韓國컴퓨터情報學會 論文誌
第15卷 第8號, 2010. 8.

2010-15-8-1-2

DHP 연관 규칙 탐사 알고리즘을 위한 해싱 메커니즘 최적화

이 형 봉*, 권 기 현**

An Optimization of Hashing Mechanism for the DHP Association Rules Mining Algorithm

Hyung-Bong Lee*, Ki-Hyeon Kwon**

요 약

DHP 연관 규칙 탐사 알고리즘의 가장 큰 특징은 단계 *k-1*에서 *k* 개의 항목으로 구성된 해시 키 조합에 대한 계수를 미리 실시하고, 이를 단계 *k*에서 후보 빈발 항목 집합을 구성할 때 전지 정보로 활용하여 그 크기를 줄임으로써 성능을 개선한다는 점에 있다. 이 때, 모든 해시 키 조합에 대한 계수를 독립적으로 관리할 수 있다면 가장 이상적이나, 메모리 소요가 너무 많으므로 여러 개의 해시 키 조합들이 계수 공간을 공유하는 직접 해싱 메커니즘을 활용한다. 그러나, 연관 규칙 탐사 알고리즘의 특성상 해시 키 조합의 분포 공간이 불규칙하여 해싱 함수에 일반적인 단순 제산 연산을 사용할 경우 직접 해싱의 효율이 저하된다. 이 논문에서는 단계 3을 위한 길이 3인 해시 키 공간을 연속되는 정수 공간으로 사상하여 직접 해싱의 효율을 극대화시키는 사상 완전 해싱 함수를 제안한다. 42개의 시험 데이터 유형을 대상으로 실험한 결과 제안된 해싱 함수는 기존 방법보다 평균 7.3%, 최대 16.9%의 성능 개선 효과가 있는 것으로 나타났고, 특히 평균 거래 길이, 평균 빈발 항목 집합의 크, 전체 항목의 개수 등이 클수록 성능 개선 정도가 높았다.

Abstract

One of the most distinguished features of the *DHP* association rules mining algorithm is that it counts the support of hash key combinations composed of *k* items at phase *k-1*, and uses the counted support for pruning candidate large itemsets to improve performance. At this time, it is desirable for each hash key combination to have a separate count variable, where it is impossible to allocate the variables owing to memory shortage. So, the algorithm uses a direct hashing mechanism in which several hash key combinations conflict and are counted in a same hash bucket. But the direct hashing mechanism is not efficient because the distribution of hash key combinations is unvalanced by the characteristics sourced from the mining process. This paper proposes a mapped perfect hashing function which maps the region of hash key combinations into a continuous integer space for phase 3 and maximizes the efficiency of direct hashing mechanism. The results of a performance test experimented on 42 test data sets shows that the average performance improvement of the proposed

• 제1저자 : 이형봉 교신저자 : 권기현
• 투고일 : 2010. 05. 19, 심사일 : 2010. 07. 09, 게재확정일 : 2010. 07. 14.
* 강릉원주대학교 컴퓨터공학과 교수 ** 강원대학교 전자정보통신공학부 교수

hashing mechanism is 7.3% compared to the existing method, and the highest performance improvement is 16.9%. Also, it shows that the proposed method is more efficient in case the length of transactions or large itemsets are long or the number of total items is large.

▸ Keyword : DHP, 직접 해싱(direct hashing), 사상 완전 해싱(mapped perfect hashing)

Ⅰ. 서 론

사건들 사이에 숨어 있는 연관 관계를 탐색하는 연관 규칙 탐색 알고리즘은 오늘날 데이터베이스의 고유 이론 분야[1] 뿐만 아니라 공학 분야[2], 기업 비즈니스 활동 분야[3, 4], 실생활과 밀접한 인문사회 과학 분야[5,6] 등 다양한 분야에서 매우 요긴하게 사용되고 있다. 이러한 연관 규칙 탐사의 필요성은 미래에도 지속될 것임은 두 말할 나위 없이 자명하다. 이와 같은 연관 규칙 탐색에 대한 수요에 부응하기 위해 최근 주요 DBMS 공급 업체들이 새로운 OLAP(On-Line Analytical Processing) 도구를 출시하는 사례[7]가 늘고 있으며, 데이터베이스 전문가나 기업인이 아닌 일반 사용자들을 위한 공개 소프트웨어 형태[8]도 제공되고 있다.

항목들 사이에 존재하는 연관 규칙은 신뢰도와 지지도 등 두 가지 관점에서의 수준을 모두 만족하여야만 그 의미를 가질 수 있다. 이를테면 구매자들이 품목 A와 B를 구매했을 때, 품목 C를 동시에 구매한 경우가 95%라고 했을 때 신뢰도가 95%인데, 만약 A, B를 동시 구매한 경우의 수 자체 즉, 지지도가 매우 소수라면 이 때의 신뢰도에는 큰 의의를 부여할 수 없다. 따라서, 연관 규칙 탐사 알고리즘의 일차적인 목적은 지지도가 일정 수준 이상이 되는 빈발 항목 집합(large itemset or frequent itemset)들을 발견하는 데 있다. 일단 빈발 항목 집합들이 발견되면, 그 각각의 내부에 존재하는 항목들 사이에서 일정 수준 이상의 신뢰도를 만족하는 연관 규칙을 찾아내는 일은 비교적 용이하다. 따라서, 대부분의 연관 규칙 탐사 알고리즘은 곧 빈발 항목 집합을 효율적으로 찾는 알고리즘을 의미하며, 이 논문에서도 이를 따른다.

연관 규칙 탐사 알고리즘의 유형을 분류하면 크게 해시 기반 알고리즘과 빈발 패턴 트리 기반 알고리즘으로 나눌 수 있는데, DHP(Direct Hashing and Pruning) 알고리즘[9]과 비교적 최근에 발표된 FPT(Frequent Pattern Tree) 알고리즘[10]이 그 각각을 대표한다. 그런데, DHP 알고리즘이 DB를 여러 번 스캔하는 반면, FPT 알고리즘은 DB를 2회만 스캔하므로 FPT 알고리즘이 월등하게 우수하다고 단정하는 경우가 있다. 그러나, 이들 두 가지 알고리즘에 대한 성능 비교 연구[11]에 의하면 105 가지의 다양한 거래 데이터 유형 중 FTP 알고리즘이 우수한 경우는 14 가지에 지나지 않는다.

이 연구에서는 DHP 알고리즘을 위한 효율적인 해싱 메커니즘을 제안하여 성능 향상에 기여함으로써, 이 알고리즘의 활용도를 더욱 높이고자 한다. 이를 위하여 2 장에서 관련 연구를 살펴보고, 3장에서 해싱 메커니즘의 최적화 방안을 제안하며, 4장에서 개선된 방법의 효과를 검증하고, 마지막 5장에서 결론으로 이 논문을 맺는다.

Ⅱ. 관련연구

2.1 연관 규칙 탐사 알고리즘 관련 용어

연관 규칙 탐사 알고리즘은 대량의 거래 자료로부터 거래 항목들 사이에 존재하는 동시 거래 관련성을 발견하는 것으로, 지지도(support), 신뢰도(confidence), 항목 집합(itemset), 빈발 항목 집합(large itemset), 선두 빈발 후보 항목 집합(frontier large itemset), 후보 항목 집합(candidate large itemset), 최대 빈발 항목 집합(maximal large itemset), 길이가 k인 항목 집합(k-item set), 전체 항목 집합 등의 용어와 표 1의 기호 들을 사용한다[9].

표 1. 연관 규칙 탐사에서 사용되는 기호
Table 1. Symbols for association rules mining

기 호	의 미
D_k, t_k, t	단계 k에서의 데이터베이스, $t_k, t \in D$
L_k, l_k, l	단계 k에서의 {빈발 항목 집합}, $l_k, l \in L_k$
F_k, f_k, f	단계 k에서의 {선두 빈발 항목 집합}, $f_k, f \in F_k$
C_k, c_k, c	단계 k에서의 {후보 빈발 항목 집합}, $c_k, c \in C_k$
H_k, h_k	단계 k에서의 직접 해쉬 테이블, 해쉬 함수
I, i_j	전체 항목 집합, $i_j \in I$
$Nnnnn$	전체 항목 집합의 크기($\|I\|$)가 $nnnn$ 임
$TxIyDz$	거래의 평균 크기 x, 빈발 항목 집합의 평균 크기 y, 총 거래 건수 $z*1000$개인 거래 데이터

2.2 연관 규칙 탐사 알고리즘의 진화

가장 초보적인 연관 규칙 탐사 알고리즘은 연관 규칙 탐사 대상이 되는 전체 항목 집합으로부터 유도되는 모든 부분 집합(공집합 제외)에 대한 계수 공간을 마련한 후, 각 거래(transaction)에 포함된 부분 집합들을 찾아 계수를 실시하는 방법이다. 이 알고리즘은 전체 DB를 한 번만 스캔한다는 측면에서 매우 효율적인데, 실제로 전체 항목 수가 아주 적은 경우에는 다른 어떤 알고리즘 보다 우수 할 수 있다[11]. 그러나, 전체 항목 수가 커지면 계수 공간 자체의 동시 확보가 곤란하다. 이를 극복하기 위해서는 가능성이 있는 빈발 항목 집합만을 추출하여 단계적으로 계수하는 방안을 모색해야 하는데, AIS(finding All frequent Item-Sets)[12]가 그 첫 번째이다.

AIS 알고리즘은 DB에 저장된 거래들을 차례로 스캔하면서 각 거래에 포함된 항목들로 구성된 후보 빈발 항목 집합(C)을 도출하여 계수한다. 이 때, 후보 빈발 항목 집합들은 한 항목 단위로 확장하고, 각 항목의 상대 빈도 통계를 고려하여 확장되는 항목 집합이 빈발 항목 집합(L_k)이 될 가능성이 낮아지거나, 더 이상 확장할 항목이 없는 순간까지만 확장을 계속한다. 이와 같이 확장된 후보 빈발 항목 집합은 이후 거래부터 계수의 대상이 되고, 거래 전체에 대한 스캔이 완료되었을 때 빈발 항목 집합 여부가 최종 결정된다. 그런데, 빈발 항목 집합으로 판명된 후보 빈발 항목 집합 중에서 통계적 가능성에 의해 확장이 중단됐던 것들에 대하여는 추가적인 확장을 더 탐색해 봐야 하므로, 이들을 기반($F_{k:}$)으로 하는 확장 및 계수 과정을 다시 실시해야 하며, 이러한 과정은 더 이상의 확장이 존재하지 않을 때까지 반복되어야 한다. 즉, AIS 알고리즘은 각 항목의 상대 빈도를 이용하여 이들로 구성된 후보 빈발 항목 집합을 점진적으로 확장해 나아감으로써 일시에 요구되는 계수 공간을 억제한다.

Apriori[13] 알고리즘은 AIS 알고리즘의 후보 빈발 항목 집합에 대한 적중률을 더욱 향상시키기 위해 단계별 후보 빈발 항목 집합(C_k)의 크기를 1만큼 늘려가는 전략을 도입하여 성능을 향상시켰다. 즉, 단계 1에서 모든 빈발 항목을 찾고, 단계 2에서는 두 빈발 항목의 조합으로 구성되는 길이 2인 후보 빈발 항목 집합(C_2)을 유도하여 계수한다. 단계 3에서는 길이 2인 빈발 항목 집합(L_2)으로부터 길이 3인 후보 빈발 항목 집합(C_3)을 도출하여 계수한다. 이러한 과정이 더 이상의 빈발 항목 집합이 발견되지 않을 때까지 계속된다.

DHP(Direct Hashing and Pruning)[9] 알고리즘은 Apriori 알고리즘의 실행 시간 대부분(70% 이상)이 단계 2에서 소모되는 특성에 주목하여, C_2의 크기를 최대한 감소시키기 위한 방법으로 직접 해싱을 도입하여 획기적인 성능 향상 효과를 얻었다. 직접 해싱은 단계 1에서 쉽게 얻어지는 두 항목의 조합에 대한 계수를 미리 실시하여 단계 2를 위해 요구되는 지지도에 미치지 못한 조합은 C_2에서 제거한다. 이를테면, {A, B}, {A, D}, {A, C}를 동일한 해시 버킷에 할당하여 함께 계수한 결과가 주어진 지지도를 넘지 못하면 이들 세 항목 집합 어느 것도 후보 항목 집합이 될 수 없음은 자명하다.

가장 나중에 제안된 FPT(Frequent Pattern Tree) [10] 알고리즘은 AIS, Apriori, DHP 알고리즘 등에서 요구되었던 여러 번의 DB 스캔을 2 회로 단축시켜 DB 스캔에 따른 부담을 줄이고자 하였다. 이 알고리즘은 단계 1에서 빈발 항목들을 찾고, 단계 2에서 빈발 항목들만이 참여는 FP-tree를 구축하여 일종의 메모리 DB 상에서 연관 규칙을 탐색하기 때문에 가장 우수한 연관 규칙 탐사 알고리즘으로 인색되는 경우가 많다. 그러나, 이 알고리즘은 DB 스캔 부담은 적지만, 메모리 소모가 많고 FP-tree 검색을 위한 계산 부담이 크기 때문에, DHP에 비해 반드시 우수하다고 단정할 수는 없다[11].

2.3 DHP 알고리즘의 해싱 메커니즘 개선

DHP 알고리즘의 가장 큰 특징은 단계 $k-1$에서 H_k에 대한 계수를 미리 실시하여 단계 k에서 C_k를 구성할 때 활용함으로써 C_k의 크기를 크게 줄인다는 점에 있다. 이 때, H_k를 길이 k인 모든 항목 조합 각각의 계수 공간이 분리되는 완전 해싱으로 구성할 수 있다면 C_k를 구성하지 않고 곧바로 L_k를 구할 수 있다. 만약 H_2를 완전 해싱으로 구성한다면 개의 계수 공간이 요구되므로, |I|가 1,000이라면 약 2MB(sizeof(int)x1,000x999/2), |I|가 7,000이라면 100MB 정도의 메모리가 각각 필요하다. 이 정도의 메모리 소요는 대용량 메모리 시스템에서는 큰 무리가 아니다. [14]에서는 대용량 메모리 시스템에서 DHP 알고리즘의 단계 1과 단계 2를 통합하여 H_2를 완전 해싱으로 구현함으로써 L_2를 단번에 구할 수 있는 개선된 DHP 알고리즘을 제시하여 20% 이상의 성능 향상 효과를 얻었다.

[15]에서는 DHP 알고리즘의 고유 특성을 그대로 유지하면서도 H_2를 위한 개선된 해싱 메커니즘을 제안하여 일반적인 방법보다 약 18% 이상의 성능 향상 효과를 얻었다. 즉, 식 ①의 단순 제산 해싱(smh:simple mod hashing) 함수 $h_{2smh}(i, j)$를 사용하는 일반적인 방법에서는 표 2에 보인 길이 2인 빈발 항목 집합의 특성에 의해 표 3과 같이 사용되지 않는 버켓의 비율이 높은데, [14]는 H2를 위한 식 ②의 사상 완전 해싱(mph:mapped perfect hashing) 함수$h_{2smh}(i, j)$

를 고안하여 해싱의 효과를 극대화시켰다. 이 논문에서는 H_3를 위한 사상 완전 해싱 함수 $h_{3smh}(i, j, k)$를 모색하고 그 효과를 검증하고자 한다.

표 2. 2-빈발 항목 집합 특성
Table 2. Characteristics of 2-large itemsets

첫항 \ 이항	A	B	C	▪▪▪▪	F	G	H
A		AB	AC	▪▪▪▪	AF	AG	AH
B			BC	▪▪▪▪	BF	BG	BH
C					CF	CG	CH
▪						▪	▪
G							GH
H							

$$h_{2smh}(i, j) = (i \times |I| + j) \bmod |H_2|.$$
(단, $i < j$).
…… 식 ①

$$H_{2mph}(i, j) = (i \times |I| + j - (i+1) \times (j+2)/2) \bmod |H_2|.$$
(단, $i < j$).
…… 식 ②

표 3. 단순 제산 연산 기반인 $h_{2smh}()$의 한계
Table 3. A pitfall of simple mod-based $h_{2smh}()$

해시키 ($\lfloor I \rfloor$=8, $\lfloor H_2 \rfloor$=9)									
	x	A,B	A,C	A,D	A,E	A,F	A,G	A,H	x
	x	B,C	B,D	B,E	B,F	B,G	B,H	x	x
	x	C,D	C,E	C,F	C,G	C,H	x	x	x
	x	D,E	D,F	D,G	D,H	x	x	x	x
	x	E,F	E,G	E,H	x	x	x	x	x
	x	F,G	F,G	x	x	x	x	x	x
	x	G,H	x	x	x	x	x	x	x
버켓 번호	0	1	2	3	4	5	6	7	8
히트 개수	0	7	6	5	4	3	2	1	0

III. DHP 알고리즘의 H_3를 위한 해싱 메커니즘 최적화

3.1 H_3를 위한 일반적인 단순 제산 연산 기반 해싱 함수의 한계

DHP 알고리즘에서 H_3를 위해 가능한 해시 키의 모든 조합을 그림 1에 보였다. 식 ①에 보인 H_2를 위한 해싱 함수 $h_{2smh}(i, j)$와 마찬가지로, H_3를 위한 해싱 함수에 식 ③의 일반적인 단순 제산 해싱 함수 $h_{3smh}(i, j, k)$를 적용할 경우, 해시 키 영역이 연속되지 않아 해싱 효과가 높지 않을 수 있다. 즉, 히트 버켓들이 균일하게 분포하지 않아 계수 공간이 편중됨으로써 C_3를 위한 전지 효과가 낮아지는 것이다. 이 문제는 그림 1의 비연속적인 해시 키 공간을 연속적인 해시 키 공간으로 변환시킴으로써 해결될 수 있다.

$$h_{3smh}(i, j, k) = (i \times |I|^2 + j \times |I| + k) \bmod |H_3|.$$
(단, $i < j < k$).
…… 식 ③

3.2 H_3를 위한 사상 완전 해싱 함수

H_3를 위한 사상 완전 해싱의 개념은 그림 1에

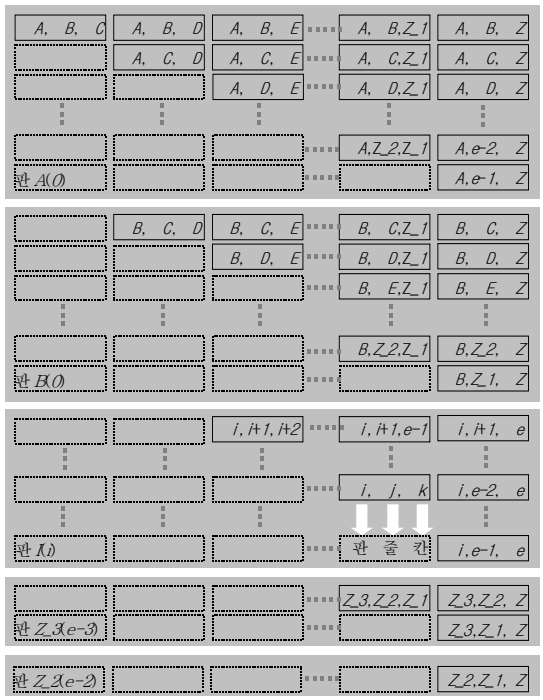


그림 1. H_3를 위한 유효 해시 키
Fig. 1. Valid hash keys for H_3

보인 유효 키들의 위치를 무용 구간 없이 연속되도록 키들의 순서를 재배치할 수 있는 해시 키 변환 방법을 말한다. 즉, 그림 1에서 3 항목으로 조합 가능한 모든 경우의 수 N_{tkeys}는 $|I| \times (|I|-1) \times (|I|-2)$이고, 이중 유효 해시 키에 해당되는 경우의 수 N_{ekeys}는 $N_{tkeys}/6$ 인데, 이들 N_{ekeys}개의 유효 해시 키들을 0~N_{ekeys}-1의 정수로 각각 대응시켜 그 결과에 단순 제산 연산을 적용한다.

3.2.1 유효 해시 키들의 배열 특성

사상 완전 해싱 함수를 도출하기 위해서는 그림 1에서 유효 해시 키들이 이루는 배열의 특성을 인지해야 하는데, 이를 분석하기 위하여 유효 해시 키의 구성 공간을 면, 행, 열로 분류한다. 이를테면 해시 키 조합 〔i, j, k〕(단, $i<j<k$, ij, k 각각은 서로 다른 항목들에게 순서대로 부여된 번호로서 0~ e의 정수, e=|I|-1)에서 i는 면을, j는 행을, 그리고 k는 열을 각각 의미한다.

■ 면 i의 행 j에 존재하는 해시 키 개수

면 i의 행 j에 존재하는 유효 해시 키의 총 개수 L_{ij}는 아래의 식 ④과 같다. 예를 들어, 면 0(=A)의 행 3(=C)에 존재하는 유효 키들의 총 개수는 4(=D)~ e(=Z)까지 계수되므로 그 수는 (e-4)+1=(e+1)-4=|I|-3-1와 같고, 면 1(=B)의 동일 행 3(=C)에 존재하는 유효 키들의 총 개수는 5(=E)~ e(=Z)까지 계수되므로 그 수는 (e-5)+1=(e+1)-5=|I|-4-1와 같아서 두 경우 모두 식 ④를 만족한다.

$$L_{ij} = (|I| - i - 2) - (j - i - 1) = (|I| - j - 1).$$
$$\text{단, } 0 \le i \le |I| - 3,\ i+1 \le j \le |I| - 2. \quad \cdots\cdots \text{식 ④}$$

■ 면 i의 행 i+1에서 행 j까지 존재하는 해시 키 개수

면 i 내에서 행 i+1에서 행 j까지 존재하는 총 유효 해시 키의 개수 TL_{ij}는 식 ④를 활용하여 아래의 식 ⑤와 같이 유도된다.

$$TL_{ij} = \sum_{k=i+1}^{j} L_{ik} = \sum_{k=i+1}^{j} (|I| - k - 1) =$$
$$(j - i)(|I| - 1) - \sum_{k=1}^{j} k + \sum_{k=1}^{i} k =$$
$$(j - i)(|I| - 1) - j(j+1)/2 + i(i+1)/2$$
$$= (i^2 - 7i - j^2 + 7j)/2.$$
$$\text{단, } 0 \le i \le |I| - 3,\ i+1 \le j \le |I| - 2. \quad \cdots\cdots \text{식 ⑤}$$

■ 면 i 안에 존재하는 해시 키 개수

면 i 내의 총 행은 i+1에서 |I|-2까지 존재하므로 면 i 내에 존재하는 전체 유효 해시 키의 개수 P_i는, 식 ④를 활용하여 아래의 식 ⑥과 같이 구할 수 있다.

$$P_i = \sum_{k=i+1}^{|I|-2} L_{ik} = \sum_{k=i+1}^{|I|-2} (|I| - k - 1) =$$
$$(|I| - 2 - i)(|I| - 1) - \sum_{k=1}^{|I|-2} k + \sum_{k=1}^{i} k =$$
$$(|I| - 2 - i)(|I| - 1) -$$
$$(|I| - 2)(|I| - 1)/2 + i(i+1)/2 =$$
$$(|I| - i - 2)(|I| - i - 1)/2.$$
$$\text{단, } 0 \le i \le |I| - 3. \quad \cdots\cdots \text{식 ⑥}$$

■ 면 0에서 면 i까지 존재하는 해시 키 개수

면 0에서 면 i까지 존재하는 총 유효 해시 키의 개수 TP_i는 식 ⑥을 활용하여 아래의 식 ⑦과 같이 얻을 수 있다.

$$TP_i = \sum_{k=0}^{i} P_k = \sum_{k=0}^{i} (|I| - k - 2)(|I| - k - 1)/2 =$$
$$(\sum_{k=1}^{i} k^2 + (3 - 2|I|) \sum_{k=1}^{i} k +$$
$$\sum_{k=0}^{i} (|I|^2 - 3|I| + 2))/2 =$$
$$(i^3 + (12 - 6|I|)i^2 + (6|I|^2 - 24|I| + 22)i +$$
$$6|I|^2 - 18|I| + 12)/6.$$
$$\text{단, } 0 \le i \le |I| - 3. \quad \cdots\cdots \text{식 ⑦}$$

■ 면 i, 행 j의 열 j+1에서 열 k까지 존재하는 해시 키 개수

면 i, 행 j의 열 j+1에서 열 k까지 존재하는 총 유효 해시 키의 개수 TC_{ijk}는 j+1에서 k까지 계수 되므로 명백하게 아래의 식 ⑧과 같다.

$$TC_{ijk} = k - (j+1) - 1 = k - j.$$
$$\text{단, } 0 \le i \le |I| - 3,\ i+1 \le j \le |I| - 2,\ j+1 \le k \le |I| - 1. \quad \cdots\cdots \text{식 ⑧}$$

3.2.2 H_3의 해시 키 변환을 위한 완전 사상 함수

식 ④~⑧을 이용하면 그림 1에 존재하는 유효 해시 키들을 0~N_{ekeys} -1의 정수 공간으로 1:1 대응 시킬 수 있는 함수의 도출이 가능하다. 즉, 임의의 유효 해시 키 〔i, j, k〕에 대응되는 정수 n_{ijk}는 면 i-1까지의 모든 유효 해시 키의

개수, 면 i 내에서 행 i+1에서 행 j-1까지의 모든 유효 해시 키의 개수, 그리고 면 i, 행 j의 열 j+1에서 열 k까지의 모든 유효 키의 개수를 모두 합하면 연속된 순차 공간에서 해당 해시 키 의 유일한 위치를 결정할 수 있다. 이를 위한 완전 사상 함수(perfect mapping function) PMF_{ijk}를 식 ④~⑧을 적용하여 일반항 형태로 나타내면 아래의 식 ⑨과 같다. 식 ⑨의 일반항에서 TP_{-1}과 TL_{ii}는 0으로 정의한다.

$$PMF_{ijk} = TP_{i-1} + TL_{ij-1} + TC_{ijk} - 1.$$

단, $TP_{-1} = 0$, $TL_{ii} = 0$, $0 \le i \le |I|-3$,
$i+1 \le j \le |I|-2$, $j+1 \le k \le |I|-1$.
…… 식 ⑨

3.2.3 H_3를 위한 사상 완전 해싱 함수

식 ⑨의 PMF_{ijk}에 의해 H_3의 유효 키로부터 연속된 정수 공간으로 변환된 결과는 단순 제산 해싱 함수의 해시 키로서 최적의 조건을 가진다. 따라서, H_3를 위한 개선된 해싱 함수로 아래의 식 ⑩에 나타낸 사상 완전 해싱 함수 $h_{3smh}(i,j,k)$를 정의한다.

$$h_{3mph}(i,j,k) = PMF_{ijk} \bmod |H_3| = (TP_{i-1} + TL_{ij-1} + TC_{ijk} - 1) \bmod |H_3|.$$

단, $TP_{-1} = 0$, $TL_{ii} = 0$, $0 \le i \le |I|-3$,
$i+1 \le j \le |I|-2$, $j+1 \le k \le |I|-1$.
…… 식 ⑩

Ⅳ. 최적화된 해싱 메커니즘의 평가

4.1 시험 환경

■ 성능 측정 시스템

제안된 해싱 함수의 성능 측정 시스템으로 표 4 사양의 알파 프로세서 기반의 유닉스 시스템을 사용하였다.

표 4. 시험 시스템의 사양
Table 4. Specifications of test system

항 목	사 양
모 델	COMPAQ WS au600
C P U	Alpha Ev67 667MHz
메 모 리	1GB
운 영 체 제	Digital Tru64UNIX 4.0F

■ 성능 측정 및 비교 방법

주어진 전체 항목 수 및 거래의 크기, 거래와 빈발 항목 집합의 평균 길이에 따라 연관 규칙 탐사 시험용 데이터를 생성하는 프로그램[16]을 사용하여 표 5와 같이 T5I2D100~T20I6D100(T는 거래의 평균 길이, I는 빈발 항목 집합의 평균 길이, D는 거래 전체의 크기, 표 1 참조)의 특성 각각에 대하여 전체 항목 개수 $|I|$가 N1000 ~N9000인 데이터를 생성하고, 이들 모두에 대하여 표 6의 세 가지 형태의 DHP 알고리즘을 각각 적용하여 1/10초 단위의 실행 시간(시스템 및 사용자 모드 실행 시간)을 측정·비교하였다.

표 5. 시험 데이터 유형
Table 5. Types of test data

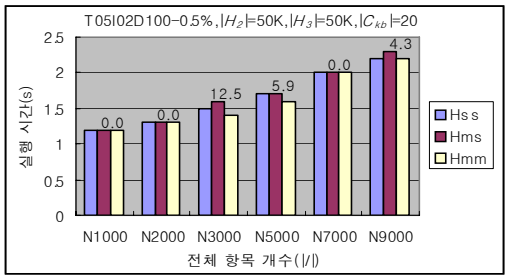
분 류 기 준	분 류 내 용
데이터 특성	T05I02D100, T10I02D100, T10I04D100, T15I04D100, T20I02D100, T20I04D100, T20I06D100
전체 항목 개수($\|I\|$)	N1000, N2000, N3000, N5000, N7000, N9000

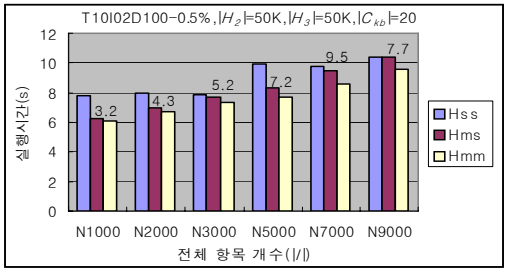
표 6. 시험 알고리즘 유형
Table 6. Type of test algorithm

알고리즘 유형(기호)		해 싱 방 법 적 용 내 용
Hss	$h_{2smh}()+h_{3smh}()$	h_2, h_3 모두 일반적인 방법(식①+식③)
Hms	$h_{2mph}()+h_{3smh}()$	Hss + 참고문헌[14] 방법(식②+식③)
Hmm	$h_{2mph}()+h_{3mph}()$	Hmm + 본 논문 제안 방법(식②+식⑩)

4.2 측정 결과 및 분석

그림 2에 표 5의 모든 시험 데이터 유형에 대하여 측정한 표 6의 알고리즘 유형들의 성능을 비교해 보였다. 그래프 위의 수치는 [14]에서 제안된 해싱 함수 $h_{2mph}(i,j)$가 적용된 알고리즘 유형인





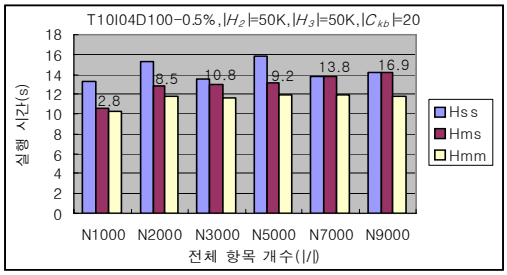
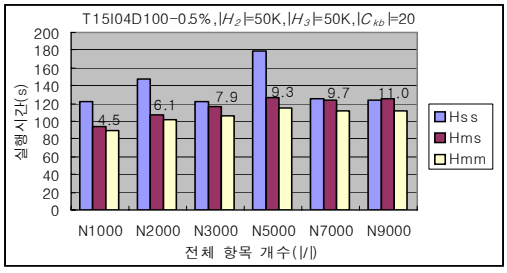
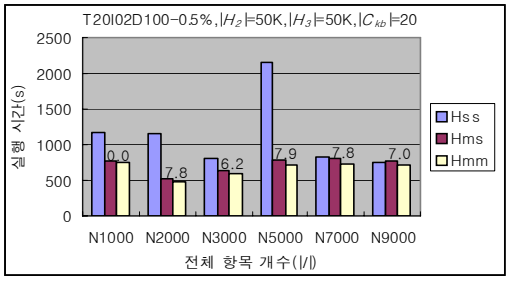
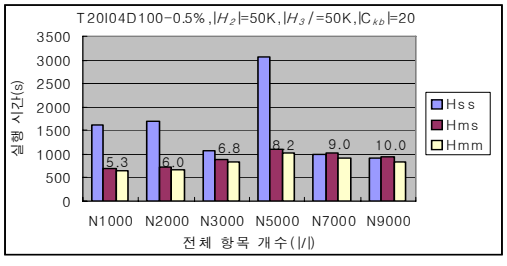


그림 2. 알고리즘 유형별 성능 측정 결과(계속됨)
Fig. 2. Results of algorithm performance test







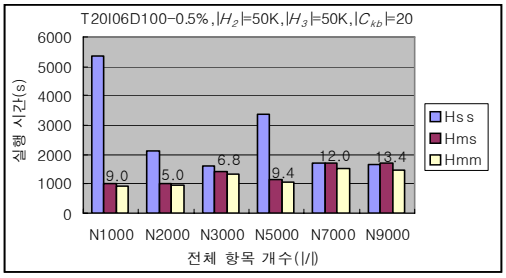


그림 2. 알고리즘 유형별 성능 측정 결과
Fig. 2. Results of algorithm performance test

H_{ms}을 기준으로, 본 논문에서 제안된 해싱 함수 $h_{3smh}(i,j,k)$가 추가로 적용된 알고리즘 유형인 H_{mm}의 성능 향상 비율(%)을 나타낸다.

4.2.1 종합적 성능 특성

그림 2의 측정 결과를 종합적 관점에서 살펴보면 제안된 H_{mm} 알고리즘은 [14]에서 제안된 H_{ms} 알고리즘을 기준으로 실행 시간 성능이 평균 7.3% 이상 개선되었고, 최대 16.9%까지 향상된 경우가 있으며, 42개 데이터 유형 중 4개의 데이터 유형에서는 개선 효과가 나타나지 않았다. 또한, 성능이 감소된 경우는 전혀 관찰되지 않았다.

4.2.2 거래 데이터 유형에 따른 성능 특성

거래 데이터를 특징짓는 요소에는 평균 거래 길이와 평균 빈발 항목 집합의 길이 등 두 가지가 있다.

■ 거래 길이에 따른 성능 특성

그림 3에 평균 거래 길이에 따른 H_{mm}의 성능 개선 추이를 보였는데, 평균 거래 길이가 10 이상으로 커질 때 성능 향상 정도가 높아짐을 보이고 있다.

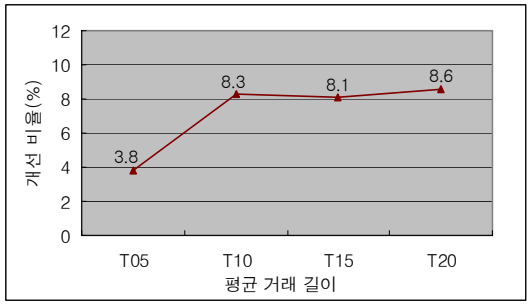


그림 3. 거래 길이에 따른 Hmm 성능 특성
Fig. 3. Trend of Hmm performance in view of transaction length

■ 평균 빈발 항목 집합 크기에 따른 성능 특성

그림 4에 평균 빈발 항목 집합 크기에 따른 H_{mm}의 성능 개선 추이를 보였다. 대체로 평균 거래 길이가 크면 H_{mm}의 성능 향상 정도도 높아짐을 알 수 있다.

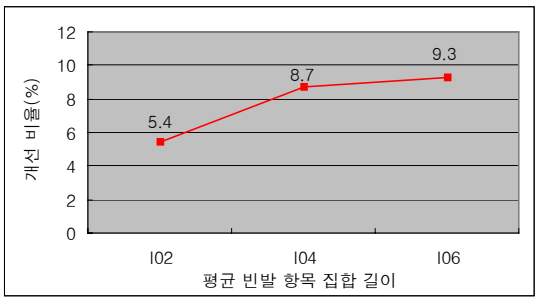


그림 4. 빈발 항목 집합 길이에 따른 성능 특성
Fig. 4. Trend of Hmm performance in view of large itemset length

■ 거래 데이터 유형에 따른 성능 특성

그림 5에는 평균 거래 길이와 평균 빈발 항목 집합 크기를 조합했을 때 나타난 H_{mm}의 성능 개선 추이를 보였다. 이 그림으로부터 평균 거래 길이와 빈발 항목 집합의 크기가 클수록 성능 개선 효과가 높아짐을 알 수 있다.

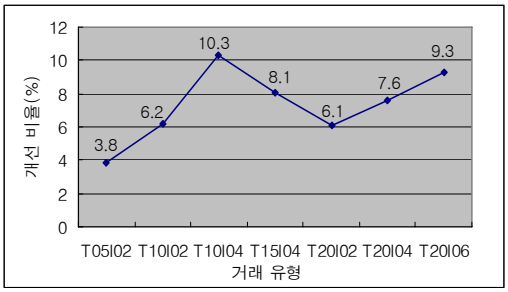


그림 5. 거래 데이터 유형에 따른 Hmm 성능 특성
Fig. 5. Trend of Hmm performance in view of transaction data type

4.2.3 전체 항목 개수에 따른 H_{mm}의 성능 특성

그림 6에 전체 항목 개수에 따른 H_{mm}의 성능 개선 추이를 보였는데, 전체 항목의 개수가 증가할 수록 H_{mm}의 성능 향상 효과도 높아짐을 알 수 있다.

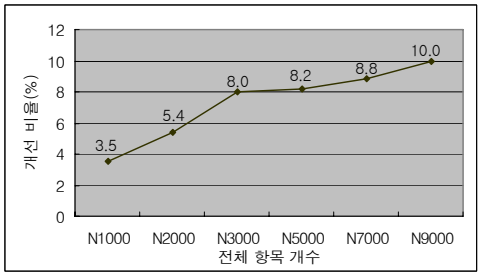


그림 6. 전체 항목 개수에 따른 Hmm 성능 특성
Fig. 6. Trend of Hmm performance in view of total number of items

V. 결론

DHP 연관 규칙 탐사 알고리즘은 단계 k에서 계수 공간으로 사용될 후보 빈발 항목 집합 Ck를 구성할 때, 그 크기를 최대한 감소시키기 위해 단계 k-1에서 k 개의 항목 조합으로 구성된 해시 키들에 대한 계수를 미리 실시하여 전지(pruning)에 활용한다. 이 때, 모든 조합에 대한 계수 공간 마련이 불가능하므로 몇 개의 키 조합들의 중복(충돌)을 허용하는 직접 해시 테이블(direct hash table) H_k를 사용한다. 이 논문에서는 C_3의 전지 비율에 직접적인 영향을 미치는 H_3의 효율을 극대화시키기 위한 해싱 함수로 사상 완전 해싱 함수(mapped perfect hashing function) $h_{3smh}(i, j, k)$를 도출하고, 그 효과를 실험을 통하여 측정하고 분석하였다.

실험 결과, 제안된 해싱 함수는 기존 방법에 대하여 전체 42 종류의 실험 데이터에 대해서 평균 7.3%, 최대 16.9%의 성능 개선 효과를 보였다. 특히, 제안된 해싱 함수는 평균 거래의 길이, 평균 빈발 항목 집합의 크기, 그리고 전체 항목의 개수 등이 클수록 기존 방법 대비 성능 향상 비율이 높아지는 특성을 보여, 탐색 대상이 되는 데이터의 규모가 크고 복잡할 수록 그 효용성이 크게 발휘되는 것으로 나타났다.

참고문헌

[1] 임승환, 권용석, 김상욱, "클러스터링과 특성분석을 이용한 구간 데이터에서 다차원 연관규칙 마이닝," 정보과학회지:컴퓨팅의 실제, 60-64쪽, 2010년. 1월.

[2] 이혜리, 류근호, 김원재, 이건영, "마이크로 어레이 데이터에서 특정 클래스 식별을 위한 이진 연관규칙 추출," 한국지능시스템학회 2009년도 추계 학술발표 논문집, 293-294쪽, 2009년. 12월.

〔3〕 송성렬, 송원문, 김은주, 김명원, "IPTV 환경에서의 점진적 데이터를 위한 효과적인 연관규칙 추출 기법," 한국정보과학회 2009 가을 학술발표논문집 제 36권, 제 2호 (C), 246-251쪽, 2009년. 11월.
〔4〕 이병엽, 박용훈, 유재수, "자동차 산업의 고객 분류 및 타겟 마케팅 모델," 한국컨텐츠학회 논문지, 제 9권 제 4호, 313-322쪽, 2009년. 4월.
〔5〕 권형준, 정동근, 홍광석, "사용자의 재생 시간을 이용한 멀티미디어 추천 시스템," 인터넷정보학회 논문지, 제 10권, 제 1호, 111-121쪽, 2009년 2월.
〔6〕 임영희, 이종욱, 박대희. 장진경, "연관 규칙 마이닝을 이용한 한국 신노년층의 생활 만족도에 관한 연구," 한국가정관리학회 2008년 추계학술대회, 164-173쪽, 2008년 11월.
〔7〕 M. Schrader, D. Vlamis, M. Nader, C. Claterbos, D. Collins, M. Campbel, F. Conrad, "Oracle Essbase & Oracle OLAP," McGraw-Hill, Oct. 2009.
〔8〕 Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten, "The WEKA Data Mining Software: An Update," SIGKDD Explorat- ions, Volume 11, Issue 1, 2009. *http://www.cs.waikato.ac.nz/~ml/weka/*
〔9〕 J. S. Park, M.-S. Chen and P. S. Yu, "An Effective Hash-Based Algorithm for Mining Association Rules," Proceedings of ACM SIGMOD, pp. 175-186, 1995.
〔10〕 J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," Proceedings ACM SIGMOD Int'l Conf. Management of Data(SIGMOD' 00), pp. 1-12, May 2000.
〔11〕 이형봉, 김진호, "FP-tree와 DHP 연관 규칙 탐사 알고리즘의 실험적 성능 비교," 정보과학회논문지:데이터베이스, 제 35권, 제 3호, 341-351쪽, 2008년. 6월,
〔12〕 R. Agrawal, T. Imielinski and A. Swami, "Mining Association Rules between Sets of Items in Large Databases," Proceedings of ACM SIGMOD on Management of Data, pp. 207-216, 1993.
〔13〕 R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," Proceedings of the 20th Inte- rnational Conference on Very Large Databases, pp. 487-499, 1994.
〔14〕 이형봉, "완전 해싱을 위한 DHP 연관 규칙탐사 알고리즘의 개선 방안," 정보과학회논문지:데이터베이스, 제 31권, 제 2호, 91-98쪽, 2004년. 4월.
〔15〕 이형봉, "DHP 연관 규칙 탐사 알고리즘을 위한 효율적인 해싱 메커니즘," 정보처리학회 논문지(D), 제 13-D권, 제 5호, 651-660쪽, 2006년. 10월.
〔16〕 R. Agrawal and et al, "Synthetic Data Generation Code for Associations and Sequential Patterns," *http://www.almaden.ibm.com/cs/projects/iis/hdb/ Projects/data_mining/mining.shtml, 1999.*

저 자 소 개

이 형 봉

1984년: 서울대학교 계산통계학과 학사.
1986년: 서울대학교 계산통계학(전산과학)과 석사.
2002년: 강원대학교 컴퓨터과학과 박사.
1986년~1993년: LG전자 컴퓨터연구소 선임 연구원.
1994년~1999년: 한국디지탈㈜ 책임컨설턴트.
1999년~2004년: 호남대학교 정보통신공학부 조교수.
2004년~현재: 강릉원주대학교 컴퓨터공학과 부교수.
관심분야 : 임베디드 시스템, 센서네트워크, 데이터 마이닝 알고리즘

권 기 현

1993년: 2월 강원대학교 전자계산학과 학사.
1995년: 2월 강원대학교 대학원 전자계산학과 석사.
2000년: 2월 강원대학교 대학원 컴퓨터과학과 박사.
1998년~2002년: 동원대학 인터넷정보과 교수.
2002년~현재: 강원대학교 전자정보통신공학부 부교수.
관심분야: 분산 시스템, 미들웨어, 임베디드 소프트웨어 등