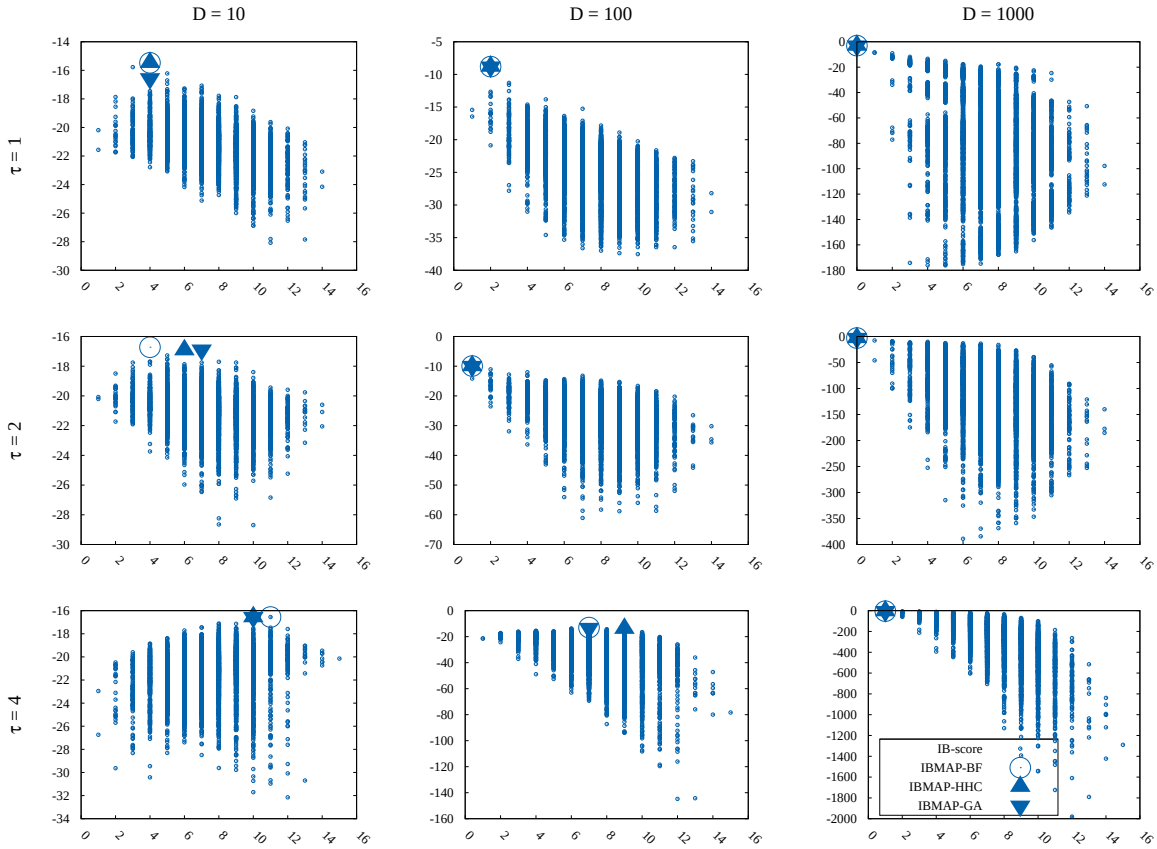

Figura 5.25: Superficie de IB-score para datos sintéticos de $n = 6$ variables, con tamaños $D \in \{10, 100, 1000\}$ en las columnas, y densidades $\tau \in \{1, 2, 4\}$ en las filas. El eje X ordena las estructuras por distancia de Hamming respecto de la estructura correcta. El eje Y muestra el IB-score de todas las estructuras del espacio de estados. La estructura encontrada por IBMAP-BF se indica con un círculo grande. Las estructuras encontradas por IBMAP-HHC y IBMAP-GA se indican con triángulos.

círculo son aquellos donde los tres algoritmos han encontrado la misma estructura. En la figura se muestra un conjunto de gráficas, ordenadas disponiendo los resultados para valores crecientes de $D \in \{10, 100, 1000\}$ sobre las columnas, y para los diferentes valores de $\tau \in \{1, 2, 4\}$ en las filas.

A partir del análisis de dichas gráficas, se observa cómo la superficie de la función IB-score se va tornando como una curva descendente a medida que se incremente el valor D, ya que se ve claramente una tendencia desde las gráficas de la izquierda hacia las de la derecha (note los cambios en la escala del eje Y). Este resultado es esperable, ya que la precisión de los tests estadísticos va mejorando a medida que se incrementa el tamaño del conjunto de entrenamiento D. Además, puede verse claramente que IBMAP-HHC y IBMAP-GA aprenden la misma estructura que IBMAP-BF en 5 de los 9 casos mostrados, y en los 4 casos restantes ambos algoritmos alcanzan estructuras con valores de IB-score tan altos como el de IBMAP-BF. Adicionalmente, puede observarse que el error de la estructura aprendida por IBMAP-HHC, IBMAP-GA y IBMAP-BF se va acercando a cero (hacia la izquierda), a medida que se incrementa D, lo que es congruente con los resultados de calidad mostrados en la Sección 5.1.

Este mismo experimento se corrió también para dominios de mayor tamaño $n \in \{20, 50\}$, ahora omitiendo el resultado de IBMAP-BF, ya que es imposible realizar búsqueda por fuerza bruta en estos espacios de estructuras. Por esta misma razón es que sólo se muestra un subconjunto de dicho espacio, muestreado uniformemente. Este muestreo se realizó generando 5 estructuras para cada posible cardinalidad de aristas del espacio de estructuras. En las Figuras 5.26 y 5.27 se muestran los resultados para $n = 20$ y $n = 50$, respectivamente. Por supuesto, en estos casos el espacio de estructuras crece exponencialmente con n: para $n = 20$, el eje X va desde 0 hasta $\binom{20}{2} = 190$, y para $n = 50$, el eje X va desde 0 hasta $\binom{50}{2} = 1225$. Los resultados que se ven en estas gráficas permiten obtener las mismas conclusiones que en el experimento de $n = 6$ variables, por lo que se omiten resultados para otros tamaños de dominio.

Para concluir esta sección, es oportuno notar que estos resultados confirman la efectividad de la estrategia de selección de estructuras de IBMAP-HHC y IBMAP-GA para maximizar la función IB-score. Dado que la función de maximización es altamente efectiva, no se considera promisorio extender este trabajo con el diseño de más algoritmos que instancien el enfoque IBMAP actual con nuevos

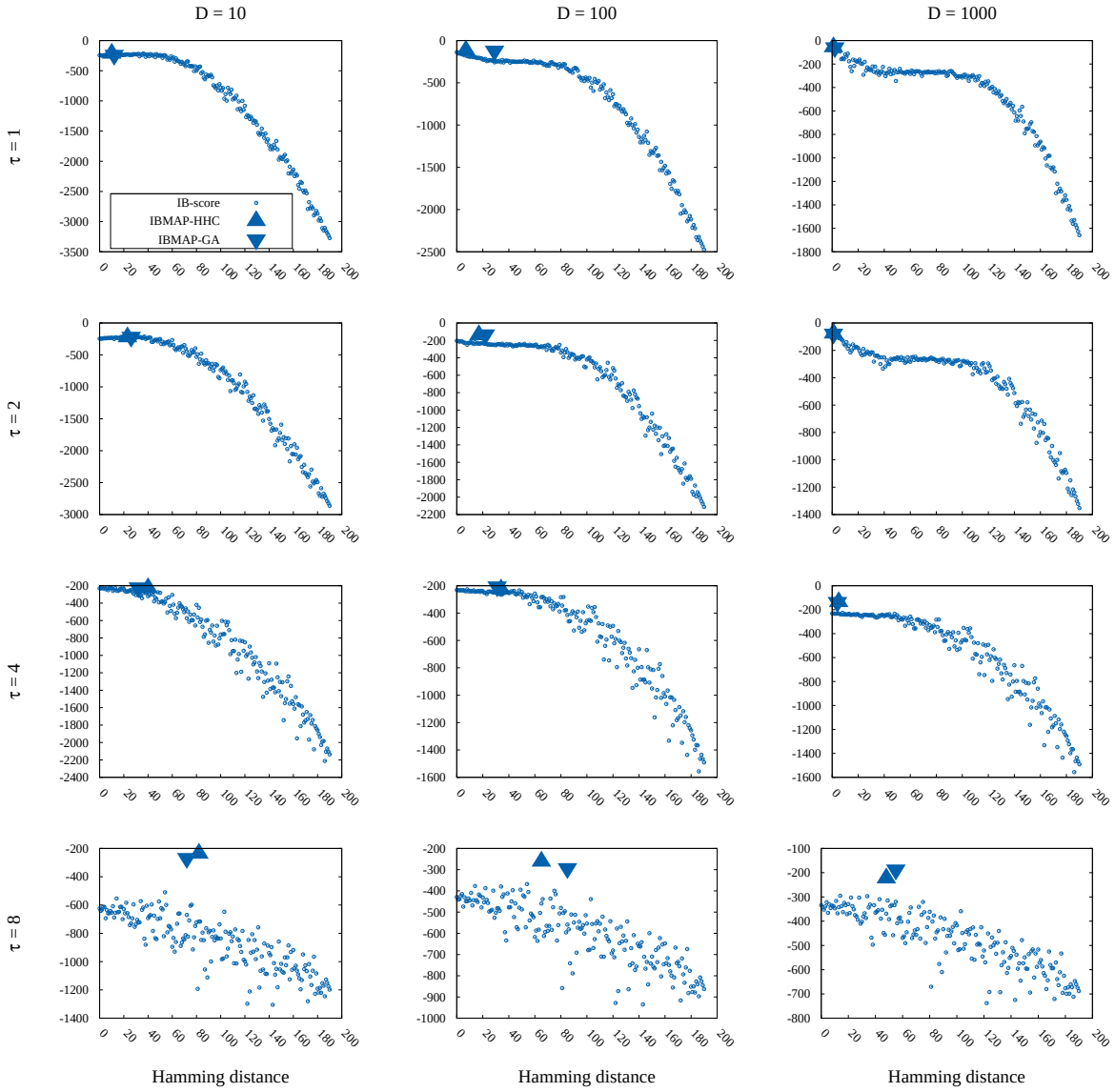

Figura 5.26: Superficie de IB-score para datos sintéticos de $n = 20$ variables, con tamaños $D \in \{10, 100, 1000\}$ en las columnas, y densidades $\tau \in \{1, 2, 4\}$ en las filas. El eje X ordena las estructuras por distancia de Hamming respecto de la estructura correcta. El eje Y muestra el IB-score de todas las estructuras del espacio de estados. Las estructuras encontradas por IBMAP-HHC y IBMAP-GA se indican con triángulos.

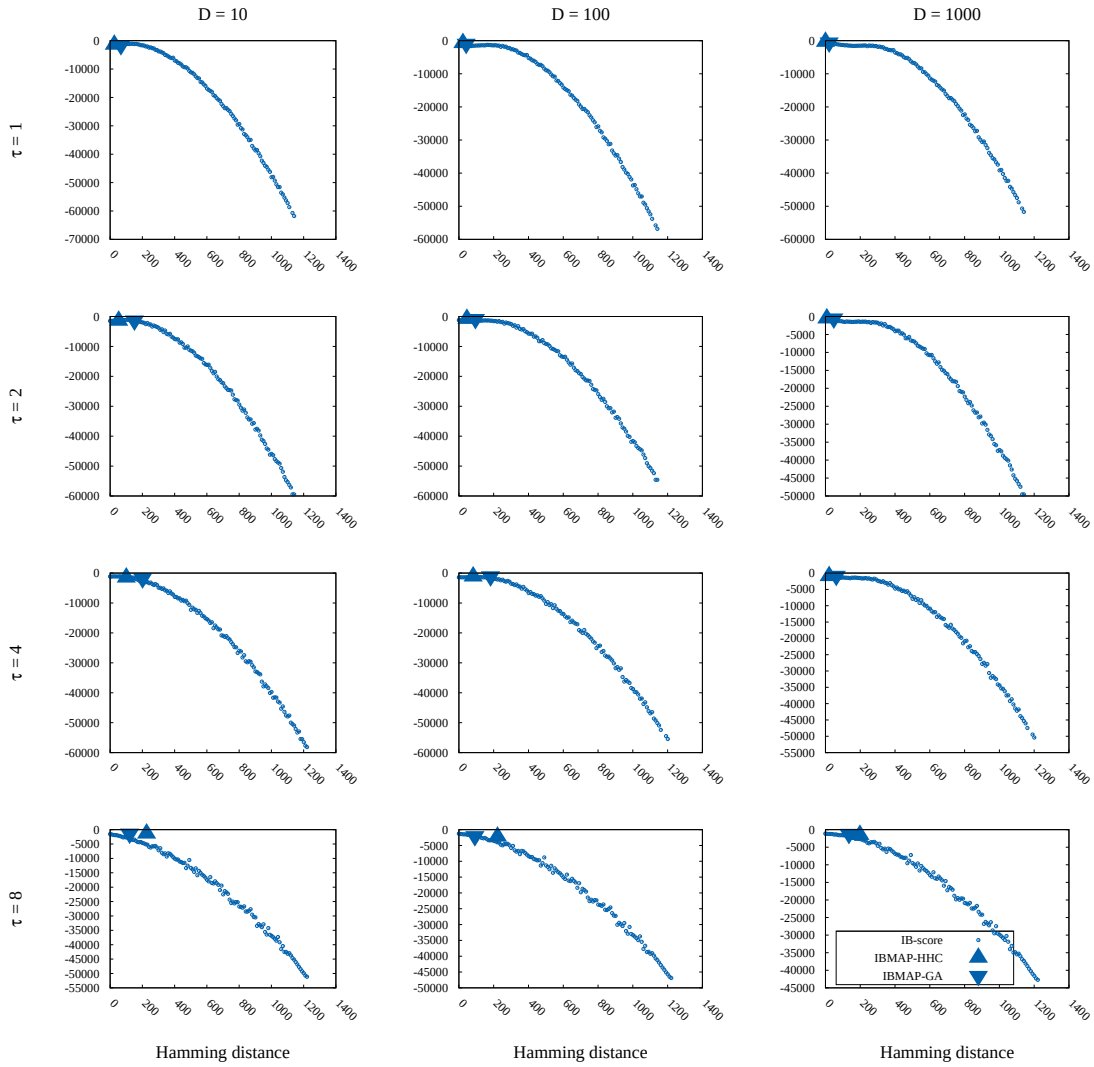

Figura 5.27: Superficie de IB-score para datos sintéticos de $n = 50$ variables, con tamaños $D \in \{10, 100, 1000\}$ en las columnas, y densidades $\tau \in \{1, 2, 4\}$ en las filas. El eje X ordena las estructuras por distancia de Hamming respecto de la estructura correcta. El eje Y muestra el IB-score de todas las estructuras del espacio de estados. Las estructuras encontradas por IBMAP-HHC y IBMAP-GA se indican con triángulos.

métodos de búsqueda. Por el contrario, como se discutirá en el Capítulo 6, las líneas de investigación futura más promisorias apuntan hacia el desarrollo de diversos mecanismos para computar $\Pr(G \mid D)$.

5.7. Aplicando IBMAP-HHC a EDAs

En esta sección, a modo accesorio, se incluyen una serie de resultados obtenidos tras evaluar IBMAP-HHC en una aplicación práctica y desafiante: los EDAs, conocidos en la literatura como *Estimation of Distribution algorithms* (Mühlenbein y Paaß, 1996; Larrañaga y Lozano, 2002). Se trata de una clase particular de algoritmos evolutivos que lleva a cabo los mismos pasos de selección y variación que los algoritmos genéticos, pero reemplazando las etapas de cruzamiento y de mutación por la estimación y el muestreo de una distribución de los mejores individuos, a fin de generar nuevas poblaciones. En un principio se estima una distribución de probabilidades a partir de los mejores individuos de la población actual, y la siguiente población es obtenida mediante el muestreo de esta distribución. En el paso de la estimación de la distribución, los EDAs toman como conjunto de datos de aprendizaje a la población actual. Para esto, cada gen está asociado a una variable aleatoria, cada individuo de la población es una asignación completa de dichas variables, y por lo tanto, se asume que la población seleccionada es un muestreo de la distribución subyacente a la distribución de las poblaciones. Por esto, utilizando la estimación de distribuciones a partir de los individuos seleccionados como los mejores permite generar poblaciones con soluciones novedosas, muestreando desde dicha distribución.

Recientemente, se han propuesto una serie de EDAs que utilizan redes de Markov para modelar la distribución de las poblaciones (Santana, 2005; Alden, 2007; Shakya y McCall, 2007; Shakya et al., 2012). Lo interesante de estos trabajos, es que la calidad del aprendizaje de estructuras de redes de Markov afecta fuertemente en los resultados de la optimización de los EDAs, por lo que en esta sección se muestra que utilizando IBMAP-HHC dentro de un EDA, se puede mejorar considerablemente la convergencia. Se trata de un caso de prueba complejo, ya que en los EDAs se realiza aprendizaje de estructuras en cada una de las generaciones de una optimización iterativa, y las poblaciones se generan mediante el muestro de la distribución aprendida. Por esto, la hipótesis a probar en esta

sección es que mientras más correcta es la estructura aprendida, más efectivo resulta el muestreo de generaciones, y más efectiva es la optimización.

Para este experimento, se consideró el algoritmo MOA (*Markovianity Optimization Algorithm*) (Shakya et al., 2012). Se trata de un EDA basado en redes de Markov que actualmente es estado del arte. MOA funciona utilizando un algoritmo de aprendizaje de estructuras muy trivial, basado en el cómputo de información mutua (IM). Este algoritmo no es parte del estado del arte de aprendizaje de estructuras, ya que se trata de un módulo de aprendizaje de estructuras de redes de Markov diseñado específicamente para el algoritmo MOA. El muestro en MOA se lleva a cabo mediante una variación del muestreador de Gibbs que requiere sólo la estructura del modelo, evitando la necesidad de aprender los parámetros numéricos. La implementación de IM en MOA toma ventaja del conocimiento de expertos, requiriendo como parámetro el máximo número de variables vecinas que puede tener una variable, es decir, el mismo concepto llamado τ en los experimentos anteriores de este capítulo. Una desventaja de utilizar este parámetro es que la calidad del algoritmo IM resulta muy sensible a los cambios del valor utilizado de τ. Por el contrario, el algoritmo IBMAP-HHC no requiere el uso de dicho parámetro. En los experimentos que mostraremos a continuación, se utiliza para IM el valor óptimo de τ, resultando en su mejor rendimiento posible.

Los experimentos llevados a cabo comparan IBMAP-HHC como un algoritmo de aprendizaje de estructuras alternativo dentro de MOA, denotando MOA' a esta versión y MOA a la versión original que utiliza IM. La tesis es que un mejor algoritmo de aprendizaje mejora la convergencia de MOA, es decir, se obtiene el óptimo computando mucho menos evaluaciones de la función de fitness. Ambas versiones fueron testeadas en dos funciones de referencia ampliamente utilizadas en la literatura de EDA's: *Royal Road* y *OneMax*. Las dos funciones son tareas de optimización de cadenas de bits, cuyo detalle puede encontrarse en Mitchell (1998). La razón de utilizar dichas funciones es que éstas son ampliamente utilizadas en el área de algoritmos evolutivos, ya que son difíciles de optimizar, porque la forma del espacio de búsqueda es muy plana durante largas áreas, y luego son discontinuas en los óptimos. En el contexto de los algoritmos evolutivos, dichas funciones modelan cada cadena de bits como un cromosoma, y cada bit como un gen. En el problema Royal Road, las variables son agrupadas en grupos de γ. El objetivo es maximizar el número de 1s en la cadena, pero la función de fitness

sólo suma γ cuando todo el grupo de variables posee el valor 1s, sino suma 0. Por ejemplo, en el caso de que $\gamma = 4$, un individuo `111110011111` se separa en los grupos `[1111] [1001] [1111]`, y sólo el primer y el tercer grupo suman 4 al fitness, que para el ejemplo es 8. La estructura de independencias subyacente a esta distribución contiene cliques de tamaño γ, uno por grupo. En nuestros experimentos utilizamos $\gamma = 1$ y $\gamma = 4$. El primer caso es conocido en la literatura como el problema *OneMax*. Para el individuo de ejemplo, el fitness de OneMax es 10. Claramente, el individuo óptimo que se desea obtener en ambos problemas es `111111111111`, con fitness igual a 12.

En nuestros experimentos, MOA se itera para 1000 generaciones o hasta que el óptimo de la función objetivo sea obtenido, lo que ocurra primero. Para varias corridas que difieren en la población inicial aleatoria, se mide el porcentaje de éxito como la fracción de veces que el óptimo ha sido hallado. Una medida de rendimiento comúnmente utilizada en EDAs es *el tamaño crítico de población* D^*; que es el tamaño mínimo de población para el cual el porcentaje de éxito de la optimización es el 100 %. Menores valores de D^* tienen un doble beneficio en el costo computacional de la optimización: (i) menos evaluaciones de la función de fitness para obtener el óptimo, y (ii) la estimación de distribuciones es más rápida. Por esto, se reporta D^*, y el número de evaluaciones del fitness requerido para este tamaño de población, denotado f^*. Mientras menores valores de D^* y f^* se requieran, más efectivo y robusto es el algoritmo. Para medir D^* en Royal Road y OneMax, corrimos MOA y MOA' 10 veces para un número creciente de poblaciones $D \in \{50, 100, 200, 400, 800, 1600, 3200\}$. Luego, para el valor D^* medido, se reporta el promedio y la desviación estándar de f^* en cada una de las corridas. En todos los experimentos, se selecciona la población en un 50 % y un elitismo del 50 %; valores comúnmente utilizados para evitar pérdida de diversidad en las poblaciones. En MOA, el parámetro τ se puso en 3 y 1 para los problemas Royal Road y OneMax, respectivamente.

En la Tabla 5.2 se presentan los resultados para el problema OneMax, y en la Tabla 5.3 para el problema Royal Road. Para ambos algoritmos MOA y MOA', cada tabla reporta los valores de D^* y también el promedio y desviación estándar de f^*, para tamaños crecientes del problema $n \in \{15, 30, 60, 90, 120\}$ para OneMax, y $n \in \{16, 32, 64, 92, 120\}$ para Royal Road (los tamaños en este caso deben ser múltiplos de $\gamma = 4$). Menores valores de D^* y f^* son mejores.

OneMax				
	MOA		MOA'	
n	D^*	f^*	D^*	f^*
15	50	267.50 (35.45)	50	202.50 (14.19)
30	200	1170.00 (94.87)	100	475.00 (42.49)
60	800	5200.00 (98.46)	200	1050.00 (52.70)
90	800	5560.00 (126.49)	400	2220.00 (63.25)
120	1600	11200.00 (871.53)	800	4400.00 (312.33)

Tabla 5.2: Resultados para MOA y MOA' (que usa IBMAP-HHC) para OneMax, para problemas de tamaño creciente (filas) en términos de tamaño crítico de población D^*, y el promedio y desviación estándar sobre 10 repeticiones del número de evaluaciones de la función de fitness f^* requerido para obtener el óptimo global. Menores valores de D^* y f^* son mejores.

Royal Road				
	MOA		MOA'	
n	D^*	f^*	D^*	f^*
16	100	545.00 (59.86)	50	337.50 (176.09)
32	400	3800.00 (210.82)	400	2140.00 (134.99)
64	800	9120.00 (252.98)	800	4440.00 (126.49)
92	1600	18400.00 (533.33)	800	5080.00 (500.67)
120	1600	31120.00 (822.31)	1600	9840.00 (386.44)

Tabla 5.3: Resultados para MOA y MOA' (que usa IBMAP-HHC) para Royal Road, para problemas de tamaño creciente (filas) en términos de tamaño crítico de población D^*, y el promedio y desviación estándar sobre 10 repeticiones del número de evaluaciones de la función de fitness f^* requerido para obtener el óptimo global. Menores valores de D^* y f^* son mejores.

En ambas tablas, los resultados muestran que MOA' siempre arroja valores menores o iguales de D^* que MOA, y también se puede ver que MOA' siempre mejora los valores de f^* de MOA. Para el problema Royal Road, la mejora más contundente se ve para el caso de $n = 92$ donde MOA' requiere un 75 % menos de evaluaciones del fitness f^* y D^* es justo la mitad. Para el problema OneMax, la mejora más contundente se observa para el caso de $n = 60$, donde MOA' requiere un 80 % menos de evaluaciones del fitness f^* y D^* es reducido a un cuarto. Una interpretación de dichos resultados es que IBMAP-HHC estima mejor la distribución en cada generación. En un experimento sobre los datos sintéticos de los experimentos anteriores, se ha confirmado que las estructuras aprendidas por MI tienen distancias de Hamming mucho peores que las de IBMAP-HHC en todos los casos. No tiene sentido reproducir dichos resultados aquí, ya que esto puede verse claramente en los resultados de MOA y MOA'.

5.8. Resumen

En este capítulo se presenta una validación empírica del enfoque IBMAP. Dicha experimentación muestra que los algoritmos que instancian este enfoque son más robustos en términos de calidad que los algoritmos del estado del arte. También se muestra que el algoritmo IBMAP-HHC es la instancia más competitiva en términos de calidad y eficiencia. La evaluación experimental se realizó mayormente sobre datos sintéticos, lo que permitió realizar un análisis sobre el desempeño de los diversos algoritmos de aprendizaje en distintas situaciones de complejidad y de disponibilidad de datos.

Primeramente, se muestra en la Sección 5.1 que IBMAP-BF obtiene las mejores calidades estructurales entre todos los algoritmos en instancian el enfoque. Sin embargo, esta técnica no es escalable a dominios de más de 6 variables binarias. Adicionalmente, dichos resultados también permitieron demostrar que las demás instanciaciones de IBMAP presentan calidades comparables a la búsqueda por fuerza bruta. Posteriormente, en la Sección 5.2 los resultados con el algoritmo IBMAP-GA demuestran que dicho enfoque realiza una optimización altamente efectiva, y que en muchos casos se mejoran las calidades obtenidas mediante búsquedas locales. Sin embargo, el costo de dicha técnica no es viable para dominios de gran dimensionalidad, sumado al hecho de que se debe correr el algoritmo

sobre una gran variedad de parámetros diferentes. No obstante, tanto esta técnica como las utilizadas por IBMAP-HC, IBMAP-HC-RR e IBMAP-HHC-RR pueden resultar útiles en situaciones donde se desea invertir tiempo de cómputo a fin de obtener modelos de alta calidad. En el resto de la experimentación se muestra que IBMAP-HHC es la instancia más eficaz, mejorando contundentemente la calidad de los modelos aprendidos a medida que crece la dimensionalidad del problema, a un costo competitivo respecto de los algoritmos del estado del arte. En la Sección 5.3 se muestran experimentos para dominios de hasta $n = 750$ variables binarias, donde la calidad estructural es mejorada contundentemente respecto de los algoritmos competidores seleccionados. En la Sección 5.5 se analizan los tiempos de corrida y la eficiencia de IBMAP-HHC, demostrando la competitividad de dicho algoritmo en términos de complejidad computacional. En la Sección 5.6 se demuestra la eficacia de IBMAP-HHC y IBMAP-GA para maximizar la función IB-score, concluyendo en que las líneas de investigación futura más promisorias apuntan hacia el desarrollo de diversos mecanismos para computar $\Pr(G \mid D)$, en vez de continuar con el diseño de mecanismos alternativos para optimizar el IB-score. Por último, en la Sección 5.4 se reportan resultados de IBMAP-HHC altamente competitivos sobre datos reales, y en la Sección 5.7 se muestra que su aplicación en los algoritmos EDAs mejora contundentemente su convergencia.

Capítulo 6

Resumen y conclusiones

En este trabajo se presenta el enfoque IBMAP para el aprendizaje de estructuras de redes de Markov. Este enfoque propone la maximización de una función de puntaje sobre el espacio de las posibles estructuras de independencia. Dicha función de puntaje es llamada IB-score, una aproximación de la probabilidad a posteriori de una estructura dados los datos $Pr(G|D)$. El IB-score se computa haciendo una conjunción de las estadísticas calculadas por un conjunto de tests que determinan las independencias de cada estructura. Este enfoque fue diseñado con el fin de mejorar la calidad de los algoritmos basados en independencia, evitando el efecto cascada producido por confiar ciegamente en los resultados de los tests estadísticos. Adicionalmente, se propone el uso del conjunto de cierre basado en mantas de Markov, como un mecanismo lógico que permite el cómputo del IB-score eficientemente, utilizando un número de tests estadísticos cuadrático en la cantidad de variables del dominio, y que permite computar IB-score incrementalmente desde estructuras vecinas.

A modo de instanciación del enfoque se presentan diversos algoritmos, que realizan la maximización con distintos métodos de optimización. La mejor instanciación de este enfoque en términos de calidad estructural y costo computacional es el algoritmo IBMAP-HHC, que maximiza el IB-score con una búsqueda heurística de ascensión de colinas. Asimismo, IBMAP-HHC-RR permite reiniciar dicha búsqueda un número arbitrario de veces, partiendo desde estructuras iniciales aleatorias. Otros de los métodos presentados, como IBMAP-HC, IBMAP-HC-RR, e IBMAP-GA, permiten flexibilizar la búsqueda de modo que sea posible

invertir tiempo de cómputo en realizar una exploración más profunda en el espacio de estructuras. Esto permite utilizar el enfoque en casos donde se requiere encontrar un modelo de alta calidad (e.g., descubrimiento de conocimiento).

El enfoque se valida mediante una evaluación experimental realizada mayormente sobre datos sintéticos aleatorios, de complejidad controlada. Primeramente se muestra que los algoritmos que instancian IBMAP mejoran las calidades estructurales de los competidores del estado del arte basados en independencia. Se reportan resultados para el algoritmo IBMAP-BF (que maximiza el IB-score con fuerza bruta), obteniendo las mejores calidades estructurales, y también se reportan resultados para las demás instanciaciones, demostrando que las calidades obtenidas por métodos más eficientes son comparables. Se muestra además una experimentación completa sobre la parametrización del algoritmo IBMAP-GA, validando que dicho enfoque mejora en muchos casos a los métodos de búsqueda local. Luego, se muestran experimentos sobre dominios de mayor tamaño, mostrando que IBMAP-HHC permite mejorar la calidad en problemas de gran dimensionalidad y complejidad, a un costo computacional competitivo respecto del costo de los competidores. Además, se reportan los resultados de un experimento que demuestra la capacidad de maximización de los algoritmos IBMAP-HHC e IBMAP-GA. De este modo se demuestra empíricamente que los métodos propuestos para realizar la maximización del IB-score son altamente efectivos en la maximización. Adicionalmente, se reportan resultados que demuestran la eficacia de utilizar IBMAP-HHC en datos del mundo real, y en una aplicación de aprendizaje de estructuras para algoritmos evolutivos.

6.1. Investigación complementaria realizada

Durante el desarrollo de esta tesis se ha llevado a cabo una serie de trabajos de investigación complementaria, realizada con otros colegas. Todas estas investigaciones están relacionadas al área de aprendizaje de redes de Markov. A continuación se describen brevemente dichos trabajos y los resultados obtenidos.

6.1.1. Algoritmo GSS

En una primera instancia de esta investigación se desarrolló el algoritmo GSS, un algoritmo basado en adaptar el algoritmo GS (Margaritis y Thrun, 2000) para mejorar el aprendizaje de la manta de Markov de cada variable. Mientras que GS confía totalmente en los resultados de los tests estadísticos, GSS decide utilizando una primer versión del IB-score que en vez de computar la posterior de estructuras, computa la posterior de la manta de Markov de una variable. GSS funciona a través de un proceso de optimización sobre todos los ordenamientos posibles de las variables, y todas las posibles respuestas de independencia de los tests ejecutados. Luego, se busca la configuración que maximiza dicha medida de calidad, según el ordenamiento.

En este trabajo se realizó una comparación experimental basada en datos sintéticos, donde se obtuvieron mejoras de hasta un 10 % en la calidad estructural, respecto de GS. Esta investigación dio el puntapié inicial hacia el diseño del enfoque IBMAP presentado en este trabajo. Este trabajo fue publicado en Bromberg y Schlüter (2009).

6.1.2. Acelerando la ejecución masiva de tests

Otra investigación relacionada se realizó luego de desarrollar los algoritmos IBMAP-BF, IBMAP-HC e IBMAP-HC-RR, a fin de disminuir su complejidad temporal. Estos algoritmos tenían un alto costo computacional, pero sus ventajas en términos de calidad eran claras. Por esto, con el fin de obtener instanciaciones más prácticas, se realizó una investigación basada en una idea sobre cómo acelerar la ejecución masiva de un gran número de tests estadísticos. Este trabajo fue publicado en Schlüter et al. (2009), presentando una estructura de datos para construir tablas de contingencia para un test de independencia, reutilizando tablas de contingencia de otros tests ejecutados previamente. Esto tiene un fuerte impacto cuando se ejecutan muchos tests de independencia, ya que la construcción de las tablas de contingencia es el paso más costoso dentro de la ejecución de un test de independencia.

Si bien este método demostró tener un alto potencial para acelerar la ejecución de los algoritmos basados en el enfoque IBMAP, la necesidad de explorar el espacio de estructuras mucho más eficientemente desvió el curso de la investigación hacia

la búsqueda de heurísticas para elegir estructuras locales vecinas mucho más eficientemente. Además, luego del desarrollo de la idea se investigó una estructura de datos de gran envergadura y desarrollada con el mismo propósito, llamada AD-tree (Moore y Lee, 1998). En la Sección 6.2 se propone una línea de trabajo a futuro para extender el estado actual de esta investigación, que está basada en el uso de esta estructura de datos.

6.1.3. Mejorando estrategias para algoritmos LGL

Otra investigación complementaria realizada durante el desarrollo de la presente tesis consistió en buscar estrategias alternativas para mejorar el paso de construcción global de la estructura en los algoritmos que utilizan la estrategia LGL (ver el Algoritmo 1). La idea de mejora consistió en atacar el hecho de que estos algoritmos generan inconsistencias cuando una de las variables pertenece a la manta de Markov de otra, pero esta otra no pertenece a su manta de Markov. Esto es un claro indicio de error en alguno de los dos aprendizajes de la manta de Markov. Por esto, como un competidor obvio se propuso evaluar la eficacia de utilizar como estrategia alternativa una regla "AND", en contraste con la regla "OR" utilizada tradicionalmente. Es decir, la regla "AND" propone agregar una arista entre dos variables en la estructura solución cuando ambas se corresponden a sus respectivas mantas de Markov. Adicionalmente, en este trabajo se propone una estrategia llamada la regla "EP" (edges probability), que se basa en computar la probabilidad de existencia o ausencia de una arista. Dichas probabilidades son computadas a través de tests de independencia.

En este trabajo se reportan los resultados de una experimentación en datos sintéticos, mostrando que este tipo de errores son causantes de una gran parte de los errores cometidos por los algoritmos que utilizan la estrategia LGL, y mostrando también que utilizar las reglas "AND" y "EP" presentan mejoras interesantes en la calidad estructural, resultando en algunos casos en reducciones de hasta el 50 % de aristas aprendidas incorrectamente. Este trabajo fue publicado en Schlüter et al. (2011).

6.1.4. Distribuciones con independencias específicas del contexto

Una línea de investigación complementaria en la que se trabajó paralelamente y se ha avanzado con resultados interesantes consiste en utilizar algoritmos basados en independencia para aprender distribuciones que no se pueden representar con un simple grafo no dirigido. Estas distribuciones contienen independencias específicas del contexto (Boutilier et al., 1996), que son independencias condicionales que en vez de cumplirse sobre todas las asignaciones del conjunto condicionante, sólo se cumplen sobre un contexto específico del conjunto condicionante. Lo interesante de esta investigación es que cuando la distribución presenta dichos patrones de independencia, los algoritmos que aprenden un grafo no dirigido tienden a aprender estructuras muy densas que terminan obscureciendo las independencias de la distribución, y que además resultan ser mucho menos compactas. Por esto se trabajó en el desarrollo de un enfoque que propone una representación alternativa de la estructura llamada *modelos canónicos*, y un algoritmo basado en independencias que aprende este tipo de modelos. Se realizó una evaluación empírica extensiva, demostrando que este método puede aprender estructuras de muy buen poder predictivo, comparando con algoritmos basados en independencia, y también con algoritmos basados en puntaje. Los resultados de este trabajo fueron publicados en Edera et al. (2013), obteniendo un premio a mejor paper de estudiantes.

6.2. Trabajo futuro

Durante el desarrollo del presente trabajo se han reconocido varias líneas de investigación que podrían realizarse a futuro. Las más importantes se revisan en las siguientes sub-secciones.

6.2.1. Relajación de IB-score

Para comenzar, la línea más importante por donde extender esta investigación es relajar la suposición de independencia entre todas las aserciones del cierre que se lleva a cabo explícitamente en la Ecuación (4.4). Dados los resultados ex-

perimentales presentados en este trabajo, la conclusión respecto de esto es que efectivamente dichas aserciones son independientes entre sí cuando el conjunto de datos es suficientemente grande. En contraste, cuando existe escasez de datos, las aserciones de independencia del conjunto de cierre no son independientes entre sí, ya que sólo los datos no son suficientes para elicitar dichas independencias. Claramente, algunas relaciones lógicas y axiomáticas entre dichas aserciones de independencia pueden contribuir a computar el IB-score más eficazmente, relajando dicha suposición.

Siguiendo estas ideas, se intentó diseñar métodos para computar las probabilidades mostradas en la Ecuación (4.3), es decir, la probabilidad de aserciones de independencia condicionadas en otras aserciones de independencia y los datos $\Pr(c_i|c_1,\ldots,c_{i-1},D)$. Para esto, se estudió el test Bayesiano en detalle (ver el Apéndice A), y se diseñó un test Bayesiano compuesto que recibe dos preguntas de independencia que se sabe a priori que están co-relacionadas. Luego se realizaron pruebas para determinar cómo el valor de uno de los tests sirve para ajustar los parámetros de información a priori del otro test que está co-relacionado. En estas pruebas se detectó que de este modo se puede mejorar la precisión de los tests de independencia obteniendo estadísticas más correctas cuando se modifica la probabilidad a priori de un test según los resultados de un test co-relacionado.

Dados estos resultados positivos, se corrieron experimentos utilizando una versión del IB-score relajado, donde se relacionan las aserciones del conjunto de cierre que implican a las mismas variables, es decir, las aserciones $(X \perp\!\!\!\perp Y|\mathbf{MB}^X)$ y $(Y \perp\!\!\!\perp X|\mathbf{MB}^Y)$. Esto se debe a que dentro del conjunto de cierre basado en mantas de Markov existen este tipo de correlaciones, ya que cuando una aserción $(X \perp\!\!\!\perp Y|\mathbf{MB}^X)$ arroja altas probabilidades de independencia, la otra aserción $(Y \perp\!\!\!\perp X|\mathbf{MB}^Y)$ también debiera arrojar altas probabilidades de independencia (igualmente en casos de arista, con relaciones de dependencia). Se corrieron una serie de experimentos similares a los mostrados en el Capítulo 5 para evaluar la calidad de las estructuras obtenidas tras maximizar por fuerza bruta el IB-score utilizando esta relajación. Como resultado, se obtuvieron mejoras despreciables en la calidad estructural. Es decir, esta relajación del IB-score no produjo mejoras significativas.

Para corroborar la naturaleza de los resultados obtenidos, también se corrieron experimentos similares a los mostrados en la Sección 5.6 para visualizar la

forma del espacio de estructuras de IB-score, y realizar una comparación entre la versión de IB-score que asume independientes a todas las aserciones del cierre, y esta versión nueva que relaja dicha suposición. En estos experimentos se pudo observar que la forma del IB-score en el espacio de estructuras son muy similares en ambos casos, lo que resulta congruente con el hecho de que esta relajación de la aproximación en el IB-score no tenga mayor impacto en la calidad de las estructuras aprendidas.

No obstante, esta línea de investigación en pos de mejorar la función IB-score parece ser la vía más promisoria para mejorar el alcance de IBMAP. Actualmente, las líneas por donde atacar este problema apuntan a desarrollar analíticamente las fórmulas desarrolladas en el Apéndice A, a fin de computar la probabilidad a posteriori del modelo de independencia y del modelo de dependencia, condicionando en los datos y en otras aserciones de independencia computadas previamente. Para esto habría que trabajar en encontrar una forma analítica de computar la Ecuación (A.1), la Ecuación (A.2) y la Ecuación (A.4) pero condicionando en otras aserciones de dependencia. Actualmente, este trabajo analítico se encuentra en desarrollo.

Adicionalmente, sería propicio diseñar un análisis de convexidad y rugosidad de las diferentes instancias del IB-score diseñadas, a fin de corroborar más específicamente cómo es que esta medida de calidad crece o decrece a medida que se agregan aristas correctas o incorrectas a una estructura dada. Además, sería propicio contemplar diversas formas del vecindario entre estructuras. En el presente trabajo se utiliza una concepción natural de la vecindad entre grafos no dirigidos, que consta de relacionar todas las estructuras que difieren en una arista. Otras formas de considerar vecindad entre estructuras podría considerar a las estructuras que difieren en k aristas. Así podría estudiarse cómo varían las convexidades y rugosidades del IB-score respecto de k, a fin de encontrar una forma del IB-score que resulte más bondadosa en términos de su maximización.

6.2.2. Diseño de conjuntos de cierre alternativos

Otro punto de diseño importante en el enfoque IBMAP es el conjunto de cierre de las estructuras. En un principio, el enfoque se diseñó de manera que se pueda instanciar con diversos cierres, he ahí la presentación de la Definición 4 en

el Capítulo 4, donde se presenta el enfoque en términos de un cierre abstracto. Luego, en la Sección 4.2 se presenta el conjunto de cierre basado en mantas de Markov, como una manera de instanciar el IB-score. Este cierre se realizó con la intención de explotar las independencias locales a cada una de las variables, aprovechando que una estructura de n variables puede descomponerse en n mantas de Markov. El resultado fue un conjunto de cierre que requiere de $n \times (n-1)$ aserciones de independencia para determinar la estructura, y que además permite el cómputo incremental del IB-score entre estructuras vecinas. Este fue uno de los factores más importantes para que la complejidad temporal de IBMAP-HHC sea competitiva respecto de los algoritmos del estado del arte. Claramente, una vía de exploración para extender el enfoque IBMAP consta de diseñar otros cierres diferentes al basado en mantas de Markov. Para esto, sería interesante estudiar cómo es que cada configuración posible del cierre determina la forma de la función IB-score sobre el espacio de estructuras (e.g., estudiar cambios de convexidad y rugosidad). De hecho, esta forma del IB-score también se vería afectada por las diferentes maneras de asumir las vecindades en el espacio de estructuras.

Hasta la actualidad, se diseñaron algunas ideas para mejorar el conjunto de cierre basado en mantas de Markov, como por ejemplo, reducir la cantidad de aserciones de independencia del cierre utilizando únicamente una aserción por cada par de variables (X, Y). Una idea fue elegir entre las aserciones del tipo $(X \perp\!\!\!\perp Y | \mathbf{MB}^X)$ y $(Y \perp\!\!\!\perp X | \mathbf{MB}^Y)$ la aserción que posea menos variables en el condicionante, ya que ésta tiene menos complejidad muestreal. Los resultados de la experimentación sobre esta idea mostraron que de este modo IB-score poseía una superficie sobre el espacio de estructuras más inestable, perdiendo calidad considerablemente. Sin embargo, esta vía de exploración no ha sido concluida, y es propicio estudiar y analizar formas alternativas de determinar una estructura con aserciones de independencia.

6.2.3. Diseño de métodos de búsqueda alternativos

En el presente trabajo se describen una serie de instanciaciones del enfoque, variando según el algoritmo de optimización utilizado. Además de la búsqueda por fuerza bruta, se presentan algunas maneras de instanciar el enfoque con búsquedas locales, algoritmos genéticos y búsqueda heurística. En el Capítulo 5

se presentan resultados que muestran que las diferentes búsquedas presentan calidades estructurales similares a la búsqueda por fuerza bruta. Adicionalmente, se demuestra empíricamente que estas búsquedas son realmente efectivas en la maximización. Dado el alcance de este trabajo, los resultados muestran que las instanciaciones descriptas son más que suficientes para mejorar el estado del arte actual en términos de calidad estructural. Sin embargo, en caso de obtener nuevas instanciaciones del IB-score y del cierre a utilizar (como proponen los apartados anteriores), el desarrollo de nuevo métodos de búsqueda en el espacio de estructuras podría ser vital para contribuir en el área. Algunos métodos de búsqueda que podrían implementarse rápidamente son: búsqueda por temple simulado, algoritmos meméticos, EDAs, entre otros.

6.2.4. Aplicación de AD-tree para aceleración de enfoque IBMAP

Por último, sería interesante estudiar la utilización de la estructura de datos AD-tree para acelerar el cómputo de IB-score en las distintas búsquedas sobre el espacio de estructuras. A primera vista, la estructura AD-tree estática (Moore y Lee, 1998) presenta una serie de estrategias para acelerar drásticamente la construcción de tablas de contingencia. Esto tendría un gran impacto sobre la complejidad temporal de los algoritmos del enfoque IBMAP. El problema que posee esta estructura de datos es que su complejidad espacial crece exponencialmente con el tamaño del dominio, y resulta prácticamente imposible de almacenar para dominios de gran tamaño, como los utilizados en la experimentación del Capítulo 5. Además, en muchos casos el tamaño del dominio es tan grande que el uso de esta estructura de datos es contraproducente, ya que el costo de construir la caché supera al costo de no utilizarla. Sin embargo, posteriormente se presentó una estructura que incorpora algunas mejoras sobre AD-tree, llamada dynamic AD-tree (Komarek y Moore, 2000), utilizando un enfoque perezoso para cargar en memoria solamente las partes de la estructura que son consultadas. Esta estructura escala mejor que el AD-tree estático en el número de atributos. En este respecto, sería interesante estudiar a fondo cómo aplicar esta estructura de datos en algoritmos basados en independencia, y analizar el impacto de su uso en la eficiencia. En particular, estudiar cómo el uso de esta estructura de datos

puede acelerar la ejecución de los algoritmos que utilizan el enfoque IBMAP.

6.2.5. Comentarios finales

En caso de que esta investigación crezca según las líneas sugeridas en los apartados anteriores, sería conveniente abstraer el diseño del enfoque de modo que soporte la instanciación de diversas formas del IB-score, nuevos conjuntos de cierre, nuevos espacios de estructuras, nuevos métodos de búsqueda, y cualquier combinación de estas piezas.

Por último, si se cambia el alcance del enfoque actual, se desprenden varias líneas de investigación interesantes. Por ejemplo, podría extenderse el alcance de IBMAP hacia el aprendizaje de redes de Markov completas. En esta línea de investigación debiera involucrarse el aprendizaje paramétrico de la estructura, quizás como parte de la búsqueda. Además, al cambiar la representación del modelo aprendido, la calidad de los modelos se evaluaría en término de otras medidas, como la eficacia de inferencia, la KL-divergence de la distribución aprendida, el Conditional marginal pseudo-likelihood, entre otros. También sería necesario comparar la calidad respecto de algoritmos basados en puntaje, como los descriptos en la Sección 2.2.3.1. Similarmente, se podría extender el alcance de IBMAP para aprendizaje de redes de Bayes, o para aprendizaje de otros modelos probabilísticos.

Apéndice A

Test Bayesiano de independencia condicional

En este apéndice se describe en detalle el test estadístico Bayesiano de independencia condicional (Margaritis, 2005). Este test de independencia es importante para el enfoque IBMAP, ya que es el único que computa las probabilidades a posteriori de independencia o de dependencia condicional. Específicamente, se explica el funcionamiento interno de dicho test en detalle, cómo se computan sus estadísticos, y se provee de un pseudo-código detallando cómo implementar el mismo.

El test Bayesiano permite la consulta de independencia condicional entre dos variables aleatorias X e Y, dado un conjunto condicionante $\mathbf{Z}$, en un conjunto de datos D con M puntos de datos. Comúnmente, D está estructurado en un formato tabular, con una columna por cada una de las variables aleatorias del dominio $\mathbf{V}$, y una fila por cada punto de datos (asignaciones completas del dominio). El test funciona realizando una comparación de la probabilidad a posteriori de dos modelos estadísticos: el modelo independiente M_{CI}, y el modelo dependiente $M_{\neg CI}$.

La probabilidad a posteriori del modelo independiente $P(M_{CI} \mid D)$ se computa en el conjunto de datos D utilizando la siguiente fórmula:

$$P(M_{CI} \mid D) = 1/\left(1 + \frac{1 - P(M_{CI})}{P(M_{CI})} \cdot \frac{P(D \mid M_{\neg CI})}{P(D \mid M_{CI})}\right), \tag{A.1}$$

donde $P(M_{CI})$ denota la probabilidad a priori del modelo independiente, $P(D \mid$

M_{CI}) es la verosimilitud de los datos dado que el modelo es independiente, y $P(D \mid M_{\neg CI})$ es la verosimilitud de los datos dado que el modelo es dependiente. La probabilidad a posteriori del modelo dependiente es su complemento a 1, es decir: $P(M_{\neg CI} \mid D) = 1 - P(M_{CI} \mid D)$.

Para computar la fórmula de la Ecuación (A.1), se requiere computar las verosimilitudes $P(D \mid M_{CI})$ y $P(D \mid M_{\neg CI})$, correspondientes a los modelos independiente y dependiente, respectivamente. La verosimilitud del modelo independiente sobre la distribución conjunta de (X, Y) se computa mediante el producto de la verosimilitud en cada una de las "slices" de $\mathbf{Z}$ (es decir, cada posible asignación completa de las variables en $\mathbf{Z}$), ya que se asume que los datos son disjuntos e independientes para cada slice. Denotando K al número de slices, la verosimilitud del modelo independiente se computa del siguiente modo:

$$P(D \mid M_{CI}) = \prod_{k=1}^{K} P(D^k \mid M_{CI}^k) = \prod_{k=1}^{K} g_k, \tag{A.2}$$

donde D^k es un subconjunto de D correspondiente a la slice k, y g_k se computa como sigue:

$$g_k = P(D^k \mid M_{CI}^k) = \left(\frac{\Gamma(\alpha)}{\Gamma(\alpha + M)} \prod_{i=1}^{I} \frac{\Gamma(\alpha_i + c_i)}{\Gamma(\alpha_i)} \right) \left(\frac{\Gamma(\beta)}{\Gamma(\beta + M)} \prod_{j=1}^{J} \frac{\Gamma(\beta_j + c_j)}{\Gamma(\beta j)} \right). \tag{A.3}$$

La forma de esta ecuación corresponde al uso de dos priors Dirichlet independientes. La elección del uso de una prior Dirichlet se debe a razones de efectividad computacional. Los valores α y β son hiper-parámetros, y c_i, c_j son las cantidades de ocurrencias de las variables X e Y en D^K. Los hiper-parámetros α y β se obtienen sumando sobre todos los hiper-parámetros α_i, y β_j, respectivamente. Las cardinalidades de X e Y son I y J respectivamente. La función gamma Γ se define como $\Gamma(x) = \int_0^{+\infty} e^{-t} y^{x-1}\, dt$. Cuando x es un entero no-negativo, $\Gamma(x+1) = x!$.

Para el modelo dependiente, la verosimilitud es un poco más compleja de computar, y consiste en sumar sobre todos los valores posibles de independencia y dependencia para las slices del conjunto condicionante. Como se describe en Margaritis y Bromberg (2009), esto se computa como:

$$P(D \mid M_{\neg CI}) = \frac{\prod_{k=1}^{K} p_k g_k + q_k h_k - \prod_{k=1}^{K} p_k g_k}{P(M_{\neg CI})}, \tag{A.4}$$

donde g_k ya ha sido computado con la Ecuación (A.3), $p_k = P(M_I^k) = P(M_{CI}^{1/K})$ es la prior del modelo independiente en el slice k, $q_k = P(M_I^k) = 1 - p_k$ es la prior del modelo dependiente en el slice k, y h_k es la verosimilitud del modelo para el slice k, que se computa como sigue:

$$h_k = P(D^k \mid M_{\neg CI}^k) = \frac{\Gamma(\gamma)}{\Gamma(\gamma + M)} \prod_{i=1}^{I} \prod_{j=1}^{J} \frac{\Gamma(\gamma_{ij} + c_{ij})}{\Gamma(\gamma_{ij})}. \tag{A.5}$$

Los valores γ y γ_{ij} son hiper-parámetros, y c_{ij} son las frecuencias de las variables X e Y en D^K. Los hiper-parámetros *gamma* se obtienen sumando sobre todos los hiper-parámetros γ_{ij}. Hacia el final de este apéndice, en la Sección A.1 se detalla un desarrollo matemático para obtener la expresión de la Ecuación (A.4).

En el Algoritmo 9 se muestra un pseudo-código del tests estadístico Bayesiano. Primeramente, el conjunto de slices del conjunto de datos se obtiene generando todas las configuraciones de las variables del conjunto condicionante $\mathbf{Z}$. Segundo, se inicializan las priors $\gamma_i j, \alpha_i, \beta_j$ y $P(M_{CI})$. En nuestra implementación, el test utiliza los mismos valores en los hiper-parámetros que se usan en Margaritis y Bromberg (2009), que son: $\gamma_{ij} = 1, \alpha_i = 2, \beta_j = 2$ y $P(M_{CI}) = 0.5$. Luego, por cada slice k de $\mathbf{Z}$, se computa la verosimilitud del modelo independiente y del modelo dependiente, usando la Ecuación (A.3) y la Ecuación (A.5), respectivamente. Luego, la verosimilitud de todas las slices se combinan en $P(D \mid M_{CI})$ y $P(D \mid M_{\neg CI})$ utilizando la Ecuación (A.2) y la Ecuación (A.4), respectivamente. Finalmente, $P(M_{CI} \mid D)$ se obtiene con la Ecuación (A.1), y $P(M_{\neg CI} \mid D) = 1 - P(M_{CI} \mid D)$. El test de independencia retorna verdadero cuando $P(M_{CI} \mid D) > P(M_{\neg CI} \mid D)$ y falso en caso contrario. La implementación de todas las fórmulas dadas se efectúa en el espacio logarítmico, para evitar problemas de desbordamiento aritmético.

A.1. Verosimilitud del modelo dependiente

En esta sección se describe un desarrollo sobre cómo obtener la verosimilitud del modelo dependiente, es decir, cómo se obtiene la fórmula de la Ecuación (A.4). En las publicaciones originales del test estadístico Bayesiano no se detalla cómo se ha obtenido dicha fórmula. Por esto, este apéndice publica un desarrollo posible.

El desarrollo, en primer lugar, comienza aplicando la regla de Bayes a la definición de $P(D \mid M_{\neg CI})$ y luego aplicando regla de la cadena al numerador, se

Algoritmo 9 Test estadístico Bayesiano$(D, X, Y, \mathbf{Z})$

1: $slices \leftarrow$ obtener slices de $\mathbf{Z}$
2: inicializar priors $\gamma_i j, \alpha_i, \beta_j$ y $P(M_{CI})$
3: **para toda** slice $k \in \mathbf{Z}$ **hacer**
4: $\quad g_k \leftarrow$ computar verosimilitud utilizando la Ecuación (A.3)
5: $\quad h_k \leftarrow$ computar verosimilitud utilizando la Ecuación (A.5)
6: $P(D \mid M_{CI}) \leftarrow$ computar utilizando la Ecuación (A.2)
7: $P(D \mid M_{\neg CI}) \leftarrow$ computar utilizando la Ecuación (A.4)
8: $P(M_{CI} \mid D) \leftarrow$ computar utilizando la Ecuación (A.1)
9: $P(M_{\neg CI} \mid D) \leftarrow 1 - P(M_{CI} \mid D)$
10: **si** $P(M_{CI} \mid D) > P(M_{\neg CI} \mid D)$ **entonces**
11: $\quad$ **retornar verdadero**
12: **sino**
13: $\quad$ **retornar falso**

obtiene:

$$P(D \mid M_{\neg CI}) = \frac{P(D, M_{\neg CI})}{P(M_{\neg CI})} = \frac{P(M_{\neg CI} \mid D) \cdot P(D)}{P(M_{\neg CI})}. \tag{A.6}$$

Como $P(M_{\neg CI} \mid D) = 1 - P(M_{CI} \mid D)$, puede re-expresarse la Ecuación (A.6) como:

$$P(D \mid M_{\neg CI}) = \frac{(1 - P(M_{CI} \mid D)) \cdot P(D)}{P(M_{\neg CI})},$$

que es igual a:

$$P(D \mid M_{\neg CI}) = \frac{P(D) - P(M_{CI} \mid D)P(D)}{P(M_{\neg CI})}. \tag{A.7}$$

Ahora, utilizando la ley de probabilidad total, el término $P(D)$ puede descomponerse como:

$$P(D) = P(D, M_{CI}) + P(D, M_{\neg CI}) = P(D, M_{CI}) + P(D, M_{\neg CI}),$$

y aplicando regla de la cadena, puede re-expresarse como:

$$P(D) = P(D \mid M_{CI})P(M_{CI}) + P(D \mid M_{\neg CI})P(M_{\neg CI}). \tag{A.8}$$

Mediante la descomposición de la Ecuación (A.8) en el producto de las verosimilitudes incondicionales para cada slice k, se obtiene:

$$P(D) = \prod_{k=1}^{K} P(D \mid M_I^k)P(M_I^k) + P(D \mid M_{\neg I}^k)P(M_{\neg I}^k). \tag{A.9}$$

Para abreviar, denotamos $p_k = P(M_I^k)$, $g_k = P(D \mid M_I^k)$, $q_k = P(M_{\neg I}^k)$, $h_k = P(D \mid M_{\neg I}^k)$, y la ecuación la Ecuación (A.9) ahora puede expresarse como sigue:

$$P(D) = \prod_{k=1}^{K} g_k p_k + h_k q_k. \tag{A.10}$$

Finalmente, como el término $P(M_{CI} \mid D)P(D)$ del numerador en la Ecuación (A.7) puede re-expresarse como $P(M_{CI}, D)$, y entonces expresarse como la otra condicional $P(D \mid M_{CI})P(M_{CI})$, descomponiendo este término en el producto de las verosimilitudes incondicionales correspondientes para cada slice k, puede expresarse como:

$$P(M_{CI} \mid D)P(D) = \prod_{k=1}^{K} P(D \mid M_I^k)P(M_I^k),$$

ó utilizando la notación abreviada:

$$P(M_{CI} \mid D)P(D) = \prod_{k=1}^{K} g_k p_k. \tag{A.11}$$

Por último, si en la Ecuación (A.7) se reemplaza $P(D)$ por el término de la derecha de la Ecuación (A.10) y $P(M_{CI} \mid D)P(D)$ por el término de la derecha de la Ecuación (A.11), se obtiene:

$$P(D \mid M_{\neg CI}) = \frac{\prod_{k=1}^{K} g_k p_k + h_k q_k - \prod_{k=1}^{K} g_k p_k}{P(M_{\neg CI})}, \tag{A.12}$$

que es justamente la fórmula mostrada en la Ecuación (A.4).

A. TEST BAYESIANO DE INDEPENDENCIA CONDICIONAL

Apéndice B

Correctitud del conjunto de cierre basado en mantas de Markov

En este apéndice se presenta una prueba formal de que el conjunto de cierre basado en mantas de Markov descripto en la Definición 5 de la Sección 4.2 es de hecho un cierre, es decir, sus aserciones de independencia determinan completamente la estructura utilizada. Para empezar, es necesario reproducir un resultado teórico importante: la propiedad de Markov de a pares (más conocida en inglés, como *pairwise Markov property*) (Koller y Friedman, 2009; Lauritzen, 1996; Pearl, 1988), que se satisface por cualquier red de Markov G de una distribución positiva e isomorfa a grafos P:

Definición 6 (Propiedad de Markov de a pares)**.** *Sea G una red de Markov de alguna distribución isomorfa a grafos P, y sea $E(G)$ el conjunto de las aristas en el grafo G, entonces:*

$$(X, Y) \notin E(G) \Leftrightarrow \ (X \perp\!\!\!\perp Y | V \setminus \{X, Y\}) \ \textit{en} \ P. \tag{B.1}$$

Ahora, basándonos además en los axiomas de Intersección y Unión Fuerte (ver la Sección 2.1.3), presentamos dos lemas auxiliares que relacionan las aserciones de independencia y de dependencia con las aristas de un grafo. Primeramente, se presenta un lema respecto de las aserciones de independencia del conjunto de cierre basado en mantas de Markov.

Lema 1.

$$(X \perp\!\!\!\perp Y | \mathbf{MB}^X \setminus \{Y\}) \Rightarrow \ (X, Y) \notin E(G). \tag{B.2}$$

Demostración. La demostración de este lema es sencilla. Primero, se puede aplicar la propiedad de Unión Fuerte al lado izquierdo de la Ecuación (B.2), para obtener $(X \perp\!\!\!\perp Y | \mathbf{V} \setminus \{X, Y\})$. Luego, si se aplica la propiedad de Markov de a pares de la Ecuación (B.1) se obtiene exactamente lo que dice el lado derecho de la Ecuación (B.2), i.e., $(X, Y) \notin E(G)$. □

Adicionalmente, se presenta un lema más, respecto de las aserciones de dependencia del conjunto de cierre basado en mantas de Markov. Para esto es necesario argumentar algo similar para la contra-positiva de la Ecuación (B.2):

Lema 2.

$$(X \not\perp\!\!\!\perp Y | \mathbf{MB}^X \setminus \{Y\}) \ \wedge \ \forall W \notin \mathbf{MB}^X (X \perp\!\!\!\perp W | \mathbf{Z}, Y) \Rightarrow \ (X, Y) \in E(G). \quad \text{(B.3)}$$

Demostración. La demostración de este lema es un poco más compleja que la demostración del Lema 1. Primeramente, al igual que en la demostración anterior, se debe aplicar la propiedad de Unión Fuerte para extender el conjunto condicionante $\mathbf{MB}^X \setminus \{Y\}$ del lado izquierdo de la Ecuación (B.3) hasta el dominio completo $\mathbf{V} \setminus \{X, Y\}$. Luego, utilizando la contra-positiva de la Ecuación (B.1) se obtiene el lado derecho de la Ecuación (B.3), i.e., $(X, Y) \in E(G)$.

En detalle, la demostración puede llevarse a cabo del siguiente modo. En una primer iteración, se toma del lado izquierdo de la Ecuación (B.3) la dependencia $(X \not\perp\!\!\!\perp Y | \mathbf{MB}^X \setminus \{Y\})$ y la independencia $(X \perp\!\!\!\perp W | \mathbf{Z}, Y)$ para una primer variable W. Aplicando la propiedad de intersección de la Ecuación (2.6), y tomando que $\mathbf{Z} = \mathbf{MB}^X \setminus \{Y\}$, se puede obtener la dependencia $(X \not\perp\!\!\!\perp Y | \mathbf{Z}, W)$. Luego, tomando esta dependencia resultante $(X \not\perp\!\!\!\perp Y | \mathbf{Z}, W)$ puede pasarse a la siguiente iteración, y aplicar la propiedad de intersección para otra aserción de independencia con otra variable W, denotada W' por conveniencia. Utilizando nuevamente la propiedad de Unión Fuerte a una aserción de la forma $(X \perp\!\!\!\perp W' | \mathbf{Z}, Y)$, puede obtenerse particularmente $(X \perp\!\!\!\perp W' | \mathbf{Z}, W, Y)$, a la que puede aplicarse Intersección para obtener finalmente $(X \not\perp\!\!\!\perp Y | \mathbf{Z}, W, W')$. El proceso se completa si se realiza lo mismo iterativamente para todas las variables restantes $\forall W \notin \mathbf{MB}^X$, llegando a obtener $(X \not\perp\!\!\!\perp Y | \mathbf{V} \setminus \{X, Y\})$. Por último, generando la contra-positiva de la propiedad de Markov de a pares mostrada en la Ecuación (B.1), se ve claramente que la aserción $(X \not\perp\!\!\!\perp Y | \mathbf{V} \setminus \{X, Y\})$ implica la existencia de una arista en el grafo, i.e.,

$$(X, Y) \in E(G) \Leftrightarrow \ (X \not\perp\!\!\!\perp Y | V \setminus \{X, Y\}) \text{ en } P, \quad \text{(B.4)}$$

donde el lado izquierdo es exactamente igual al lado derecho de la Ecuación (B.3). □

Finalmente, con ambos lemas puede probarse nuestro teorema principal de un modo muy sencillo:

Teorema 1. *Sea G una estructura de independencias no dirigida de una distribución positiva isomorfa a grafos $P(\mathbf{V})$. El* conjunto de cierre basado en mantas de Markov *de G es un conjunto de aserciones de independencia condicional que es suficiente para determinar completamente la estructura G.*

Demostración. Probamos el teorema demostrando que todas las aristas y no aristas en G están determinadas por las aserciones de independencia contenidas en el conjunto de cierre basado en mantas de Markov $\mathcal{C}(G)$. Demostramos esto separadamente para ausencia y existencia de aristas entre dos variables X e Y:

i) **Para ausencia de arista:** Sea $(X, Y) \notin E(G)$. Entonces, por definición, el cierre contiene las dos aserciones de independencia: $(X \perp\!\!\!\perp Y | \mathbf{MB}^X \setminus \{Y\})$ y $(Y \perp\!\!\!\perp X | \mathbf{MB}^Y \setminus \{X\})$, que, por la Ecuación (B.2) del Lema 1 ambas implican $(X, Y) \notin E(G)$.

ii) **Para existencia de arista:**
Similarmente, sea $(X, Y) \in E(G)$. Entonces, por definición, el cierre contiene la aserción de dependencia: $(X \not\perp\!\!\!\perp Y | \mathbf{MB}^X \setminus \{Y\})$. También, para toda variable W tal que $(X, W) \notin E(G)$ (i.e., $W \notin \mathbf{MB}^X$), el cierre contiene $(X \perp\!\!\!\perp W | \mathbf{MB}^X)$. Entonces, por la Ecuación (B.3) del Lema 2 tenemos que $(X, Y) \in E(G)$.

□

B. CORRECTITUD DEL CONJUNTO DE CIERRE BASADO EN MANTAS DE MARKOV

Apéndice C

El Algoritmo HHC-MN

Este apéndice explica brevemente cómo funciona HHC, y como se adaptó para que el mismo aprenda redes de Markov, resultando en el algoritmo HHC-MN que presentamos en el Capítulo 5 como algoritmo competidor.

El algoritmo HHC (Aliferis et al., 2010b) aprende una estructura mediante el aprendizaje del conjunto PC de cada variable, es decir, mediante el aprendizaje de los hijos y los padres (PC significa parents and children) de cada variable. Para esto, HHC aprende la estructura global utilizando la estrategia LGL explicada en el Algoritmo 1, y utilizando el algoritmo HITON-PC (Aliferis et al., 2003, 2010a) para aprender la manta de Markov de cada variable aleatoria.

El pseudo-código de HITON-PC puede verse en el Algoritmo 10, reproducido desde Aliferis et al. (2010a)[Figura 6, página 192]. El nombre completo de este algoritmo es *interleaved HITON-PC with symmetry correction*. Para aprender el PC de una variable X, este algoritmo comienza con un conjunto candidato vacío $\mathbf{PC}(X)$, ordenando el resto de las variables del dominio por prioridad, según asociación con X. La prioridad se genera según la asociación con X, la que se obtiene mediante un test estadístico incondicional. La lista que contiene este orden de asociación se llama ABIERTA. Luego, se descartan las variables que se encuentran independientes de X incondicionalmente. A continuación, el algoritmo utiliza una heurística de inclusión que acepta cada variable en el conjunto $\mathbf{PC}(X)$ candidato. Si alguna variable en $\mathbf{PC}(X)$ es independiente de X dado algún subconjunto del conjunto $\mathbf{PC}(X)$, entonces ésta es removida del conjunto $\mathbf{PC}(X)$, y ya no vuelve a ser considerada. La función de inclusión y la estrategia

de eliminación se iteran intercaladamente hasta que ya no hay más variables para incluir.

La complejidad del algoritmo HITON-PC es $O(n2^{\tau})$, donde τ es el máximo tamaño del conjunto PC encontrado. Por esto, la complejidad de HHC es $O(n^2 2^{\tau})$, ya que HITON-PC se ejecuta una vez por cada una de las variables del dominio. Para redes de Markov, el conjunto equivalente al conjunto PC de una variable son sus variables vecinas, que se corresponde con la manta de Markov. Por esta razón, se espera que HITON-PC aprenda la manta de Markov de redes de Markov, y entonces puede utilizarse como parte del algoritmo HHC para aprender estructuras no dirigidas. Esto no se prueba analíticamente en este trabajo. pero sí se confirma empíricamente para todos los casos considerados en el Capítulo 5. Para obtener una red de Markov, simplemente debe omitirse el último paso de HHC, que orienta las aristas. Por esto, se denota a tal algoritmo como HHC-MN.

Algoritmo 10 HITON-PC($X, \mathbf{V}$).

/* Inicialización */
$\mathbf{PC}(X) \longleftarrow \emptyset$.
/* Función heurística de inclusión */
Ordenar en lista ABIERTA las variables de $\mathbf{V} - \{X\}$, según asociación con X
Remover de ABIERTA las variables sin asociación con X
Insertar al final de $\mathbf{PC}(X)$ la primer variable de ABIERTA y removerla de ABIERTA
/* Estrategia de eliminación */
para cada $Y \in \mathbf{PC}(X)$ **hacer**
  si $(\exists \mathbf{Z} \subseteq \mathbf{PC}(X) - \{Y\} : (X \perp\!\!\!\perp Y | \mathbf{Z}))$ **entonces**
    remover Y de $\mathbf{PC}(X)$
/* Estrategia de intercalado */
repetir
  heurística de inclusión y estrategia de eliminación
hasta que $ABIERTA = \emptyset$
retornar$\mathbf{PC}(X)$

Bibliografía

A. Agresti. *Categorical Data Analysis*. Wiley, 2nd edition, 2002.

M. Alden. *MARLEDA: Effective Distribution Estimation Through Markov Random Fields*. PhD thesis, Dept of CS, University of Texas Austin, 2007.

C. Aliferis, I. Tsamardinos, y A. Statnikov. HITON, a novel Markov blanket algorithm for optimal variable selection. *AMIA Fall*, 2003.

C. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, y X. Koutsoukos. Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part I: Algorithms and Empirical Evaluation. *JMLR*, 11:171–234, March 2010a. ISSN 1532-4435.

C. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, y X. Koutsoukos. Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part II: Analysis and Extensions. *JMLR*, 11:235–284, March 2010b. ISSN 1532-4435.

L. Amgoud y C. Cayrol. A Reasoning Model Based on the Production of Acceptable Arguments. *Annals of Mathematics and Artificial Intelligence*, 34:197–215, March 2002. ISSN 1012-2443.

D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, y A. Ng. Discriminative Learning of Markov Random Fields for Segmentation of 3D Range Data. *Proceedings of the CVPR*, 2005.

A. Asuncion y D. Newman. UCI machine learning repository, 2007.

BIBLIOGRAFÍA

F. Barahona. On the computational complexity of Ising spin glass models. *Journal of Physics A: Mathematical and General*, 15(10):3241–3253, 1982.

J. Besag. Efficiency of pseudolikelihood estimation for simple Gaussian fields. *Biometrica*, 64:616–618, 1977.

J. Besag, J. York, y A. Mollie. Bayesian image restoration with two applications in spatial statistics. *Annals of the Inst. of Stat. Math.*, 43:1–59, 1991.

C. Boutilier, N. Friedman, M. Goldszmidt, y Daphne Koller. Context-specific independence in Bayesian networks. In *Proceedings of the Twelfth international conference on Uncertainty in artificial intelligence*, pages 115–123. Morgan Kaufmann Publishers Inc., 1996.

F. Bromberg y D. Margaritis. Efficient and robust independence-based Markov network structure discovery. In *Proceedings of IJCAI*, January 2007.

F. Bromberg y D. Margaritis. Improving the Reliability of Causal Discovery from Small Data Sets using Argumentation. *JMLR*, 10:301–340, Feb 2009.

F. Bromberg y F. Schlüter. Variante de Grow Shrink para mejorar la calidad de Markov blankets. In *XXXV Latin American Informatics Conference, Pelotas, Brasil.*, September 2009. http://dharma.frm.utn.edu.ar/fschluter/p/09b.pdf.

F. Bromberg, D. Margaritis, y V. Honavar. Efficient Markov network structure discovery using independence tests. In *In Proc SIAM Data Mining*, page 06, 2006.

F. Bromberg, D. Margaritis, y V. Honavar. Efficient Markov network structure discovery using independence tests. *JAIR*, 35:449–485, July 2009.

F. Bromberg, F. Schlüter, y A. Edera. Independence-based MAP for Markov networks structure discovery. In *23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 497–504. IEEE, 2011.

K. Cai, J. Bu, C. Chen, y G. Qiu. A novel dependency language model for information retrieval. *Journal of Zhejiang University - Science A*, 8:871–882, 2007. ISSN 1673-565X. 10.1631/jzus.2007.A0871.

W. Cochran. Some methods of strengthening the common χ tests. *Biometrics.*, page 10:417–451, 1954.

G. Cooper. The computational complexity of probabilistic inference using bayesian belief networks. *Artificial Intelligence*, 42(2-3):393 – 405, 1990. ISSN 0004-3702. doi: DOI:10.1016/0004-3702(90)90060-D.

T. Cover y J. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991. ISBN 0-471-06259-6.

N. Cressie. Statistics for spatial data. *Terra Nova*, 4(5):613–617, 1992. ISSN 1365-3121. doi: 10.1111/j.1365-3121.1992.tb00605.x.

J. Davis y P. Domingos. Bottom-up learning of Markov network structure. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 271–278, 2010.

S. Della Pietra, V. Della Pietra, y J. Lafferty. Inducing Features of Random Fields. *IEEE Trans. PAMI.*, 19(4):380–393, 1997.

A. Edera, F. Schlüter, y F. Bromberg. Learning Markov networks with context-specific independences. In *IEEE 25th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 553–560, Nov 2013. doi: 10.1109/ICTAI.2013.88.

N. Friedman, M. Linial, I. Nachman, y D. Pe'er. Using Bayesian Networks to Analyze Expression Data. *Computational Biology*, 7:601–620, 2000.

S. Fu y M. C. Desmarais. Fast Markov blanket discovery algorithm via local learning within single pass. In *Proceedings of the Canadian Society for computational studies of intelligence, 21st conference on Advances in artificial intelligence*, Canadian AI'08, pages 96–107, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 3-540-68821-8, 978-3-540-68821-1.

V. Ganapathi, D. Vickrey, J. Duchi, y D. Koller. Constrained Approximate Maximum Entropy Learning of Markov Random Fields. In *Uncertainty in Artificial Intelligence*, pages 196–203, 2008.

BIBLIOGRAFÍA

P. Gandhi, F. Bromberg, y D. Margaritis. Learning Markov Network Structure using Few Independence Tests. In *SIAM International Conference on Data Mining*, pages 680–691, 2008.

J. Hammersley y P. Clifford. Markov fields on finite graphs and lattices. 1971.

D. Heckerman, D. Geiger, y D. Chickering. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 1995.

S. Hettich y S. Bay. The UCI KDD archive, 1999.

H. Höfling y R. Tibshirani. Estimation of Sparse Binary Pairwise Markov Networks using Pseudo-likelihoods. *Journal of Machine Learning Research*, 10: 883–906, 2009.

A. Hyvärinen y P. Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–709, 2005.

V. Karyotis. Markov random fields for malware propagation: the case of chain networks. *Comm. Letters.*, 14:875–877, September 2010. ISSN 1089-7798.

D. Koller y N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

D. Koller y M. Sahami. Toward Optimal Feature Selection. pages 284–292. Morgan Kaufmann, 1996.

P. Komarek y A. Moore. A dynamic adaptation of ad-trees for efficient machine learning on large data sets. In *ICML*, pages 495–502. Citeseer, 2000.

W. Lam y F. Bacchus. Learning Bayesian belief networks: an approach based on the MDL principle. *Computational Intelligence*, 10:269–293, 1994.

P. Larrañaga y J. Lozano. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Pubs, 2002.

P. Larrañaga, H. Karshenas, C. Bielza, y R. Santana. A review on probabilistic graphical models in evolutionary computation. *Journal of Heuristics*, 18(5): 795–819, 2012.

S. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

S. Lee, V. Ganapathi, y D. Koller. Efficient structure learning of Markov networks using L1-regularization. In *NIPS*, 2006.

S. Li. *Markov random field modeling in image analysis*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2001. ISBN 4-431-70309-8.

D. Lowd y J. Davis. Improving markov network structure learning using decision trees. *Journal of Machine Learning Research*, 15:501–532, 2014.

D. Margaritis. Distribution-Free Learning of Bayesian Network Structure in Continuous Domains. In *Proceedings of AAAI*, 2005.

D. Margaritis y F. Bromberg. Efficient Markov Network Discovery Using Particle Filter. *Comp. Intel.*, 25(4):367–394, 2009.

D. Margaritis y S. Thrun. Bayesian network induction via local neighborhoods. In *Proceedings of NIPS*, 2000.

A. McCallum. Efficiently inducing features of conditional random fields. In *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, 2003.

D. Metzler y W. Croft. A Markov random field model for term dependencies. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 472–479, New York, NY, USA, 2005. ACM. ISBN 1-59593-034-5.

T. Minka. Algorithms for maximum-likelihood logistic regression. Technical report, Dept of Statistics, Carnegie Mellon University, 2001.

T. Minka. Power EP. Technical Report MSR-TR-2004-149, Microsoft Research, Cambridge, January 2004.

M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA, 1998. ISBN 0262631857.

J. Mooij. libDAI: A Free and Open Source C++ Library for Discrete Approximate Inference in Graphical Models. *J. Mach. Learn. Res.*, 11:2169–2173, August 2010. ISSN 1532-4435.

BIBLIOGRAFÍA

A. Moore y M. Lee. Cached suficient statistics for e cient machine learning with large datasets. *Journal of Artificial Intelligence Research*, 8:67–91, 1998.

H. Mühlenbein y G. Paaß. From recombination of genes to the estimation of distributions I. binary parameters. In Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, y Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature — PPSN IV*, volume 1141 of *Lecture Notes in Computer Science*, pages 178–187. Springer Berlin / Heidelberg, 1996. 10.1007/3-540-61723-X_982.

J. Peña, R. Nilsson, J. Björkegren, y J. Tegnér. Towards scalable and data efficient learning of Markov boundaries. *Int. J. Approx. Reasoning*, pages 211–232, 2007.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc., 1988.

J. Pearl y A. Paz. GRAPHOIDS : A graph based logic for reasonning about relevance relations. Technical Report 850038 (R-53-L), Cognitive Systems Laboratory, University of California, Los Angeles, 1985.

P. Ravikumar, M. Wainwright, y J. Lafferty. High-dimensional Ising model selection using L1-regularized logistic regression. *Annals of Statistics*, 38:1287–1319, 2010. doi: 10.1214/09-AOS691.

S. Russel y P. Norvig. *Artificial Intelligence. A Modern Approach.* 2nd Ed., 2002.

R. Santana. Estimation of distribution algorithms with kikuchi approximations. *Evol. Comput.*, 13(1):67–97, January 2005. ISSN 1063-6560. doi: 10.1162/1063656053583496.

F. Schlüter. A survey on independence-based Markov networks learning. *Artificial Intelligence Review*, pages 1–25, 2012. ISSN 0269-2821.

F. Schlüter, F. Bromberg, y S. Perez. Speeding up the execution of a large number of statistical tests of independence. In *Proceedings of ASAI 2010, Argentinean Symposioum of Artificial Intelligence*, pages 48–59, 2009.

F. Schlüter, F. Bromberg, y L. Abraham. Strategies for piecing-together local-to-global markov network learning algorithms. In *Proceedings of ASAI 2011, Argentinean Symposioum of Artificial Intelligence*, pages 96–107, 2011.

F. Schlüter, F. Bromberg, y A. Edera. The IBMAP approach for Markov network structure learning. *Annals of Mathematics and Artificial Intelligence*, pages 1–27, 2014. ISSN 1012-2443. doi: 10.1007/s10472-014-9419-5.

M. Schmidt, K. Murphy, G. Fung, y R. Rosales. Structure learning in random fields for heart motion abnormality detection. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1 –8, june 2008. doi: 10.1109/CVPR.2008.4587367.

S. Shakya y J. McCall. Optimization by estimation of distribution with deum framework based on markov random fields. *International Journal of Automation and Computing*, 4(3):262–272, 2007.

S. Shakya, R. Santana, y J. Lozano. A markovianity based optimisation algorithm. *Genetic Programming and Evolvable Machines*, 13(2):159–195, 2012.

S. Shekhar, P. Zhang, Y. Huang, y R. Vatsavai. Trends in Spatial Data Mining. In *Trends in Spatial Data Mining*, chapter 19, pages 357–379. AAAI Press / The MIT Press, 2004.

P. Spirtes, C. Glymour, y R. Scheines. *Causation, Prediction, and Search*. Adaptive Computation and Machine Learning Series. MIT Press, 2000.

I. Tsamardinos, C. Aliferis, y A. Statnikov. Algorithms for large scale Markov blanket discovery. In *FLAIRS*, 2003.

I. Tsamardinos, L. Brown, y C. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65:31–78, 2006.

J. Van Haaren y J. Davis. Markov network structure learning: A randomized feature generation approach. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

J. Van Haaren, J. Davis, M. Lappenschaar, y A. Hommersom. Exploring disease interactions using markov networks. In *Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

S. Vishwanathan, N. Schraudolph, M. Schmidt, y K. Murphy. Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 969–976, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2.

M. Wainwright y M. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Found. Trends Mach. Learn.*, 1:1–305, January 2008. ISSN 1935-8237. doi: 10.1561/2200000001.

M. Wainwright, T. Jaakkola, y A. Willsky. Tree-reweighted belief propagation algorithms and approximate ML estimation by pseudo-moment matching. In *In AISTATS*, 2003.

J. Winn y C. Bishop. Variational Message Passing. *J. Mach. Learn. Res.*, 6: 661–694, December 2005. ISSN 1532-4435.

S. Yaramakala y D. Margaritis. Speculative Markov blanket discovery for optimal feature selection. In *Data Mining, Fifth IEEE International Conference on*, page 4 pp., nov. 2005. doi: 10.1109/ICDM.2005.134.

J. Yedidia, W. Freeman, y Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *Information Theory, IEEE Transactions on*, 51(7):2282 – 2312, july 2005. ISSN 0018-9448. doi: 10.1109/TIT.2005.850085.