Uso del algoritmo de colonia de hormigas en el aprendizaje de redes bayesianas

Guillermo Ramos F., Abraham Sánchez L., Fabian Aguilar C.,
María B. Bernábe L., Rogelio González V.

Benemérita Universidad Autónoma de Puebla,
Computer Science Department,
México

sase.ramses@gmail.com, asanchez@cs.buap.mx, slipknot_1964@live.com.mx,
beatriz.bernabe@gmail.com, rgonzalez@cs.buap.mx

Resumen. Las redes bayesianas modelan un fenómeno mediante un conjunto de variables y las relaciones de dependencia entre ellas. Dado este modelo, se puede hacer inferencia bayesiana. Estos modelos pueden tener diversas aplicaciones, para clasificación, predicción, diagnóstico, etc. Además, pueden dar información interesante en cuanto a cómo se relacionan las variables, las cuales pueden ser interpretadas en ocasiones como relaciones de causa-efecto. El problema radica en que la obtención de la estructura de una red es un problema NP-Duro. En este trabajo se propone un algoritmo para el aprendizaje en redes bayesianas basado en una metaheurística que ha sido aplicada con éxito, la optimización de la colonia de hormigas. Se presentan varios ejemplos para validar nuestra propuesta y comparar en tiempo y clasificación dichos resultados con el algoritmo clásico K2 y el software GeNIe.

Palabras clave: Redes bayesianas, inferencia, aprendizaje, colonia de hormigas.

The Use of the Ant Colony Algorithm in the Learning of Bayesian Networks

Abstract. The bayesian networks model phenomena through a set of variables and their dependence. With these models is possible to develop bayesian inference. The models have several applications in data classification, data prediction, diagnostics, etc. Also, these models provide revelant information about the relationship between the variables and sometimes can be interpreted as a cause-effect relationship. The underlying problem is that the determination of the network structure is an NP-Hard. In this work an algorithm for learning in bayesian networks have been successfuly applied based on the ant colony metaheuristic. Several examples are presented to validate the proposed method and the results of time and classification are compared with those of the traditional algorithm $K2$ and the GeNIe software.

Keywords: Bayesian networks, inference, learning, ant colony.

1. Introducción

Las redes bayesianas (RBs), también conocidas como redes de creencias probabilísticas o redes causales, son herramientas de representación de conocimiento capaces de gestionar eficazmente las relaciones de dependencia/independencia entre las variables aleatorias que componen el dominio del problema que queremos modelar. Estas tienen dos componentes: a) una estructura gráfica, o más precisamente un grafo acíclico dirigido (DAG), y b) un conjunto de parámetros, que en conjunto especifican una distribución de probabilidad conjunta sobre las variables aleatorias.

Los parámetros son un conjunto de medidas de probabilidad condicional, que dan forma a las relaciones. Por esta razón, las RBs tienen muchas ventajas, por ejemplo: logran presentar las interrelaciones de los elementos como un todo, y no sólo por sus partes (por su representación multivariable; tratan el problema del ruido de los datos experimentales), describen las complejas relaciones de los elementos con naturaleza probabilística y no lineal, representan las relaciones causales de las interacciones y manejan eventos que no han sido observados, y la incertidumbre inherente a ellos, etc.

El problema es obtener la estructura y parámetros de una RB a partir de un conjunto de muestras de los elementos de la red. La dificultad radica en el número de posibles redes bayesianas en el espacio de búsqueda que crece en forma exponencial, este problema se define como NP-duro [1].

De forma general, podemos decir que el problema del aprendizaje consiste en construir, partir de un conjunto de datos, el modelo que mejor represente la realidad, mejor dicho, una porción del mundo real en la cual estamos interesados. Como en el caso de la construcción manual de redes bayesianas, el aprendizaje de este tipo de modelos tiene dos aspectos: a) el aprendizaje paramétrico, y b) el aprendizaje estructural [3], [2]. A continuación detallamos algunos conceptos que nos serviran como base a nuestra propuesta.

El presente trabajo se centra en el aprendizaje estructural de una RB en tiempo polinomial, mediante la metaheuristica de la colonia de hormigas (del inglés ACO, Ant Colony Optimization). Una vez determinada la red se compara su efectividad al momento de clasificar. Posteriormente se describe el algoritmo planteado, y para finalizar este trabajo se muestran los resultados de las RB obtenidas en tres ejemplos distintos, así como el tiempo que tomo decretarlas y su efectividad al momento de clasificar datos; comparando dichos resultados con el clásico algoritmo K2 y el programa Genie & smile.

2. Conceptos básicos

En esta sección se revisan brevemente algunos conceptos básicos relacionados con la RB y cómo aprender de ellos, así como otros conceptos relacionados con ACO [4].

2.1. Redes bayesianas

Las siguientes definiciones proveen un marco adecuado para describir en detalle, lo que es una RB desde la definición de un dígrafo dirigido acíclico (del inglés DAG, Directed Acyclic Graph), hasta la distribución de probabilidad que la red representa, mostrando su cualidad principal de independencia condicional.

El aprendizaje de la estructura de una RB se centraliza en la búsqueda de una estructura óptima para un conjunto de datos cuando se escoge el mejor puntaje entre esta; a esto se le conoce como aprendizaje basado en la búsqueda.

La red bayesiana es un DAG $G = (V, E)$, donde el conjunto de nodos "V" representará a nuestras variables de nuestro sistema $(X_1, X_2, ..., X_n)$, "E" como el conjunto de arcos, estas son las relaciones directas entre las variables. Para cada variable $x_i \in V$ tenemos una familia de distribuciones condicionales $P(x_i|Pa(x_i))$, donde $Pa(x_i)$ representa el conjunto de los padres de la variable x_i. A partir de estas distribuciones condicionales podemos recuperar la distribución conjunta sobre V:

$$P(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} P(x_i|Pa(x_i)). \tag{1}$$

Esta fórmula muestra la distribución conjunta por medio de la presencia o ausencia de nuestras conexiones entre pares de las variables. Una propiedad deseable e importante de una métrica es la forma en cómo se descompone en la presencia de datos completos, es decir, la función de puntuación se puede descomponer de la siguiente manera:

$$f(G : D) = \sum_{i=1}^{n} f(x_i, Pa(x_i) : N_{x_i,pa(x_i)}, \tag{2}$$

donde $N_{x_i,pa(x_i)}$ son las estadísticas de la variable x_i y $Pa(x_i)$ en D, es decir, el número de instancias en D que coincidan con cada posible creación de instancias de x_i y $Pa(x_i)$. En este caso se busca un procedimiento capaz de realizar una búsqueda local con la cual podamos cambiar la posición de los arcos para evaluarlos de una manera eficiente y reutilizar la mayor parte de los cálculos.

2.2. Algoritmo B

El algoritmo B es una heurística de construcción voraz, la cual en cada paso agrega un arco con el máximo aumento en el puntaje en la métrica f, pero evitando la inclusión de ciclos dirigidos [5]. El algoritmo termina cuando la adición de cualquier arco no incrementa el valor de la métrica.

2.3. Métrica K2

Entre las distintas técnicas para poder construir una red bayesiana a partir de sus datos, utilizamos el algoritmo K2, ya que este está basado en la optimización

1. INICIALIZACION
a) **for** $i = 1$ **to** n **do**
$Pa(x_i) = \emptyset$
b) **for** $i = 1$ and $j = 1$ **to** n **do**
if $(i \neq j)$ **then** $A[i, j] = f(x_i, x_j) - f(x_i, \emptyset)$
2. CICLO:
Repeat
i. Selecciona dos indices (i, j) tal que
$(i, j) = \arg \max_{i',j'} A[i', j']$
ii. **if** $A[i, j] > 0$ **then** $Pa(x_i) = Pa(x_i) \cup \{x_j\}$
iii. $A[i, j] = -\infty$
iv. **for all** $x_a \in$ Ancestros$(x_j) \cup \{x_j\}$ **and** $x_b \in$ Descendientes$(x_i) \cup \{x_i\}$ **do**
$A[a, b] = -\infty$
v. **for** $k = 1$ **to** n **do**
if $(A[i, k] > -\infty)$ then $A[i, k] = f(x_i, Pa(x_i) \cup \{x_k\}) - f(x_i, Pa(x_i))$
until $\forall i, j$ $(A[i, j] \leq 0$ or $A[i, j] = -\infty)$

Fig. 1. Algoritmo B

de una medida, y esta a su vez es utilizada para la exploración en el espacio formando las redes que contienen nuestras variables en la base de datos [6]. Parte de un DAG y va añadiendo arcos, modificándolos o eliminándolos para poder obtener una red con la mejor medida. Para una red G y una base de datos D, su medida K2 es la siguiente:

$$f(G : D) = \log P(G) + \sum_{i=1}^{n} \left[\sum_{k=1}^{si} \left[\log \frac{\Gamma(\eta_{ik})}{\Gamma(N_{ik}, \eta_{ik})} + \sum_{j=1}^{ri} \log \frac{\Gamma(N_{ik}, \eta_{ik})}{\eta_{ik}} \right] \right], \quad (3)$$

donde:

- N_{ik} es la frecuencia de las configuraciones encontradas en la base de datos D de las variables x_i.
- n es el número de variables, tomando su j-ésimo valor y sus padres en G tomando su k-ésima configuración, donde si es el número de configuraciones posibles del conjunto de padres.
- Γ_i es el número de valores que puede tomar la variable x_i.
- Además, $N_{ik} = \sum_{j=1}^{\Gamma_i} N_{ik}$.
- Γ es la función Gamma.
- El parámetro η puede interpretarse como el tamaño muestral equivalente.

La métrica ha adoptado el nombre del algoritmo, de modo que se conoce como la métrica K2, es decir:

Suponiendo una distribución uniforme a priori de $P(G)$ y usando $log(P(G))$ instanciado de $P(G, D)$, tenemos una métrica descomponible:

$$f_{K2}(G, D) = \sum_{i=1}^{n} f_{K2}(x_i, pa(x_i) : N_{x_i, pa(x_i)}), \tag{4}$$

$$f_{K2}(x_i, pa(x_i) : N_{x_i, pa(x_i)}) = \sum_{j=1}^{qi} \left(\log \left(\frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \right) + \sum_{k=1}^{ri} \log(N_{jnk}!) \right). \tag{5}$$

2.4. Optimización basada en colonia de hormigas

El algoritmo ACO está basado en como las hormigas reales realizan su búsqueda para encontrar comida o un nuevo hogar, ellas deciden cual es el camino más corto para poder encontrar el recurso que necesitan, cuando caminan dejan una pequeña sustancia en el suelo denominada "feromona" [7]. Solo ellas pueden detectar ese aroma y al elegir su camino, tienen que escoger de una manera probabilística los caminos que están marcados con más concentración de feromonas [8].

Cuando no hay feromonas en el lugar, escogen un camino al azar, pero después de un periodo transcurrido, los caminos más cortos serán frecuentados más rápido y a su vez, estos caminos tendrán más feromonas, y esto hace posible que las hormigas utilicen dichos caminos. Este efecto significa que las hormigas han encontrado el camino más corto, es decir, aunque una sola hormiga llegue a la solución, la solución óptima es el resultado del comportamiento cooperativo de las hormigas. Cada hormiga representa una posible solución al problema desplazándose a través de una secuencia finita de nodos, los movimientos que realizan son seleccionados aplicando una búsqueda local, resolviendo los problemas específicos de información local y de la información compartida sobre la feromona [7].

A la feromona la modelamos mediante una matriz τ, donde τ_{ij} contiene el nivel de feromona depositada en el arco del nodo i al nodo j. En los primeros sistemas de hormigas, una hormiga k en el nodo i seleccionará el siguiente nodo j para visitar con la probabilidad:

$$p_k(i, j) = \begin{cases} \frac{[\tau_{ij}]^{\alpha}[n_{ij}]^{\beta}}{\sum_{u \in j_k(i)}}, & \text{si } j \in j_k(i) \\ 0, & \text{en otro caso,} \end{cases} \tag{6}$$

donde n_{ij} representa la información heurística sobre el problema, $j_k(i)$ es el conjunto de nodos vecinos del nodo i que aún no han sido visitados por la hormiga k, α y β son dos parámetros que determinan la importancia relativa de la feromona con respecto a la información heurística.

En cada iteración del algoritmo de cada hormiga, utilizando la regla de transición anterior, se construye progresivamente una solución. La matriz de la feromona se actualiza de la siguiente manera:

- Actualización global: Sólo la hormiga, que construyó la mejor solución refuerza el nivel de feromona en los arcos que forman parte de la mejor solución S^+ obtenida hasta el momento. Esto dirige la búsqueda en el vecindario de la mejor solución.
- Actualización local: Al construir una solución, si una hormiga realiza la transición del nodo i al nodo j, entonces el nivel de feromonas del arco correspondiente se modifica. Esta regla favorece la exploración de otros arcos, evitando así la convergencia prematura; sin la actualización local de todas las hormigas que buscarían en todo el vecindario la mejor solución encontrada hasta el momento.
- Uso de un optimizador local: algunas o todas de las soluciones obtenidas por las hormigas están optimizadas a nivel local mediante el uso de un método de búsqueda local. Esta técnica es particularmente útil para muchos problemas de optimización combinatoria, donde en la práctica se obtienen mejores resultados cuando se acoplan este tipo de algoritmos con optimizadores locales.

3. Redes de aprendizaje bayesiano con colonia de hormigas

En esta sección mostraremos la definición de los componentes que necesitamos para poder aplicar la metaheurística ACO en nuestro problema:

- Representación adecuada del problema: Significa poder construir las posibles soluciones utilizando una regla de probabilidad para poder pasar de un estado i a un estado j.
- Información heurística: Conocimientos específicos que utilizan el proceso de la búsqueda n_{ij}, es decir cuando nos movemos del estado i al estado j.
- Regla(s) para actualizar la matriz de feromonas τ.
- Regla de transición probabilística que utiliza la heurística n y la feromona τ.
- Optimizador local.

Ahora, vamos a definir todos los componentes para nuestro problema de aprendizaje:

- Representación del problema: La representación del problema es un grafo donde los estados del problema son los DAGs con n nodos. Por lo tanto, un estado G_h será un grafo con los nodos $x_i \in V$ y exactamente h arcos y ningún ciclo dirigido. La construcción incremental de la hormiga en la solución se inicia desde el grafo vacío G_0 (arcs-less dag) y procede mediante la adición de un arco $x_j \to x_i$ para el estado actual G_h, es decir, $G_{h+1} = G_h \cup \{x_j \to x_i\}$. La solución final será el estado G_h en el que la hormiga decide dejar la fase de construcción.
- Información heurística:

$$n_{ij} = f(x_i, Pa(x_i) \cup \{x_j\}) - f(x_i, Pa(x_i)).$$

- Reglas de actualización de feromonas: Las reglas globales y locales de actualización se consideran las mismas que se han descrito anteriormente.
- Regla de transición probabilística: El siguiente arco va a ser incluido en el grafo actual G, por una hormiga seleccionada de una manera similar a la utilizada por el algoritmo B, pero utilizando una regla de decisión estocstica (en lugar de una regla determinista) que también tiene en cuenta la feromona depositada en cada arco.
- Optimizador local: Es una mejora donde, en cada paso, el mejor movimiento de acuerdo con la métrica y los operadores utilizados se seleccionan. La complejidad de estos movimientos es $O(n^2)$, donde n es el número de variables. Debemos tener en cuenta que si se utiliza una métrica descomponible, un gran número de cálculos puede ser reutilizado desde las etapas anteriores del algoritmo. También hay que tomar en cuenta que los operadores de transición elegidos contienen el elegido para la hormiga B. Por lo tanto, una vez que una hormiga ha obtenido una solución, entonces, mediante la supresión o inversión de un arco, el algoritmo puede escapar de un óptimo local eventual alcanzado por la hormiga.

4. Resultados experimentales

Para comparar los resultados se utilizaron tres conjuntos de datos disponibles en UCI Machine Learning Repository y se evaluaron las estructuras obtenidas por ACO con respecto al algoritmo K2 y la herramienta GeNIe & Smile de la Universidad de Pittsburgh. Se utilizaron los siguientes parámetros, $\alpha = 1$, $\beta = 2{,}0$, 10 hormigas y 20 iteraciones.

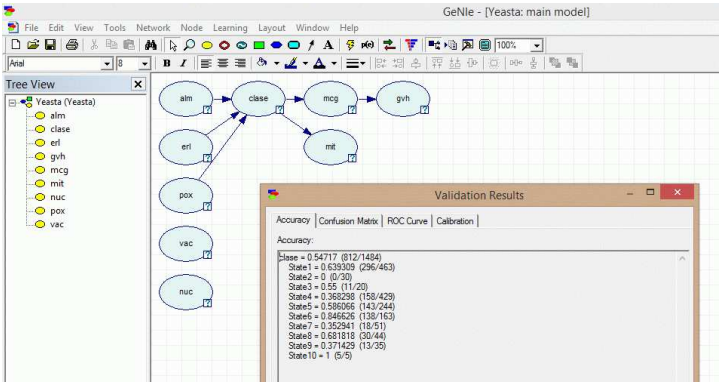
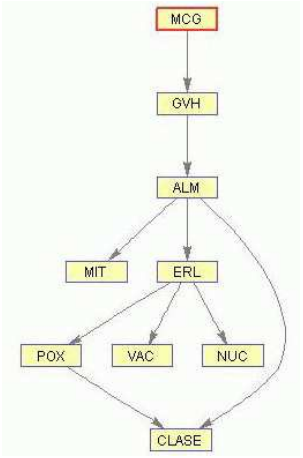


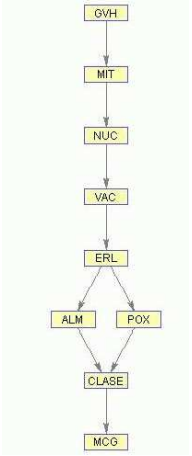
Fig. 2. RB para el ejemplo Yeast utilizando el software GeNIe

En los casos correspondientes donde hay registros incompletos, estos fueron llenados por medio de interpolación, no se utilizaron otras técnicas de preprocesamiento.



Clase	0.415094	616/1484
Estado 1	0.704104	326/463
Estado 2	0	0/30
Estado 3	0.55	11/20
Estado 4	0.317016	136/429
Estado 5	0.00409836	1/244
Estado 6	0.871166	142/163
Estado 7	0	0/51
Estado 8	0	0/44
Estado 9	0	0/35
Estado 10	0	0/5

Fig. 3. RB para el ejemplo Yeast utilizando el algoritmo K2



Clase	0.472372	701/1484
Estado 1	0.7473	346/463
Estado 2	0	0/30
Estado 3	0.55	11/20
Estado 4	0.314685	135/429
Estado 5	0.0409836	10/244
Estado 6	0.858896	140/163
Estado 7	0.25902	13/51
Estado 8	0.613636	27/44
Estado 9	0.542857	19/35
Estado 10	0	0/5

Fig. 4. RB para el ejemplo Yeast utilizando el algoritmo ACO

Las siguientes figuras muestran las RBs resultantes con la herramienta GeNIe y con los dos algoritmos propuestos, K2 y el algoritmo ACO.

En el primer conjunto de datos de nombre Yeast, obtenemos una clasificación baja por parte de las 3 herramientas, donde GeNIe obtiene el mejor resultado con casi 55 %, y le siguen ACO con 47 % y K2 con 41,5 %, ver las figuras 2, 3 y 4. Con tiempos de 1.83, 6.2 y 5.7 segundos respectivamente.

Para el conjunto de datos Mammographic Mass los resultados para GeNIe, ACO y K2 fueron 84,6 %, 83,6 % y 83,5 % respectivamente, ver las figuras 5, 6 y 7. Con tiempos de 1.03, 5.9 y 5.45 segundos respectivamente.

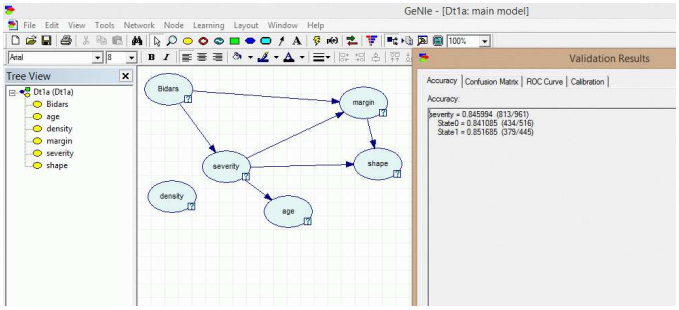


Fig. 5. RB para el ejemplo Mammographic mass utilizando el software GeNIe

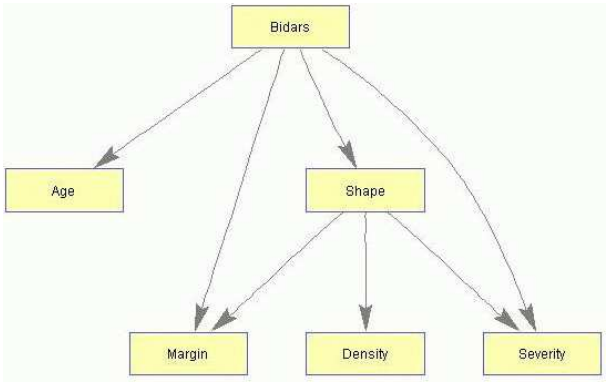


Fig. 6. RB para el ejemplo Mammographic mass utilizando el algoritmo K2

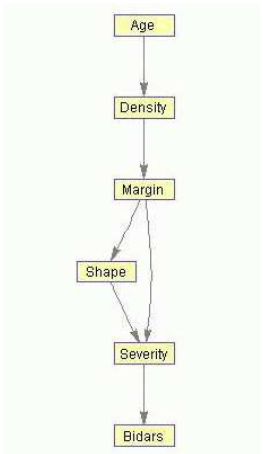


Fig. 7. RB para el ejemplo Mammographic mass utilizando el algoritmo ACO

Para el conjunto de datos Banknote Authentication los resultados para GeNIe, ACO y K2 fueron $95,6\,\%, 89{,}3\,\% y 82\,\%$ respectivamente, ver las figuras 8, 9 y 10. Con tiempos de 1.16, 5.78 y 5.14 segundos respectivamente.

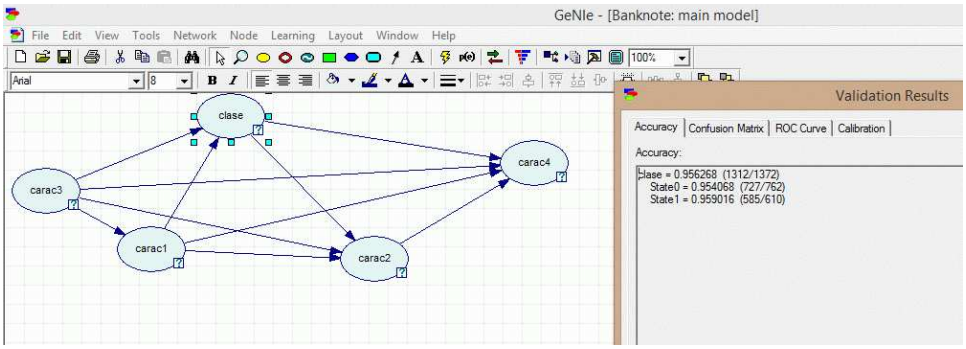


Fig. 8. RB para el ejemplo Banknote Authentication utilizando el software GeNIe

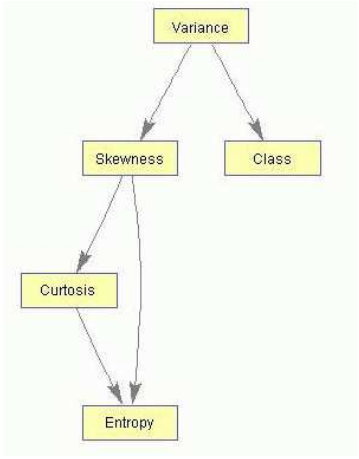


Fig. 9. RB para el ejemplo Banknote Authentication utilizando el algoritmo K2

Las estructuras de las RBs fueron obtenidas por las 3 propuestas (herramienta GeNIe, algoritmo K2 y algoritmo ACO) en pocos segundos, que es el objetivo principal de esta investigación.

Podemos concluir que nuestro algoritmo alcanzo resultados mejores al algoritmo K2, pero que algunas herramientas superan su desempeño, se espera mejorar el algoritmo para obtener mejores resultados en la obtención de estructuras para RBs.

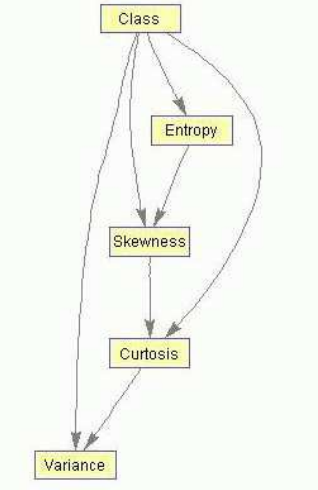


Fig. 10. RB para el ejemplo Banknote Authentication utilizando el algoritmo ACO

5. Conclusiones y trabajo futuro

En este trabajo, obtuvimos diferentes estructuras de una RB dependiendo del algoritmo utilizado y del software, ya que cada software hace de forma diferente la inferencia de los datos y algunos suelen ser mejores que otros, podemos concluir lo siguiente: Hay un buen rendimiento y logramos el objetivo, que fue evitar la búsqueda redundante e ineficiente de hacer todas las permutaciones posibles para obtener la red óptima con el algoritmo de la colonia de hormigas, mientras que en el K2, hay que darle el orden de cómo debe hacer la inferencia en la red.

De acuerdo con lo anterior, se plantea realizar, en trabajos futuros, el aprendizaje de la Red bayesiana, utilizando como base el algoritmo de Colonia de Hormigas para reducir el número de iteraciones en un espacio de búsqueda según el algoritmo K2 y el uso de operadores para moverse entre clases de equivalencia de acuerdo con el algoritmo ACO-B [9].

Referencias

1. Cooper, G. F., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Machine Learning, Vol. 9, pp. 309–348 (1992)
2. Spirtes, P., Meek, C.: Learning Bayesian networks with discrete variables from data. In: Proc. of the First International Conference on Knowledge Discovery and Data Mining, pp. 294–300 (1995)
3. Chickering, D. M.: Search operators for learning equivalence classes of Bayesian network structures. Technical Report, R231, UCLA Cognitive Systems Laboratory (1995)
4. de Campos, L. M., Fernández L., J. M. , Gámez, J. A., Puerta, J. M.: Ant colony optimization for learning Bayesian networks. International Journal of Approximate Reasoning, Vol. 31, pp. 291–311 (2002)

5. Buntine, W.: Theory refinement on Bayesian networks. In: Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, San Mateo, pp. 52–60 (1991)
6. Cooper, G., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Machine Learning, Vol. 9, No. 4, pp. 309-348 (1992)
7. M. Dorigo, T. Stützle: Ant colony optimization, The MIT Press (2004)
8. Dorigo, M., Maniezzo, V., Colorni, A.: The Ant System: Optimization by a Colony of Cooperating Agents. IEEE Trans. on Systems, Man and Cybernetics, Part B, Vol. 26, pp. 29-41 (1996)
9. Puerta C., José M.: Métodos locales y distribuidos para la construcción de redes de creencia estáticas y dinámicas. Tesis de la Universidad de Granada, España (2001)