한국 퍼지 및 지능시스템학회 논문지 2000, Vol. 10, No. 3 (2000)232~241

최적 규칙 발견 시스템의 구현: 개념 계층과 정보 이득 및 라프셋에 의한 통합 접근

An Implementation of Optimal Rules Discovery System: An Integrated Approach Based on Concept Hierarchies, Information Gain, and Rough Sets

정 홍 · 김진상

Hong Chung and Jin-Sang Kim

계명대학교 컴퓨터전자공학부

요 약

본 연구는 대량의 데이터에서 효율적으로 최적 규칙을 발견하기 위해 개념 계층과 정보 이득 및 라프셋 이론에 기반한 통합 방법을 제시하고, 이를 최적 규칙 발견 시스템으로 구현한다. 본 방법은 데이터베이스에 있는 데이터에서 일반화된 지식을 추출하기 위한 속성중심의 개념 상승 기법과 불필요한 속성 및 속성값을 제거하기 위한 지식 감축 기법을 적용하며, 최적 규칙의 도출을 위해 속성의 중요도를 사용한다. 본 시스템은 먼저, 속성값 개념의 일반화에 의해 중복 튜플을 제거함으로써 데이터베이스의 크기를 줄이고, 결정속성에 영향을 주지않는 조건속성을 제거함으로써 속성의 수를 줄이며, 마지막으로 속성간의 종속관계를 분석하고 불필요한 속성값을 제거하여 간략화된 최적 규칙을 유도한다. 그리고 실제 데이터에 적용하여 결정 규칙을 유도하고 그 규칙을 새로운 데이터에 테스트해 봄으로써 새로운 데이터에도 잘 적용됨을 보인다.

ABSTRACT

This study suggests an integrated method based on concept hierarchies, information gain, and rough set theory for efficient discovery of optimal rules from a large amount of data, and implements an optimal rules discovery system. Our approach applies attribute-oriented concept ascension technique to extract generalized knowledge from a database, knowledge reduction technique to remove superfluous attributes and attribute values, and significance of attributes to induce optimal rules. The system first reduces the size of database by removing the duplicate tuples through the concept generalization of attribute values, reduces the number of attributes by means of eliminating the condition attributes which have no influences on the decision attributes, and finally induces simplified optimal rules by removing the superfluous attribute values by analyzing the dependency relationships among the attributes. And we induce some decision rules from actual data by using the system and test the rules to new data, and evaluate that the rules are well suited to them.

1. 서 론

지식 발견은 데이터베이스나 정보 저장소에 있는 대량의 데이터에서 유용한 지식을 찾고자 하는 요구에 따라 많은 관심을 받고 있으며, 최근 대규모 데이터베이스에서 지식을 발견하기 위한 연구 및 개발 활동이 활발하게 이루어지고 있다[4].

지식발견 방법에 있어서 지식 감축(knowledge reduction)[11], 속성중심(attribute-oriented) 접근[6], 결정 트리(decision trees)에 의한 귀납[12] 등 상당한 연구가 진전되고 있으며, 또한 발견된 지식에 대한 추론방법의 개발이 진행되고 있다. 속성중심 접근을 바탕으로 개발된 지식 발견 시스템에는 1993년 Simon Fraser 대학에서 개발한 DBLearn[6]이 있다. 이 시스템은 속성별로 업무에 적합한 데이터의 부분집합을 만들어 일반화 관계로 축약하고, 여기서 데이터의 일반 특성을 추출하는 속성중심 귀납법을 적용하고 있다. 또 결정 트리의 귀납에 의한 기계학습 시스템으로서 ID3[15]와 이를 점진적 학습으로 확장한 ID5[15]가 있는데, 이는 어떤 개념에 대한 사례가 주어졌을 때 이로부터 개념을 구별할 수 있는 의사결정 트리 형태의 분류규칙을 생성시킨다. 최근에는 라프셋(rough sets) 이론을 적용한 지식발견 연구가 진행되고 있는데, 이는 데이터베이스에서 불필요한 속성을 제거하여

본 연구는 1998년도 계명대학교 교무처 연구지원팀에서 지급한 부설연구소 연구비로 이루어졌음

간략한 지식을 유도하는 것으로, 구현된 시스템에는 DataLogic, Rosetta[13] 등이 있다.

데이터베이스에서의 지식발견은 데이터베이스로부터 관심있는 지식을 발견하고 고수준의 언어로 지식을 표현하는 학습 형태로서, 상기와 같은 여러가지 기법들이 있으나, 단 하나의 기법의 적용으로는 각기 한계성 때문에 유용한 지식의 발견이 불충분하므로 이들의 특징을 잘 통합하고 발전시킨 새로운 기법이 필요하다[7].

본 연구에서는 데이터베이스의 일반화를 위한 개념 계층 방법과 라프셋에 의한 불필요 속성의 감축 방법을 통합 적용함으로써 유용하고 간략한 최적 규칙을 자동적으로 생성하는 효율적 지식발견 방법을 설계하고 구현한다. 이 방법의 첫 단계는 속성중심의 개념 상승에 의해 데이터베이스의 일반화를 도모하고, 두번째 단계는 속성 감축(attribute reduction) 기법을 적용하여 불필요 속성과 속성값을 제거하는데, 최적 감축을 위한 속성의 중요도 결정에는 결정 트리 방법에서의 정보 이득(information gain) 측정 기법을 이용한다. 이를 위해 개념 상승에 의한 데이터베이스의 일반화, 정보 이득 측정에 의한 속성의 중요도 계산, 중요도를 이용한 최적 감축, 속성값의 효율적인 감축 방법을 연구하고 각각의 알고리즘을 설계하여 시스템으로 구현한다.

2. 개념 계층과 개념 상승

2.1 개념 계층

개념 계층은 데이터베이스의 속성에 있어서 일반화 관계의 집합이다. 일반화 관계는 속성값의 전체집합과 이를 일반화한 단일값간의 관계이다. 즉, 속성 a의 일반화 관계는 a의 정의역이 $\{A_1, A_2, \cdots A_k\}$이고 개념으로 표현된 단일값이 B일 때, $\{A_1, A_2, \cdots A_k\} \subset B$로 표현되며, 이때 B는 각 $A_i(1 \le i \le k)$의 일반화이다. 개념 계층은 트리(tree) 형태로 표현하는데, 이를 개념 트리(concept trees)라 한다.

개념 트리는 자동적으로 또는 반자동적으로 구성할 수 있는데[6], 수치 속성은 클러스터링 방법 혹은 통계적 방법에 의하여 이산적 개념 트리로 자동 조직화될 수 있고, 비수치 속성은 속성간 유사도나 거리 등 상관 관계[3]에 의하여 구성될 수 있다. 본 연구에서 수치 속성에 대해서는 완전 자동화가 가능한 클러스터링 방법을 사용하고, 비수치 속성에서는 실용적이고 간단한 전문가 지식을 이용하고자 한다.

수치 속성은 Fisher가 제안한 개념적 클러스터링 시스템인 COBWEB[5]에 의하여 자동으로 조직화할 수 있는데, 이는 속성집합으로 기술된 객체를 분류 트리로 구성하는데 있어서 CU(Category Utility)라는 품질 척도(quality measure)를 사용한다. 즉, 클러스터 C를 n개의 상호배타적 클래스 $C_1, \cdots, C_n$으로 분할하는데 있어서 CU는 분할 후 클래스내의 유사성(intra-class similarity) 및 클래스간의 상이성(inter-class dissimilarity)을 의미하는 적합도(goodness)의 증가로 정의한다. 그러나 이 방법은 분류하는데 많은 메모리와 시간을 소요하므로 범주 데이터에만 적용되고 연속적 수치 데이터에는 사용하기 어렵다[3]. 본 연구에서는 Chu 등이 개발한 CoBase[3]에서 지식베이스를 구축하기 위한 TAH(Type Abstraction Hierarchies)의 생성에 CU를 근사적으로 계산하는 방법을 속성 단위 및 이진 분할 단위로 간략화하여 개념 트리의 자동 생성에 사용한다. TAH에서는 클러스터링의 척도로서 RE(Relaxation Error)를 사용하는데, 이는 클러스터 C가 x_i의 집합으로 구성되어 있을 때 실제 속성값과 일반화한 값간의 평균 차이로 정의한다.

$$\text{속성값 } x_i\text{의 } RE(x_i) = \sum_{j=1}^{n} P(x_j) \mid x_i - x_j \mid \quad (1)$$

$P(x_j)$: C에서 속성값 x_j의 발생 확률

$RE(x_i)$를 C의 모든 속성값 x_i에 대하여 합하면 다음과 같다.

$$C \text{ 전체의 } RE(C) = \sum_{i=1}^{n} P(x_i) RE(x_i) \quad (2)$$

C의 분할 $P = \{C_1, \cdots C_n\}$에서 분할 P의 RE는 다음과 같이 정의한다.

$$RE(P) = \sum_{k=1}^{n} P(C_k) RE(C_k) \quad (3)$$

$P(C_k)$: C_k의 속성값 수를 C의 속성값수로 나눈 값

일반적으로 $RE(P) < RE(C)$인데, 이는 분할함으로써 RE가 감소함을 의미하므로, 최적 분할은 가장 적은 값을 갖는 $RE(P)$를 갖도록 분할한다.

그런데 하나의 클러스터를 n개의 서브클러스터로 분할하는 조합의 수는 n에 지수적이므로 최적분할 계산은 지수적 시간복잡도를 가진다. 따라서 본 연구에서는 계산시간을 줄이기 위해 이진분할을 먼저 하고, 이진분할중 큰 서브클러스터를 또 이진분할하는 방법을 사용한다. 즉, 이진분할에서 시작하여 가장 큰 RE를 가지는 서브클러스터를 찾아 사용자가 원하는 t개의 서브클러스터가 생성될 때까지 반복 이진분할 한다.

분할방법은 알고리즘-1, 알고리즘-2와 같다.

알고리즘-1 Partition(C, T)

/* input cluster $C = \{x_1, \cdots, x_n\}$, T: 분할수 */

```
begin
  if n < T then return C;
  /* n: C에 있는 유일한(distinct) 값의 수 */
  for i = 1 to T − 1 do
    C_i = subcluster which has maximum RE in C;
    cut = BinaryCut(C_i);
    create subcluster C_i1, C_i2 in C_i using cut;
  end for
  return C_1, C_2, ···, C_T;
  end
```

알고리즘-2 BinaryCut(*C*)

```
begin
  MinRE = MaxInteger;
  for h = 1 to n − 1 do
    partition C into subcluster C_1={x_1, ···, x_h} and
    C_2={x_h+1, ··· x_n};
    calculate RE(P);   /* 분할의 RE */
    if RE(P) < MinRE then
      MinRE = RE(P);  /* minimum RE */
      cut = h; /* optimal partition */
    end if
  end for
  return cut;
end
```

상기 알고리즘-1은 입력 클러스터 C를 사용자가 원하는 분할수 만큼 이진분할 알고리즘-2를 호출한다. 따라서 시간 복잡도는 $O(nt)$이다.

2.2 개념 상승

데이터베이스의 일반화인 개념 상승은 각 튜플의 속성값을 관련 속성의 개념 트리에서 상위수준의 개념으로 대치시킴으로써 수행된다[6]. 일반화시키고자 하는 수준은 응용별 개념 계층에 따라 다르다. 개념 계층의 상승은 데이터베이스가 일반화된 고수준의 개념을 가지며, 이때 중복되는 튜플은 합병하여 튜플 수를 줄인다.

개념이 상승된 일반화 데이터베이스에서 규칙을 도출할 때 조건속성은 동일한데 결정속성이 상이한 모순된 결정규칙이 생성되는 현상 즉, 결정속성에 대한 조건속성의 충돌이 발생할 수 있다. 이를 해결하기 위한 방법은 첫째, 충돌이 발생한 튜플을 모두 제거하는 것인데, 이는 정보의 손실에 의하여 일부 규칙만 생성된다. 둘째, 확률이 적은 튜플을 제거하는 것인데, 이는 유도된 규칙이 한쪽으로 편향되어 신뢰성이 결여된 규칙이 생성된다. 본 연구에서는 이를 모두 수용하는 방법을 사용한다. 즉, 모순된 두개 이상의 규칙을 두개 이상의 결정속성 값을 가지는 하나의 규칙으로 처리하여 각각의 결정속성 값에 확률을 부여한다.

개념 상승시 고려해야 할 또다른 문제는 빈도가 매우 적은 튜플의 처리인데, 이를 일반화 규칙으로 유도했을 때 규칙의 신뢰도가 매우 낮을 가능성이 있다. 따라서, 개념상승 관계에서 거의 나타나지 않는 튜플은 예외사항으로 간주하여 규칙의 일반화 이전에 사용자가 정한 임계치보다 작을 때 제거한다. 즉, 필터 임계치는 일반화 관계에 있는 매우 작은 빈도의 튜플을 걸러내는 작은 값의 백분율이다.

일반화 임계치는 최종관계가 가질 수 있는 개념의 수준으로서, 작은 임계치는 많은 수의 속성값을 가지므로 복잡한 규칙이 유도되어 일반화가 미비하다. 반대로 큰 임계치는 적은 수의 속성값을 가지므로 간단한 규칙이 유도되나 일반화가 과하게 되어 중요한 정보가 손실될 수 있다. 이는 사용자가 응용에 따라 적합한 임계치를 정하도록 한다.

개념 상승 알고리즘은 알고리즘-3과 같다.

알고리즘-3 Concept_Ascension

```
input: VLDB, HT, T, F
  /* VLDB: 속성 A_i(1≤i≤n)의 집합을 가진 DB,
     HT: 개념 트리,
     T: 속성 A_i의 상승수준 임계치,
     F: 필터 임계치 */
output: GDB  /* GDB: 일반화 DB */
begin
  for i = 1 to n  /* n: 속성 수 */
   if A_i∈HT then
   L = 0;
   repeat
   substitute value of HTi for value of tuple;
   L = L + 1;
   until Ti = L
   end if
   end for
  merge duplicate tuples and accumulate count;
  calculate frequency of each tuple;
  if frequency<F then
    eliminate the tuple;
   end if
   store GDB;
end
```

상기 알고리즘-3은 속성별로 상승수준 임계치 만큼 개념을 상승시키므로 속성수를 n, 임계치를 t라 할 때 시간복잡도는 $O(nt)$이다.

3. 속성의 중요도 결정

속성의 중요도를 계산하는 방법에는 통계학에서 사용하는 χ^2 적합도 검증, 결정 트리에 의한 기계학습에서 사용하는 정보 이득 측정[9], 라프셋 이론에서의 속성간 종속도 계산 방법 등이 있는데, 이중 정보 이득 측정 방법이 규칙의 발견에 있어서 우수하므로 [17], 본 연구에서는 이 방법을 사용한다.

사례집합 K가 가지고 있는 정보값은 다음과 같은 엔트로피(Entropy)로 나타낼 수 있다.

$$E(K) = \sum_{i=1}^{m} P_i \log_2(1/P_i) = -\sum_{i=1}^{m} P_i \log_2 P_i \quad (4)$$

여기서 P_i는 클래스 K_i가 사례집합 K에서 차지하는 비율이다.

속성 X_j가 $|X_j|$가지의 속성값을 가지고, 클래스 K_i가 속성 X_j를 사용하여 집합 K를 나누었을 경우 정보값 $E'(X_j)$는 다음과 같다.

$$E'(X_j) = \sum_{i=1}^{|X_j|} Wi * E(S_i) \quad (5)$$

여기서 $E(S_i)$는 X_j 속성의 i번째 클래스의 값을 가지는 경우 하위 사례집합 S_i의 정보값이고, W_i는 가중치로서 다음과 같다.

$$W_i = \frac{S_i\text{에서의 사례의 수}}{K\text{에서의 사례의 수}}$$

사례집합 K에서 속성 X_j로 분류한 결정 트리에 의해 획득한 정보값 gain(X_j)는 다음과 같다.

$$\text{gain}(X_j) = E(K) - E'(X_j) \quad (6)$$

그리고 속성별 정보 이득을 상대적으로 평가하기 위해 gain(X_j)를 정규화한 속성의 중요도 $S(X_j)$를 다음과 같이 정의한다.

$$S(X_j) = \text{gain}(X_j)/E(K) \quad (7)$$

여기서 $S(X_j)$는 0에서 1까지의 값을 가지게 되는데, 1에 가까울수록 속성의 중요도가 크며 결정속성에 영향을 많이 미친다고 볼 수 있다.

4. 라프셋과 속성 감축

4.1 라프셋

라프셋은 식별 불가능(indiscernible) 객체의 클래스로 구성된 동치관계를 기본으로 한다[11]. 본 연구에서는 일반 지식의 발견이 아닌 일반화 규칙의 발견을 다루므로 객체라는 용어 대신에 결정규칙이라는 용어로 사용한다.

표 1. 결정규칙 테이블

U	a	b	c	d	e
1	0	0	1	0	0
2	1	0	2	1	1
3	1	1	1	0	0
4	0	2	1	1	1
5	1	2	1	0	1
6	1	0	1	0	0
7	1	2	2	1	1
8	0	0	2	1	1

본 연구에서 결정규칙 시스템 S는 다음과 같이 정의한다.

$S = \{U, A, V\}$
$U = \{x_1, x_2, \cdots x_n\}$인 결정규칙의 유한집합
$A = C \cup D$로서, C는 조건속성, D는 결정속성
$V = \bigcup_{p \in A} Vp$이며, Vp는 속성 p의 정의역

예를 들어 $U = \{1, 2, 3, 4, 5, 6, 7, 8\}$, $C = \{a, b, c, d\}$, $D = \{e\}$, $Va = Vd = Ve = \{0, 1\}$, $Vb = Vc = \{0, 1, 2\}$일 때, 결정규칙 시스템을 표 1과 같은 결정규칙 테이블로 나타낼 수 있다.

$P \subset A$, $x_i, x_j \in U$일 때 U의 동치관계 R은 $\{(x_i, x_j) \in U \times U$: for every $p \in P$, $p(x_i) = p(x_j)\}$로 정의한다[2]. 즉, S에 있는 속성집합 P에서 $p \in P$일 때 $p(x_i) = p(x_j)$이면 x_i와 x_j는 동치이며, x_i의 동치 클래스는 $[x_i]_R$로 표현한다.

$P \subseteq R$이면 $\bigcap P$ 또한 동치관계로서 IND(P)로 표현하며 P에 대한 식별불가능 관계라 한다[11].

$S = \{U, A, V\}$에서 U의 요소 x_i에 대해 동치관계 R의 동치클래스를 A의 기본집합이라 하며, $X \subset U$일 때, X의 R-긍정영역(positive region)은 다음과 같이 정의된다[11].

$$POS_R(X) = \{x_i \in U \mid [x_i]_R \subset X\}$$

즉, 긍정영역은 지식 R을 사용하여 집합 X의 원소로 확실히 분류되는 객체의 집합이다.

$P, Q \subset R$에서, 지식 Q는 지식 P에 $k(0 \le k \le 1)$만큼 종속될 때, 종속도 k는 다음과 같이 정의된다[11].

$$k(P,Q) = \frac{|POS_P(Q)|}{|U|} \quad (8)$$

$k=1$이면 Q는 P에 완전히 종속되며, 그렇지 않으면 부분적으로 종속이다. P를 조건속성, Q를 결정속성이라 할 때, k는 $P \Rightarrow_k Q$인 결정 테이블의 품질척도라 할 수 있다. 속성간 종속도는 결정규칙 집합의 품질척도로 사용되지만 속성 감축의 기준으로도 이용된다.

4.2 속성 감축

결정규칙 시스템에서 속성간의 관계를 분석하여 불필요한 속성을 발견하면 이를 제거함으로써 간략화할 수 있다. 속성 감축은 불필요한 속성을 제거하고 전체 속성집합과 같은 품질 척도를 갖는 최소의 부분 속성집합으로 정의한다[8].

$S=\{U, A, V\}$에서 $A=C\cup D$이고 $B\subset C$일 때, $POS_B(D)=POS_{B-\{p\}}(D)$라면 D에 대해 속성 $p\in B$는 B에서 불필요(dispensable) 속성이고, 그렇지 않으면 필요(indispensable) 속성이다. 모든 $p\in B$가 필요 속성이면 B는 독립이다.

특정 속성이 규칙 시스템에서 불필요하다면 원 시스템의 종속관계에 영향을 주지 않고 규칙 시스템에서 제거할 수 있다. D에 대해 C에 있는 필수 속성집합이 C의 core로, 속성 감축에서 제거할 수 없는 속성이다.

$$CORE(C, D) = \{a\in C \mid POS_C(D) \neq POS_{C-\{a\}}(D)\}$$

감축은 동치관계 R을 변화시키지 않으므로, 집합 $X\subseteq U$가 주어지면 $POS_R(X)$도 변하지 않는다. 이는 감축된 기본집합으로 규칙 생성에 사용하더라도 X에 관련된 분류의 정확도가 변하지 않으므로, 원래의 속성집합보다 감축을 사용하는 것은 더 간결한 결정규칙을 생성할 수 있음을 의미한다.

표 1에서 $C=\{a, b, c, d\}$, $D=\{e\}$일 때,

$$POS_C(D) = POS_{C-\{a\}}(D)$$
$$POS_C(D) \neq POS_{C-\{b\}}(D)$$
$$POS_C(D) = POS_{C-\{c\}}(D)$$
$$POS_C(D) = POS_{C-\{d\}}(D)$$

임으로 $CORE(C, D)=\{b\}$이다. 따라서 감축의 가능성은 $\{b\}$, $\{a, b\}$, $\{b, c\}$, $\{b, d\}$, $\{a, b, c\}$, $\{b, c, d\}$, $\{a, b, d\}$, $\{a, b, c, d\}$인데, 이를 모두 조사해 보면 $\{b, c\}$, $\{b, d\}$만이 D에 대해 독립이므로 감축 $RED(C, D) = \{b, c\}, \{b, d\}$이다.

n개의 조건을 가진 규칙은 최고 2^n-1개의 부분집합에 대한 감축 가능여부를 조사해야 하는데, 이는 지수적 시간 복잡도를 가질 뿐 아니라, 많은 감축중 어느 것이 가장 좋은 것인지 판단할 수가 없다. 실제 많은 응용에 한 개 또는 몇 개의 감축만 있어도 되는 경우가 많으며, 또한 가장 좋은 감축을 찾아낼 수 있다면 이것으로 최적의 결정규칙을 도출할 수 있다. 따라서, 본 연구에서는 모든 경우의 조사가 아닌 휴리스틱(heuristic)한 방법으로 좋은 감축을 찾아내는 방법을 사용한다. 이 방법은 Cercone[1]의 연구를 기반으로 한 것으로, 가장 좋은 감축을 찾아내기 위해서 속성의 중요도 순서로 부분집합을 구성하여 가장 먼저 감축으로 형성되는 속성집합을 최적 감축으로 판단한다.

감축은 조건속성 집합에서 결정속성에 필수적인 core 속성을 추출하고 core 속성을 포함하는 속성의 조합에 의하여 감축을 계산하는 집합론적 방법을 사용하나, 본 연구에서는 계산의 효율성을 위해 종속도를 조사하는 방법을 사용한다.

$S=\{U, A, V\}$, $A=C\cup D$에서 조건속성 C에 대한 결정속성 D의 종속도가 $k(C, D)$일 때 $k(C, D)=k(C-\{a\}, D)$이면 $a\in C$는 불필요하고 그렇지 않으면 필수 속성이므로, $B\subseteq C$, $a\in B$일 때 다음 조건을 만족하면 B는 C의 감축이다[1].

$$k(B, D) = k(C, D) \text{ and } k(B, D) \neq k(B-\{a\}, D)$$

속성 감축 알고리즘은 알고리즘-4와 같다.

알고리즘-4 Attribute_Reduct

```
input: GDB, C, D
    /* GDB: 일반화 DB,
       C: 조건속성 집합,
       D: 결정속성 집합 */
output: RED   / * 감축 집합 */
begin
   generate discernible matrix from GDB;
   CO = core;
   AR = C - CO;  /* AR: 남은 속성집합 */
   calculate significance for each attribute a∈AR;
   sort AR by significance;
   RED = CO;
   while K(RED, D) ≠ K(C, D) do/* 감축 생성 */
   select next a_j in AR;
   RED = RED∪{a_j};
   AR = AR - {a_j};
   calculate K(RED, D);  /* 종속도 계산 */
   end while
N = |RED|;  /* 감축집합의 속성수 */
for j = 1 to N  /* 불필요 속성 조사 */
   if a_j ∉ CO then
```

```
        RED = RED − {a_j};
        calculate K(RED,D);
        if K(RED, D) ≠ K(C, D) then
            RED = RED∪{a_j};
            else AR = AR∪{a_j};
          end if
        end if
      end for
      merge duplicate tuples and count;
    end
```

속성수를 n이라 할 때 이 알고리즘의 시간 복잡도는 $O(n^2)$이다. 이는 라프셋 이론에서의 속성감축 시간 복잡도가 $O(2^n)$임에 비해 매우 효율적이다.

4.3 식별가능 행렬과 함수

속성 및 속성값을 효율적으로 감축하는데 식별가능(discernible) 행렬과 식별가능 함수를 사용할 수 있다[14,16].

결정규칙 시스템 $S=(U, A, V)$에서 조건속성 C에 대한 두 객체 $x_i,x_j \in U$ 간의 차이 측정은 거리함수 $\delta_C(x_i, x_j)$로 표시한다[10].

$$\delta_C(x_i, x_j) = \{a \in C \mid v_a(x_i) \neq v_a(x_j)\}$$

C가 조건속성, D가 결정속성이면 D에 대한 C의 식별가능 행렬 $M_D(C)=\{m_{ij}\}_{nxn}$는 다음과 같이 정의한다.

$$(m_{ij}) = \{a \in C \mid \delta_C(x_i, x_j) \text{ and } w(x_i, x_j)\} \quad (9)$$
$$\text{for } i, j = 1, 2, \cdots, n$$
$$\text{where } w(x_i, x_j) = x_i \in POSc(D) \text{ and } x_j \notin POSc(D)$$
$$\text{or } x_i \in POSc(D) \text{ and } x_j \in POSc(D)$$
$$\text{or } x_i, x_j \in POSc(D) \text{ and } x_i, x_j \notin IND(D)$$

m_{ij}는 x_i, x_j를 분별하는 속성의 집합으로, $M_D(C)$는 대칭이므로 m_{ij}는 $1 \leq j < i \leq n$에 대하여 계산한다.

예를 들어 표 2와 같은 결정규칙 테이블이 있다고 하자.

표 2. 결정 규칙 테이블

U	b	c	e
1	0	1	0
2	0	2	1
3	1	1	0
4	2	1	1
5	2	2	1

표 3. 감축 {b, c}의 식별가능 행렬

U	1	2	3	4	5
1					
2	c				
3		bc			
4	b		b		
5	bc		bc		

표 4. 감축된 결정규칙 테이블

U	a	b	e
1	0	1	0
2	-	2	1
3	1	-	0
4	2	-	1
5	2	-	1
5'	-	2	1

표 2에서 {b, c}를 조건속성, {e}를 결정속성이라 할 때, 식별가능 행렬은 표 3과 같다.

결정규칙 시스템에서 각 객체에 대한 감축을 유도하기 위해 식별가능 함수를 다음과 같이 정의한다.

$$f_D^x(C) = \prod_{xj \in U} \{\sum \delta_C(x_i, x_j) \mid \delta_C(x_i, x_j) \neq \emptyset\} \quad (10)$$

$\sum \delta_C(x_i, x_j)$: 속성집합 $\delta_C(x_i, x_j)$에 할당된 논리변수의 논리합

표 3에서 각 객체들에 대한 식별가능 함수 유도하고 흡수법칙에 의해 간략화하면 다음과 같다.

$$f_D^1(C) = cb(b \vee c) = bc$$
$$f_D^2(C) = c(b \vee c) = c$$
$$f_D^3(C) = (b \vee c)b(b \vee c) = b$$
$$f_D^4(C) = bb = b$$
$$f_D^5(C) = (b \vee c)(b \vee c) = b \vee c$$

위 객체별 식별가능 함수를 감축된 결정규칙 테이블로 표시하면 표 4와 같다.

본 연구에서는 계산의 효율을 위해 식별가능 행렬과 함수를 이용하여 속성값을 감축하고 최소 결정 테이블을 유도한다.

표 4를 수식 형태의 결정규칙으로 표현하면 다음과 같다.

$$b0c1 \vee b1 \rightarrow e0$$
$$c2 \vee b2 \rightarrow e1$$

본 연구에서는 최종적으로 일반화된 결정 규칙을 개념적으로 나타내기 위해 수식을 다음과 같이 IF-THEN 규칙으로 변환하여 표현한다.

IF $b=0$ and $c=1$ or $b=1$ THEN $e=0$
IF $c=2$ or $b=2$ THEN $e=1$

5. 시스템 구현 및 실험

본 최적 규칙 발견 시스템은 Windows98에서 VB5.0으로 구현하고 데이터베이스는 ACCESS97을 사용한다. 실험 데이터는 계명대학교 성적 데이터베이스를 사용하여 졸업성적과 입학성적간의 관련성을 규칙으로 유도하고 그 결과를 평가한다.

5.1 시스템의 구조

본 시스템은 그림 1과 같이 사용자 인터페이스, 개념 트리 생성 모듈, 규칙 생성 기관, 데이터베이스로 구성된다. 사용자 인터페이스는 각종 사용자 입출력과 실행과정을 대화식으로 처리하며, 규칙 생성 기관은 개념 상승 모듈, 감축 모듈, 규칙변환 모듈로 구성된다. 대규모 데이터베이스로부터 신뢰성있는 규칙을 유도할 수 있다는 의미에서 규칙 생성의 대상이 되는 데이터베이스를 VLDB(Very Large-scale Database)로 표시한다.

- 개념 트리 생성 모듈 : VLDB로부터 필요 속 성별 개념 트리를 생성한다.
- 개념 상승 모듈 : 개념 트리를 사용하여 VLDB를 개념 상승한 일반화 DB로 변환한다. 개념 상승수준 임계치와 예외사항을 제 거하기 위한 필터 임계치는 응용에 따라 사용자가 입력한다.
- 감축 모듈 : 일반화 DB로부터 불필요한 속성 및 속성값을 제거하여 최적 규칙을 도출한다. 규칙 유도를 위한 조건속성과 결정속성을 사용자가 입력한다.
- 규칙 변환 모듈 : 최적 규칙을 개념적인 언어 로 표현한 일반화 규칙으로 변환한다.

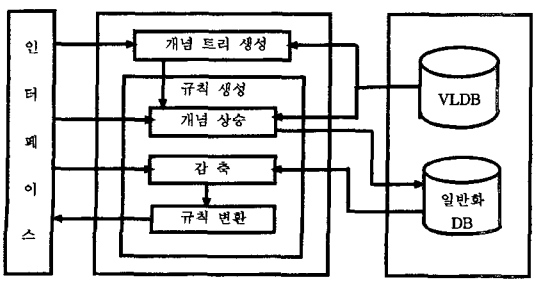


그림 1. 시스템 구조

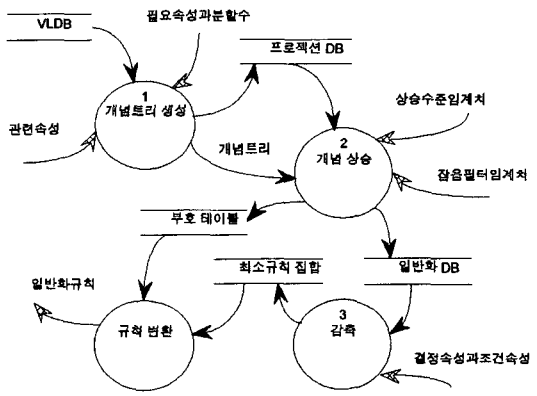


그림 2. 시스템의 자료흐름도

개념 트리 생성과 규칙 생성 기관을 중심으로 한 자료흐름도는 그림 2와 같다. 부호 테이블은 일반화 DB의 속성값을 효율적으로 처리하기 위해 부호로 변환시키기 위한 것이다. 이 테이블은 최적 규칙집합을 일반화 규칙으로 변환시킬 때 사용된다.

5.2 실험

본 실험에서는 98년 계명대학교 공과대학을 졸업한 학생들에 대하여 입학성적이 졸업평점에 미치는 영향을 규칙으로 유도해 본다.

졸업 데이터베이스에서 규칙 생성과 관련된 속성은 출신지역, 재수구분, 내신점수, 수능점수, 면접등급, 졸업평점이다. 개념 트리의 생성에 필요한 속성은 출신지역, 내신점수, 수능점수, 졸업평점이며, 출신지역을 제외한 속성들은 수치이므로 자동으로 개념 트리 생성이 가능하다.

출신지역은 기존의 지식에 의하여 수동적으로 대도시, 중도시, 소도시로 구분하여 생성하고, 내신점수, 수

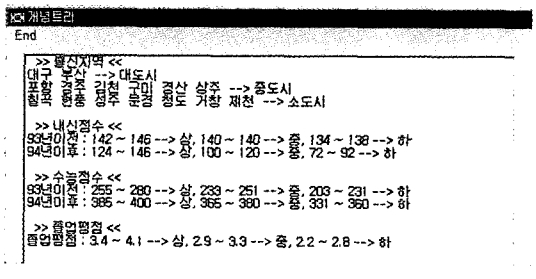


그림 3. 생성된 개념 계층

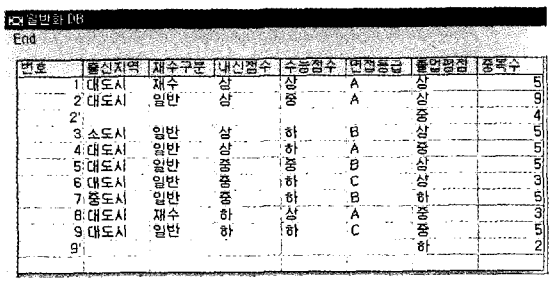
능점수, 졸업평점은 클러스터링 방법에 의해 상, 중, 하로 자동생성한다. 본 시스템에 의하여 자동생성된 개념계층은 그림 3과 같다.

개념 트리가 리프노드와 루트노드를 포함하여 3레벨이므로 상승수준 임계치는 2로 설정하며, 100개의 사례중 적어도 2개 이상의 사례가 일반화 규칙으로 지지를 받을 수 있을 것으로 판단되어 필터 임계치는 2%로 설정한다. 몇 개의 튜플에서 조건속성이 동일하나 졸업평점이 상이한 모순이 발생하는데, 이는 중복된 튜플의 수를 사용하여 추후 생성되는 규칙에 확률로 표시한다. 졸업 데이터베이스를 그림 3의 개념 계층으로 상승시킨 일반화 DB는 그림 4와 같다.

본 실험에서는 졸업평점을 결정속성으로 하고, 졸업평점과 관련된 규칙을 유도하기 위해 출신지역, 재수구분, 내신점수, 수능점수, 면접등급을 조건속성으로 한다.

식별가능 행렬을 구성하여 조사한 결과 단일 속성은 내신점수와 수능점수인데, 이것이 감축의 core이다. 그리고 core 속성을 제외한 속성인 출신지역, 재수구분, 면접등급의 중요도를 계산하면 출신지역이 0.419, 면접등급이 0.303, 재수구분이 0.088로, 재수구분은 졸업평점에 거의 영향을 미치지 않음을 알 수 있다.

core 속성과 속성의 중요도를 사용하여 일반화 DB를 감축하면 {출신지역, 내신점수, 수능점수}가 최적 감축으로 생성된다. 이를 언어변수로 변환하여 표현하고, 모순된 규칙에 대해서는 확률로 나타낸 최종 일반화 규칙은 그림 5와 같다.


일반화 DB
End

번호	출신지역	재수구분	내신점수	수능점수	면접등급	졸업평점	중복수
1	대도시	재수	상	상	A	상	5
2	대도시	일반	상	중	A	상	9
2'						중	4
3	소도시	일반	상	하	B	상	5
4	대도시	일반	상	하	A	중	5
5	대도시	일반	중	중	B	상	5
6	대도시	일반	중	하	C	상	3
7	중도시	일반	중	하	B	하	5
8	대도시	재수	하	상	A	중	3
9	대도시	일반	하	하	C	중	5
9'						하	2

그림 4. 생성된 일반화 DB

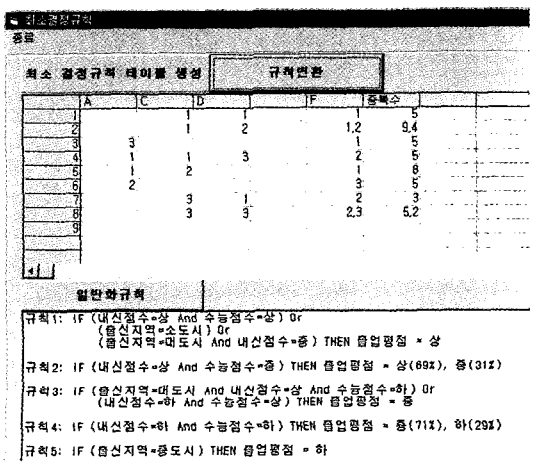


그림 5. 최소 규칙 집합

5.3 평가

본 연구에서 제안한 방법을 속성중심 귀납법[6] 및 라프셋에 의한 지식감축 방법[11]과 비교하여 평가해 보면, 먼저 규칙의 간략성에 있어서 속성중심 귀납법은 개념 상승만 하여 규칙을 유도하므로 중복 규칙을 포함하고 있을 가능성이 있음에 비해 본 방법은 속성간 종속성을 분석하여 불필요 속성 및 속성값을 제거하므로 간략성이 매우 높다. 규칙의 추상성에 있어서 라프셋 방법은 원시 데이터 그대로 감축을 하므로 속성값들의 추상성이 부족함에 비해 본 방법은 속성값을 먼저 일반화시켜 규칙으로 표현하므로 추상성이

표 5. 규칙의 검증

	규칙	졸업평점	확률	테스트 데이터의 확률		
1	내신점수 = 상 and 수능점수 = 상	상	100%	상	83%	81%
				중	17%	
	출신지역 = 소도시			상	75%	
				중	13%	
				하	12%	
	출신지역 = 대도시 and 수능점수 = 중			상	85%	
				중	15%	
2	내신점수 = 상 and 수능점수 = 중	상	69%	상	75%	75%
		중	31%	중	25%	25%
3	출신지역 = 대도시 and 내신점수 = 상 and 수능점수 = 하	중	100%	상	11%	69%
				중	67%	
				하	22%	
	내신점수 = 하 and 수능점수 = 상			상	10%	
				중	70%	
				하	20%	
4	내신점수 = 하 and 수능점수 = 하	중	71%	중	67%	67%
		하	29%	하	33%	33%
5	출신지역 = 중도시	하	100%	중	17%	83%
				하	83%	

높아 규칙을 이해하기가 쉽다. 그리고, 규칙의 최소화 및 최적화에 있어서 속성중심 귀납법은 고려하고 있지 않으며, 라프셋 방법은 모든 가능 감축을 모두 계산하므로 감축 시간이 지수적인 시간복잡도를 가질 뿐 아니라 어느 감축이 최적인지 판단을 할 수 없다. 이에 비해 본 방법은 속성의 중요도를 고려하여 하나의 최적 감축을 생성하므로 감축 속도가 $O(n^2)$의 시간복잡도를 가지며, 또한 감축의 최적성을 판단할 수 있다.

본 실험 결과를 99년 졸업 데이터로 검증한 결과는 표 5와 같다. 테스트 데이터의 확률은 테스트 데이터를 각 규칙에 적용했을 때, 졸업평점이 일치하는 비율이다.

이 표로부터 실험 결과의 신뢰도를 분석해 보면 각 규칙별로 다음과 같이 계산된다.

규칙 1: $1-(100-81)/100 = 81\%$
규칙 2: $1-\{(75-69)/75+(31-25)/31\}/2 = 86\%$
규칙 3: $1-(100-69)/100 = 69\%$
규칙 4: $1-\{(71-67)/71+(33-29)/33\}/2 = 91\%$
규칙 5: $1-(100-83)/100 = 83\%$

따라서 전체 규칙의 평균 신뢰도는 82%로서 98년 졸업생 자료로부터 유도한 결정규칙이 99년 졸업생 자료에도 잘 적용됨을 보인다.

6. 결 론

본 연구에서는 대량의 데이터로부터 최적의 규칙을 발견하기 위해 데이터베이스를 개념적으로 일반화하여 데이터의 크기를 줄이고, 결정속성에 영향을 미치는 조건속성의 중요도에 따라 불필요한 조건속성을 제거하여 속성의 수를 줄이며, 속성간의 종속관계를 분석하여 불필요한 속성값을 제거하는 방법으로 간략화된 형태의 최적 규칙을 도출하는 방법을 제시했다. 이를 위해 클러스터링 방법에 의한 개념계층 생성의 자동화, 개념 상승에 의한 데이터베이스의 일반화, 정보이득 측정에 의한 속성의 중요도 계산, 중요도를 이용한 속성 감축에 의한 최적 감축, 식별가능 행렬을 이용한 효율적인 속성값 감축을 연구하고 알고리즘을 설계했으며 프로토타입 시스템으로 구현했다. 그리고 본 시스템이 기존 시스템에 비해 간략성과 추상성이 좋은 지식을 도출하고, 최소화 및 최적화된 규칙을 유도하며, 생성된 규칙이 새로운 데이터에 잘 적용될 수 있는 가능성을 실험을 통하여 보였다.

그런데 본 시스템을 유효하게 사용하려면 신뢰성 있는 데이터의 입력이 필수적이다. 본 연구에서는 모순된 규칙에 대해서 확률이라는 정량적 값을 표시하거나 필터를 사용하여 신뢰성이 적은 규칙을 제거하는 방법을 사용하였으나, 이는 완벽한 해결책이라 볼 수 없다. 따라서 향후 과제로는 불완전한 데이터를 더 효과적으로 처리하는 방법을 연구해야 하며, 신뢰성이 적은 데이터를 적절하게 처리할 수 있는 방법과 알고리즘을 계속 연구해야 한다.

참고문헌

[1] N. Cercone, H. Hamilton, X. Hu, and N. Shan, "Data Mining Using Attribute-Oriented Generalization and Information Reduction," *Rough Sets and Data Mining*, T. Lin and N. Cercone (eds), Kluwer, pp. 199-227, 1997.

[2] D. Cheung, A. Fu and J. Han, "Knowledge Discovery in Databases: A Rule-Based Attribute-Oriented Approach," http://www. kdnuggets.com/, 1999.

[3] W. Chu, H. Yang, K. Chiang, M. Minock, G. Chow, and C. Larson, "CoBase: A Scalable and Extensible Cooperative Information System," *Intelligent Integration of Information*, G. Wiederhold (eds), JIIS, Vol. 6, No. 2/3, pp. 223-259, 1996.

[4] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy (eds), *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, 1995.

[5] D. Fisher, "Knowledge Acquisition via Incremental Conceptual Clustering," *Machine Learning*, vol. 2, no. 2, pp. 139-172, 1987.

[6] J. Han, Y. Cai, and N. Cercone, "Knowledge Discovery in Databases: An Attribute-Oriented Approach," *Proceeding of the 18th Conference on Very Large Data Bases*, Vancouver, Canada, pp. 340-355, 1992.

[7] X. Hu, N. Cercone, and J. Han, "An Attribute-Oriented Rough Set Approach for Knowledge Discovery in Databases," *Proc. RSKD'93*, Banff, Alberta, Canada, pp. 90-99,Oct. 12-15, 1993.

[8] X. Hu, N. Cercone, and W. Ziarko, "Generation of Multiple Knowledge from Databases Based on Rough Set Theory," *Rough Sets and Data Mining*, T. Lin and N. Cercone (eds), Kluwer, pp. 109-121, 1997.

[9] M. Kamber, L. Winstone, W. Gong, S. Cheng, and J. Han, "Generalization and Decision Tree Induction: Efficient Classification in Data Mining," http:// www.kdnuggets.com/, 1999.

[10] M. Kryszkiewicz and Henryk Rybinski, "Finding Reducts in Composed Information Systems," *Proc. RSKD'93*, Banff, Alberta, Canada, pp. 261-273, 12-15 Oct., 1993.

[11] Z. Paw"lak, Rough Sets-*Theoretical Aspects of Reasoning about Data*, Kluwer, 1991

[12] J. Quinlan, "Induction of Decision Trees," *Machine Learning*, Vol. 1, No. 1, pp. 81-106, 1986.

[13] *Siftware: Tools for Data Mining and Knowledge*

Discovery, http://www. kdnuggets.com/, 1999.
[14] A. Skowron and C. Rauszer, "The Discernibility Matrices and Functions in Information Systems," Slowinski (eds), *Intelligent Decision Support-Handbook of Advances and Applications of the Rough Set Theory*, Kluwer, pp. 311-362, 1991.
[15] 박영택, 이강로 "ID3계열의 귀납적 기계학습," 정보과학회지, 제13권, 제5호, pp. 6-18, 1995.
[16] 이성주, 정환묵, 최완규, 러프집합과 응용, 조선대학교 출판국, 1998.
[17] 정 홍, 최경옥, 정환묵, "Generation of Approximation Rules Using Information Gain," *FUZZ-IEEE '99, The 8th Int'l Conf. on Fuzzy Systems*, Seoul, Korea, Aug. 22-25, 1999.

정 홍 (Hong Chung)

제 9 권 제 4 호 참조

김 진 상 (Jin-Sang Kim)

1974년~1978년 : 경북대학교 수학교육학과 이학사
1979년~1981년 : 한국과학기술원 전산학과 이학석사
1987년~1991년 : 영국 Imperial College 전산학과 박사과정
1981년~1982년 : 한국과학기술원 전산개발센터 연구원
1982년~현재 : 계명대학교 컴퓨터공학과 부교수
관심분야 : 인공지능, 기계학습, 지식발견, 에이전트 시스템