Chapter 44

CAUSAL DISCOVERY

Hong Yao, Cory J. Butz, and Howard J. Hamilton
Department of Computer Science, University of Regina
Regina, SK, S4S 0A2, Canada
{yao2hong, butz, hamilton} @cs.uregina.ca

Abstract Many algorithms have been proposed for learning a causal network from data. It has been shown, however, that learning all the conditional independencies in a probability distribution is a NP-hard problem. In this chapter, we present an alternative method for learning a causal network from data. Our approach is novel in that it learns functional dependencies in the sample distribution rather than probabilistic independencies. Our method is based on the fact that functional dependency logically implies probabilistic conditional independency. The effectiveness of the proposed approach is explicitly demonstrated using fifteen real-world datasets.

Keywords: Causal networks, functional dependency, conditional independency

1. Introduction

Causal networks (CNs) (Pearl, 1988) have been successfully established as a framework for uncertainty reasoning. A CN is a *directed acyclic graph* (DAG) together with a corresponding set of *conditional probability distributions* (CPDs). Each node in the DAG represents a variable of interest, while an edge can be interpreted as direct casual influence. CNs facilitate knowledge acquisition as the *conditional independencies* (CIs) (Wong *et al.*, 2000) encoded in the DAG indicate that the product of the CPDs is a joint probability distribution.

Numerous algorithms have been proposed for learning a CN from data (Neapolitan, 2003). Developing a method for learning a CN from data is tantamount to obtaining an effective graphical representation of the CIs holding in the data. It has been shown, however, that discovering all the CIs in a probability distribution is a NP-hard problem (Bouckaert, 1994). In addition, choosing an initial DAG is important for reducing the search space, as many learning algorithms use greedy search techniques.

In this chapter, we present a method, called FD2CN, for learning a CN from data using *functional dependencies* (FDs) (Maier, 1983). We have recently developed a method for learning FDs from data (Yao *et al.*, 2002). Learning FDs from data is useful, since it has been shown that FD logically implies CI (Butz *et al.*, 1999). We show how to combine the obtained FDs with the chain rule of probability to construct a DAG of a CN. Given a set of FDs obtained from data, an ordering of variables is obtained such that the Markov boundaries of some variables are determined. Representing joint probability distribution of variables in the resulting ordering by the chain rule, the Markov boundaries of the other variables are determined. A DAG of a CN is constructed by designating each Markov boundary of variable as the parent set of the variable. During this process, we take full advantage of known results in both CNs (Pearl, 1988) and relational databases (Maier, 1983). We demonstrate the effectiveness of our approach using fifteen real-world datasets. The DAG constructed in our approach can also be used as an initial DAG for previous approaches. The work here further illustrates the intrinsic relationship between CNs and relational databases (Wong *et al.*, 2000; Wong and Butz, 2001).

The remainder of this chapter is organized as follows. Background knowledge is given in Section 2. In Section 3, the theoretical foundation of our approach is provided. The algorithm to construct a CN is developed in Section 4. In Section 5, the experimental results are presented. Conclusions are drawn in Section 6.

2. Background Knowledge

Let U be a finite set of discrete variables, each with a finite domain. Let V be the Cartesian product of the variable domains. A *joint probability distribution* (Pearl, 1988) $p(U)$ is a function p on V such that $0 \leq p(v) \leq 1$ for each configuration $v \in V$ and $\sum_{v \in V} p(v) = 1.0$. The *marginal distribution* $p(X)$ for $X \subseteq U$ is defined as $\sum_{U-X} p(U)$. If $p(X) > 0$, then the *conditional probability distribution* $p(Y|X)$ for $X, Y \subseteq U$ is defined as $p(XY)/p(X)$. In this chapter, we may write a_i for the singleton set $\{a_i\}$, and we use the terms attribute and variable interchangeably. Similarly, for the terms tuple and configuration.

DEFINITION 44.1 A *causal network* (CN) is a *directed acyclic graph* (DAG) $\mathcal{D}$ together with a *conditional probability distribution* (CPD) $p(a_i|P_i)$ for each variable a_i in $\mathcal{D}$, where P_i denotes the parent set of a_i in $\mathcal{D}$.

The DAG $\mathcal{D}$ graphically encodes CIs regarding the variables in U.

DEFINITION 44.2 *(Wong et al., 2000).* Let X, Y, and Z be three disjoint sets of variables. X is said to be *conditionally independent* of Y given Z, denoted $I(X, Z, Y)$, if $p(X|Y, Z) = p(X|Z)$.

As previously mentioned, the CIs encoded in the DAG $\mathcal{D}$ indicate that the product of the given CPDs is a joint probability distribution $p(U)$.

EXAMPLE 44.3 One CN on the set $U = \{a_1, a_2, a_3, a_4, a_5, a_6\}$ is the DAG in Figure 44.1(i) together with the CPDs $p(a_1)$, $p(a_2|a_1)$, $p(a_3|a_1)$, $p(a_4|a_2)$, $p(a_5|a_3)$, and $p(a_6|a_4, a_5)$. This DAG encodes, in particular, $I(a_3, a_1, a_2)$, $I(a_4, a_2, a_1a_3)$, $I(a_5, a_3, a_1a_2a_4)$ and $I(a_6, a_4a_5, a_1a_2a_3)$. By the chain rule, the joint probability distribution $p(U)$ can be expressed as:

$$p(U) = p(a_1)p(a_2|a_1)p(a_3|a_1, a_2)p(a_4|a_1, a_2, a_3)p(a_5|a_1, a_2, a_3, a_4) p(a_6|a_1, a_2, a_3, a_4, a_5).$$

The above CIs can be used to rewrite $p(U)$ as:

$$p(U) = p(a_1)p(a_2|a_1)p(a_3|a_1)p(a_4|a_2)p(a_5|a_3)p(a_6|a_4, a_5).$$

The term CN is somewhat misleading because it may be possible to reverse a directed edge without disturbing the encoded CI information. For example, the two CNs in Figure 44.1 encode the same independency information. Thus, it is perhaps better to view a CN as encoding independency information rather than causal relationships.

Using CI information encoded in the DAG, the Markov boundary of a variable can be defined.

DEFINITION 44.4 Let $U = \{a_1, \ldots, a_n\}$, and $\mathcal{O}$ be an ordering $\langle a_1, \ldots, a_n \rangle$ of variables of U. Let $U_i = \{a_1, \ldots, a_{i-1}\}$ be a subset of U with respect to the ordering $\mathcal{O}$. A *Markov boundary* of a variable a_i over U_i, denoted B_i, is any subset X of U_i such that $p(U_i)$ satisfies $I(a_i, X, U_i - X - a_i)$ and $a_i \notin X$, but $p(U_i)$ does not satisfy $I(a_i, X', U_i - X' - a_i)$ for any $X' \subset X$.

EXAMPLE 44.5 Recall the DAG in Figure 44.1(i). Let the ordering $\mathcal{O}$ be $\langle a_1, a_2, a_3, a_4, a_5, a_6 \rangle$. Then $U_1 = \{\}$, $U_2 = \{a_1\}$, $U_3 = \{a_1, a_2\}$, $U_4 = \{a_1, a_2, a_3\}$, $U_5 = \{a_1, a_2, a_3, a_4\}$, and $U_6 = \{a_1, a_2, a_3, a_4, a_5\}$. The Markov boundary of variable a_3 over U_3 is $B_3 = \{a_1\}$, since $p(U_3)$ satisfies $I(a_3, a_1, a_2)$, but does not satisfy $I(a_3, \phi, a_1a_2)$.

The Markov boundary B_i of a variable a_i over U_i encodes CI $I(a_i, B_i, U_i - B_i - a_i)$ over $p(U_i)$. Using the Markov boundary of each variable, a boundary DAG is defined as follows.

DEFINITION 44.6 Let $p(U)$ be a joint probability distribution over U, $\mathcal{O}$ be an ordering $\langle a_1, \ldots, a_n \rangle$ of variables of U, and $U_i = \{a_1, \ldots, a_{i-1}\}$ be a

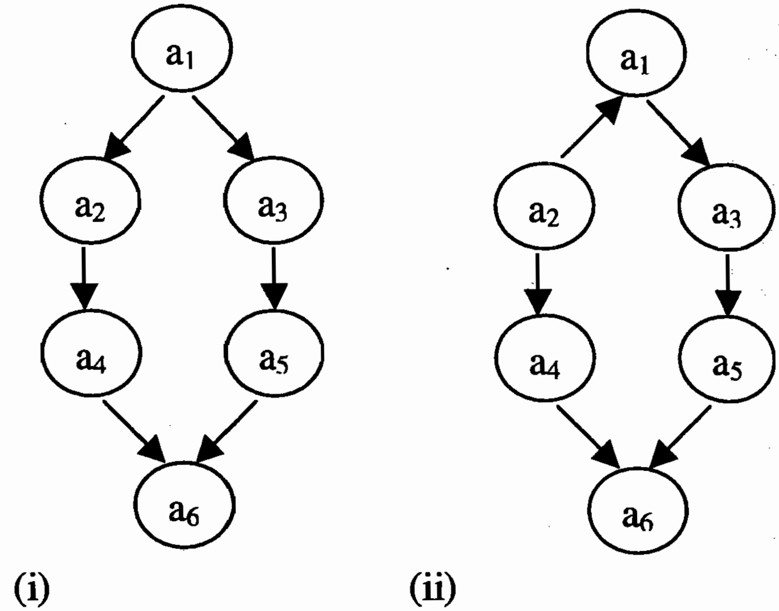

Figure 44.1. Two causal networks.

subset of U with respect to the ordering $\mathcal{O}$. $\{B_1, \ldots, B_n\}$ is an ordered set of subsets of U such that each B_i is a Markov boundary of a_i over the U_i. The DAG created by designating each B_i as a parent set of variable a_i is called a *boundary DAG* of $p(U)$ relative to $\mathcal{O}$.

The next theorem (Pearl, 1988) indicates that the boundary DAG of $p(U)$ relative to an ordering $\mathcal{O}$ is a DAG of CN of $p(U)$.

THEOREM 44.7 Let $p(U)$ be a joint probability distribution and $\mathcal{O}$ be an ordering of variables of U. If $\mathcal{D}$ is a boundary DAG of $p(U)$ relative to $\mathcal{O}$, then $\mathcal{D}$ is a DAG of CN of $p(U)$.

It is important to realize that we can construct a DAG of a CN after the Markov boundary of each variable has been obtained according to Definition 44.6 and Theorem 44.7.

EXAMPLE 44.8 Let $U = \{a_1, \ldots, a_6\}$ and $\mathcal{O} = \langle a_2, a_1, a_3, a_4, a_5, a_6 \rangle$ be an ordering of the variables of U. With respect to the ordering $\mathcal{O}$, $U_1 = \{a_2\}$,

$U_2 = \{\}$, $U_3 = \{a_1, a_2\}$, $U_4 = \{a_1, a_2, a_3\}$, $U_5 = \{a_1, a_2, a_3, a_4\}$, and $U_6 = \{a_1, a_2, a_3, a_4, a_5\}$. Supposing we assign $B_1 = \{a_2\}$, $B_2 = \{\}$, $B_3 = \{a_1\}$, $B_4 = \{a_2\}$, $B_5 = \{a_3\}$ and $B_6 = \{a_4, a_5\}$, then the DAG shown in Figure 44.1 (ii) is the learned DAG of CN of $p(U)$.

3. Theoretical Foundation

In this section, several theorems relevant to our approach are provided. We define a relation $r(U)$ as a finite set of tuples over U. We begin with functional dependency (Maier, 1983).

DEFINITION 44.9 Let $r(U)$ be a relation over U and $X, Y \subseteq U$. The *functional dependency* (FD) $X \rightarrow Y$ is satisfied by $r(U)$ if every two tuples t_1 and t_2 of $r(U)$ that agree on X also agree on Y.

If a relation $r(U)$ satisfies the FD $X \rightarrow Y$, but not $X' \rightarrow Y$ for every $X' \subset X$, then $X \rightarrow Y$ is called *left-reduced* (Maier, 1983).

The next theorem shows that FD logically implies CI.

THEOREM 44.10 (Butz *et al.*, 1999). Let $r(U)$ be a relation over U. Let $p(U)$ be a joint distribution over $r(U)$. Let $X, Y \subseteq U$ and $Z = U - XY$. Having FD $X \rightarrow Y$ satisfied by $r(U)$ is a sufficient condition for the CI $I(Y, X, Z)$ to be satisfied by $p(U)$.

By exploiting the implication relationship between functional dependency and conditional independency, we can relate a left-reduced FD $X \rightarrow a_i$ to the Markov boundary of variable a_i.

THEOREM 44.11 Let $U = \{a_1, \ldots, a_n\}$, U_i be a subset of U, and $X \subseteq U_i$. If FD $X \rightarrow a_i$ is left-reduced, then X is the Markov boundary of variable a_i over U_i.

Proof: Since $X \rightarrow a_i$ is a FD and $X \subseteq U_i$, according to Definition 44.9, $X \rightarrow a_i$ holds over $U_i \cup \{a_i\}$. Since FD $X \rightarrow a_i$ is left-reduced, by Theorem 44.10, $p(U_i)$ satisfies $I(a_i, X, U_i - X - a_i)$ but not $I(a_i, X', U_i - X' - a_i)$ for any $X' \subset X$.

Theorem 44.11 indicates that the Markov boundary of variable a_i over U_i can be learned from a left-reduced FD $X \rightarrow a_i$, if X is a subset of U_i. We define those variables that can be learned from a set of left-reduced FDs as follows.

DEFINITION 44.12 Let $U = \{a_1, \ldots, a_n\}$, $X \subset U$, $a_i \notin X$, and F be a set of left-reduced FDs over U. If exists $X \rightarrow a_i \in F$, then a_i is a *decided* variable. Otherwise, a_i is an *undecided* variable.

Considering a variable as decided indicates that its Markov boundary can be learned from FDs implied in data.

4. Learning a DAG of CN by FDs

In this section, we use learned FDs to construct a CN. We illustrate our algorithm using the heart disease dataset, which contains 13 attributes and 230 rows, from the UCI Machine Learning Repository (Blake and Merz, 1998).

EXAMPLE 44.13 The heart disease dataset has $U = \{a_1, \ldots, a_{13}\}$. Using FD_Mine, the discovered set of left-reduced FDs is $F = \{a_1a_5 \to a_3, a_1a_5 \to a_6, a_1a_5 \to a_{11}, a_1a_5 \to a_{13}, a_1a_8 \to a_7, a_4a_5a_9 \to a_2, a_1a_5a_{10} \to a_4, a_1a_2a_5 \to a_8, a_1a_5a_{10} \to a_9, a_1a_5a_{10} \to a_{12}\}$.

As indicated by Theorem 44.7, a DAG of a CN is constructed when the Markov boundary of each variable relative to an ordering $\mathcal{O}$ is obtained. We obtain the Markov boundary of each variable in two steps. First, in Section 4.1, we show how to obtain an ordering $\mathcal{O}$ such that the Markov boundary of each decided variable with respect to $\mathcal{O}$ can be obtained from the given FDs. Second, we determine the Markov boundary of each undecided variable by the chain rule in Section 4.2.

4.1 Learning an Ordering of Variables from FDs

Given a set F of left-reduced FDs, the algorithm in Figure 44.2 will determine an ordering $\mathcal{O}$ of the variables of U such that the Markov boundary of each decided variable with respect to $\mathcal{O}$ can be obtained from F.

We use the FDs in Example 44.13 to demonstrate how Algorithm 1 works.

EXAMPLE 44.14 First, the FD $a_1a_5 \to a_3$ is selected in line 2 by Algorithm 1. In line 3, since a_3 is not in Y for any $Y \to a_i$ $(a_i \neq a_3)$ in F, variable a_3 is removed for U in line 4, thus $U = \{a_1, a_2, a_4, \ldots, a_{13}\}$. We obtain $\mathcal{O} = \langle a_3 \rangle$ in line 5. In line 6, FD $a_1a_5 \to a_3$ is removed from F. Because F is not empty, Algorithm 1 continues performing lines 2 – 8. At this time, the FD $a_1a_5 \to a_6$ is selected in line 2. Variable a_6 is removed from U in line 4, $U = \{a_1, a_2, a_4, a_5, a_7, \ldots, a_{13}\}$. We obtain $\mathcal{O} = \langle a_6, a_3 \rangle$ in line 5. By recursively performing lines 2 to 8 until F is empty, we obtain an ordering $\mathcal{O} = \langle a_{12}, a_9, a_4, a_2, a_8, a_7, a_{13}, a_{11}, a_6, a_3 \rangle$ and $U = \{a_1, a_5, a_{10}\}$. In line 9, we prepend U to the head of $\mathcal{O}$. Thus, for the variables in U, $\mathcal{O} = \langle a_1, a_5, a_{10}, a_{12}, a_9, a_4, a_2, a_8, a_7, a_{13}, a_{11}, a_6, a_3 \rangle$.

The next theorem guarantees that the Markov boundary of each decided variable is determined by $\mathcal{O}$ as obtained by Algorithm 1.

THEOREM 44.15 Let $U = \{a_1, \ldots, a_n\}$, $\mathcal{O} = \langle a_1, \ldots, a_n \rangle$ be an ordering obtained by the Algorithm 1, and $U_i = \{a_1, \ldots, a_{i-1}\}$ be a subset of U with

Algorithm 1.
Input: $U = \{a_1, \ldots, a_n\}$, and a set F of left-reduced FDs.
Output: an ordered list $\mathcal{O}$ of the variables of U.
Begin
1. $\mathcal{O} = \langle\rangle$.
2. for each $X \rightarrow a_i \in F$
3. if $a_i \notin Y$ for all $Y \rightarrow y \in F$ $(y \neq a_i)$
4. $U = U - \{a_i\}$.
5. prepend a_i to the head of $\mathcal{O}$.
6. $F = F - \{Y \rightarrow a_i | Y \rightarrow a_i \in F\}$.
7. end if
8. end for
9. prepend U to the head of $\mathcal{O}$.
10. return($\mathcal{O}$)
End

Figure 44.2. An algorithm to obtain an ordering $\mathcal{O}$ of the variables of U.

respect to $\mathcal{O}$. Let B_i be the Markov boundary of a_i over U_i. If a_i is a decided variable, then $B_i \subseteq U_i$.

Proof: Since a_i is a decided variable, then there exists a FD $X \rightarrow a_i$. When Algorithm 1 performs lines 2-8, then a_i is always deleted from U before any variable of X according to lines 3-5 of Algorithm 1. Thus, for any $a_j \in X$, a_j is always before a_i in $\mathcal{O}$, so we have $X \subseteq U_i$. Since $B_i = X$, according to Theorem 44.11, then $B_i \subseteq U_i$.

4.2 Learning the Markov Boundaries of Undecided Variables

Once an ordering $\mathcal{O} = \langle a_1, \ldots, a_n \rangle$ of variables of U is obtained by Algorithm 1, the joint probability distribution $p(U)$ can be expressed using the chain rule as follows.

$$p(U) = p(a_1) \ldots p(a_j | a_1, \ldots, a_{j-1}) \ldots p(a_n | a_1, \ldots, a_{n-1}). \qquad (44.1)$$

If a_i is an undecided variable, then there is no FD $X \rightarrow a_i \in F$. According to Algorithm 1, a_i is not deleted from U and is prepended to the head of $\mathcal{O}$ at the line 9 of Algorithm 1. This indicates that all undecided variables appear before all decided variables in $\mathcal{O}$.

Suppose $a_1, \ldots, a_j$ are all the undecided variables, and $B_{j+1}, \ldots, B_n$ are the Markov boundaries of all the decided variables. By Definition 44.4, CI $I(a_i, B_i, U_i - B_i - a_i)$ holds for each variable $a_i, j+1 \leq i \leq n$. Thus, each

$p(a_i|a_1, \ldots, a_{i-1}) = p(a_i|B_i)$. Equation 61.1 can be rewritten as:

$$p(U) = p(a_1) \ldots p(a_j|a_1, \ldots, a_{j-1}) p(a_{j+1}|B_{j+1}) \ldots p(a_n|B_n). \quad (44.2)$$

By assigning $B_k = \{a_1, \ldots, a_{k-1}\}$ as the Markov boundary of each undecided variable $a_k, 1 \leq k \leq j$, Equation 61.2 can be expressed as:

$$p(U) = p(a_1|B_1) \ldots p(a_j|B_j) \ldots p(a_n|B_n) = \prod_{a_i \in U} p(a_i|B_i). \quad (44.3)$$

Equation 61.5 indicated that a joint probability distribution $p(U)$ can be represented by the Markov boundary of all variables of U. Thus, a boundary DAG relative to $\mathcal{D}$ can be constructed. Based on the above analysis, we developed the algorithm shown in Figure 44.3, called FD2CN, which learns a DAG $\mathcal{D}$ of a CN from a dataset $r(U)$.

Algorithm 2. FD2CN
Input: A dataset $r(U)$ over variable set U.
Output: A DAG $\mathcal{D}\langle U, E\rangle$ of a CN learned from $r(U)$.
Begin
1. F= FD_Mine($r(U)$). //return a set of left-reduced FDs
2. Obtaining an ordering $\mathcal{O}$ using Algorithm 1.
3. $U_i = \{\}$.
4. while $\mathcal{O}$ is not empty
5. $\quad a_i$= pophead($\mathcal{O}$).
6. $\quad$ if there exists a FD $X \rightarrow a_i \in F$ then
7. $\qquad B_i = X$
8. $\quad$ else
9. $\qquad B_i = U_i$.
10. $\quad U_i = U_i \cup \{a_i\}$.
11. end while
12. Constructing a DAG $\mathcal{D}\langle U, E\rangle$ such that $\{(b, a_i) \in E | b \in B_i,\ a_i, b \in U\}$.
End

Figure 44.3. The algorithm, FD2CN, to learn a DAG of a CN from data.

In line 6 of Algorithm FD2BN, we determined whether or not a variable is decided. If so, in the line 7, we obtain its Markov boundary. If not, in line 9, we obtain its Markov boundary. In the line 12, a DAG of a CN is constructed by making each B_i as the parent set of a_i of U. In other words, if $b_i \in B_i$, then we add a edge $b \rightarrow a_i$ in DAG $\mathcal{D}$. According to Definition 44.6 and Theorem 44.7, we know the constructed DAG $\mathcal{D}$ is a DAG of a CN.

EXAMPLE 44.16 Applying Algorithm FD2CN on the heart disease dataset, F in line 1 is obtained as in Example 44.13. $\mathcal{O}$ in line 2 is obtained as in Example 44.14. According to obtained $\mathcal{O}$, we obtain $B_1 = \{\}$, $B_5 = \{a_1\}$, $B_{10} =$

$\{a_1, a_5\}$, $B_{12} = \{a_1, a_5, a_{10}\}$, $B_9 = \{a_1, a_5, a_{10}\}$, $B_4 = \{a_1, a_5, a_{10}\}$, $B_2 = \{a_4, a_5, a_9\}$, $B_8 = \{a_1, a_5, a_2\}$, $B_7 = \{a_1, a_8\}$, $B_{13} = \{a_1, a_5\}$, $B_{11} = \{a_1, a_5\}$, $B_6 = \{a_1, a_5\}$, and $B_3 = \{a_1, a_5\}$. By making each B_i as the parent set of a_i of U, a $\mathcal{D}$ is constructed. For example, since $B_2 = \{a_4, a_5, a_9\}$, then $\mathcal{D}$ have edges $a_4 \rightarrow a_2$, $a_5 \rightarrow a_2$, and $a_9 \rightarrow a_2$. the DAG of a CN learned from the heart disease dataset in Figure 44.5 is depicted in Figure 44.4.

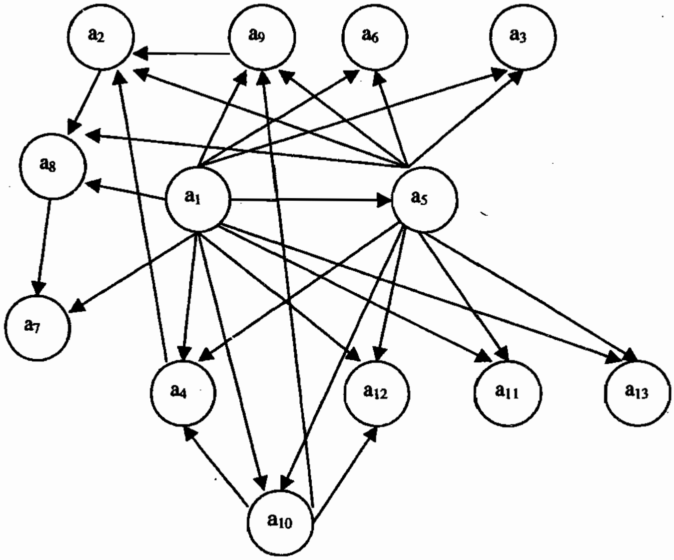

Figure 44.4. The learned DAG of a CN from the heart disease dataset.

5. Experimental Results

Experiments were carried out on fifteen real-world datasets obtained from the UCI Machine Learning Repositories (Blake and Merz, 1998). The results are shown in Figure 44.5. The last column gives the elapsed time to construct a CN, measured on a 1GHz Pentium III PC with 256 MB RAM. The results show that the processing time is mainly determined by the number of attributes. Since it also indicates that many FDs hold in some datasets, our proposed approach is a feasible way to learn a CN.

6. Conclusion

In this chapter, we presented a novel method for learning a CN. Although a CN encodes probabilistic conditional independencies, our method is based on

Dataset Name	# of attributes	# of rows	# of FDs	Time (seconds)
Abalone	8	4,177	60	1
Breast-cancer	10	191	3	0
Bridge	13	108	62	2
Cancer-Wisconsin	10	699	19	1
Chess	7	28,056	1	3
Crx	16	690	1,099	10
Echocardiogram	13	132	583	0
Glass	10	142	119	0
Heart disease	13	270	10	2
Hepatitis	20	155	8,250	1,327
Imports-85	26	205	4,176	8,322
Iris	5	150	4	0
Led	8	50	11	0
Nursery	9	12,960	1	16
Pendigits	17	7,494	29,934	920

Figure 44.5. Experimental results using fifteen real-world datasets.

learning FDs (Yao *et al.*, 2002). Since functional dependency logically implies conditional independency (Butz *et al.*, 1999), we described how to construct a CN from data dependencies. We implemented our approach and encouraging experimental results have been obtained.

Since functional dependency is a special case of conditional independency, we acknowledge that our approach may not utilize all the independency information encoded in the sample data. However, previous methods also suffer from this disadvantage as learning all CIs from sample data is a NP-hard problem (Bouckaert, 1994).

References

Bouckaert, R. (1994). Properties of learning algorithms for Bayesian belief networks. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, 102–109.

Butz, C. J., Wong, S. K. M., and Yao, Y. Y. (1999) On Data and Probabilistic Dependencies, *IEEE Canadian Conference on Electrical and Computer Engineering*, 1692-1697.

Maier, D. (1983). *The Theory of Relational Databases*, Computer Science Press.

Neapolitan, R. E. (2003) *Learning Bayesian Networks*, Prentice Hall.

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

Blake, C.L. and Merz, C.J. (1998). UCI Repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science.

Wong, S. K. M., Butz, C. J., and Wu, D. (2000). On the Implication Problem for Probabilistic Conditional Independency, *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 30(6), 785-805.

Wong, S. K. M., and Butz, C. J. (2001), Constructing the Dependency Structure of a Multi-Agent Probabilistic Network, *IEEE Transactions on Knowledge and Data Engineering*, 13(3), 395-415.

Yao, H., Hamilton, H. J., and Butz, C. J. (2002). FD_Mine: Discovering Functional Dependencies in a Database Using Equivalences, In *Proceedings of the Second IEEE International Conference on Data Mining*, 729-732.

DATA MINING AND KNOWLEDGE DISCOVERY HANDBOOK

edited by

Oded Maimon and Lior Rokach
Tel-Aviv University, Israel

Springer