ISSN 1598-0170 (Print)
ISSN 2287-1136 (Online)
http://www.jksii.or.kr

상위 K 하이 유틸리티 패턴 마이닝 기법 성능분석☆

Performance Analysis of Top-K High Utility Pattern Mining Methods

양 흥 모[1] 윤 은 일[1*] 김 철 홍[2]
Heungmo Ryang Unil Yun Chulhong Kim

요 약

전통적인 빈발 패턴 마이닝은 데이터베이스로부터 사용자 정의 최소 임계치 이상의 빈도수를 가지는 유효 패턴들을 식별한다. 적절한 임계치 설정은 해당 도메인에 대한 사전 지식을 요구하므로 쉬운 작업이 아니다. 따라서 임계치 설정을 통한 마이닝 결과의 정밀한 제어 불가능으로 인해 도메인 지식을 기반으로 하지 않는 패턴 마이닝 방법이 필요하게 되었다. 상위 K 빈발 패턴 마이닝은 이러한 문제를 해결하기 위해 제안되었으며, 임계치 설정 없이 상위 K개의 중요 패턴들을 마이닝 한다. 사용자는 이를 적용함으로써 데이터베이스에 상관없이 가장 높은 빈도수의 패턴부터 K번째로 높은 빈도수의 패턴까지 찾아낼 수 있다. 비록 상위 K 빈발 패턴 마이닝이 임계치 설정 없이 상위 K개의 중요 패턴들을 마이닝 하지만, 트랜잭션 내 아이템 수량과 데이터베이스 내 서로 다른 아이템 중요도를 고려하지 못하여 많은 실세계 응용의 요구에 부합하지 못한다. 하이 유틸리티 패턴 마이닝은 아이템 중요도가 포함된 비 바이너리 데이터베이스의 특성을 고려하기 위해 제안되었으나 최소 임계치를 필요로 한다. 최근 임계치 설정 없는 하이 유틸리티 패턴 마이닝을 위한 상위 K 하이 유틸리티 패턴 마이닝이 개발되었으며, 이를 통해 사용자는 사전 지식 없이 원하는 수의 패턴을 마이닝 할 수 있다. 본 논문은 상위 K 하이 유틸리티 패턴 마이닝을 위한 알고리즘을 분석한다. 최신 알고리즘에 대한 성능분석을 통해 개선사항 및 발전 방향에 대해 고찰한다.

☞ 주제어 : 하이 유틸리티 패턴, 상위 K 마이닝, 임계치 설정, 하이 유틸리티 패턴 마이닝, 상위 K 하이 유틸리티 패턴 마이닝, 성능분석

ABSTRACT

Traditional frequent pattern mining discovers valid patterns with no smaller frequency than a user-defined minimum threshold from databases. In this framework, an enormous number of patterns may be extracted by a too low threshold, which makes result analysis difficult, and a too high one may generate no valid pattern. Setting an appropriate threshold is not an easy task since it requires the prior knowledge for its domain. Therefore, a pattern mining approach that is not based on the domain knowledge became needed due to inability of the framework to predict and control mining results precisely according to the given threshold. Top-*k* frequent pattern mining was proposed to solve the problem, and it mines top-*k* important patterns without any threshold setting. Through this method, users can find patterns from ones with the highest frequency to ones with the *k*-th highest frequency regardless of databases. In this paper, we provide knowledge both on frequent and top-k pattern mining. Although top-*k* frequent pattern mining extracts top-*k* significant patterns without the setting, it cannot consider both item quantities in transactions and relative importance of items in databases, and this is why the method cannot meet requirements of many real-world applications. That is, patterns with low frequency can be meaningful, and vice versa, in the applications. High utility pattern mining was proposed to reflect the characteristics of non-binary databases and requires a minimum threshold. Recently, top-*k* high utility pattern mining has been developed, through which users can mine the desired number of high utility patterns without the prior knowledge. In this paper, we analyze two algorithms related to top-*k* high utility pattern mining in detail. We also conduct various experiments for the algorithms on real datasets and study improvement point and development direction of top-*k* high utility pattern mining through performance analysis with respect to the experimental results.

☞ keyword : High utility patterns, Top-K mining, Threshold setting, High utility pattern mining, Top-K high utility pattern mining, Performance analysis

[1] Dept. of Computer Engineering, Sejong University, Seoul, 143-747, Korea
[2] Electronics and Telecommunication Research Institute, Daejeon, 305-700, Korea
* Corresponding author (yunei@sejong.ac.kr)
[Received 25 August 2015, Reviewed 27 August 2015, Accepted 7 November 2015]
☆ 본 연구는 2015년도 정부 교육과학기술부의 재원으로 한국 연구재단(NRF)의 지원을 받아 수행된 연구 사업이며(NRF No. 20152062051, NRF No. 20155054624 and NRF No. 20135005682), 중소기업청에서 지원하는 2015년도 산학연협력 기업부설연구소 지원사업(No. C0261068)의 연구수행으로 인한 결과물이고, 국토교통과학기술진흥원의 철도기술연구사업으로 지원된『기존 안전검지장치 및 현장 운영데이터 기반 실시간 철도안전 통합 감시제어시스템 개발』 연구의 2세부 과제인 「실시간 철도안전 의사결정 지원시스템 개발」과제(No. 15RTRP-B082515-02)의 지원으로 수행되었음.
☆ 본 논문은 2015년도 인터넷정보학회 춘계학술발표대회 우수논문 추천에 따라 확장 및 수정된 논문임.

1. 서 론

데이터베이스 내 지식 발견 (Knowledge Discovery in Databases; KDD) 과정의 분석 단계인 데이터 마이닝은 거대한 데이터베이스에 숨겨져 있는 유용한 정보를 찾아낸다. 패턴 마이닝 [1, 2]은 데이터 마이닝을 위한 기법 중에 하나로서 데이터베이스로부터 패턴 형태의 중요 정보를 추출한다. 패턴 마이닝의 근본적인 연구 주제인 빈발 패턴 마이닝 [3, 4]은 사용자 정의 최소 빈도수 이상으로 빈발한 모든 패턴들을 추출한다. 비록 빈발 패턴 마이닝이 데이터 마이닝 분야에서 중요한 역할을 수행해 왔지만, 아이템 발생이 0 또는 1로 표현된 이진 데이터베이스 내 모든 아이템을 동일한 중요도로 고려한다. 실제 응용의 비 이진 데이터베이스에서는 빈발하지 않은 패턴들이 더 유용할 수 있으며, 빈발 패턴 마이닝은 이러한 패턴들을 마이닝 하지 못한다. 최근 아이템 중요도를 기반으로 이진 데이터베이스로부터 중요 패턴들을 마이닝 하기 위한 하이 유틸리티 패턴 마이닝 [5, 6]이 중요한 주제로서 연구되고 있다.

전통적인 패턴 마이닝은 사용자로부터 주어진 최소 임계치 이상의 빈도수 또는 유틸리티 같은 기준 값을 지닌 패턴들을 찾아낸다. 따라서 마이닝 결과는 사용자 정의 임계치에 좌우되며, 낮은 임계치는 많은 수의 패턴 추출을 야기함으로써 결과 분석을 어렵게 하고, 높은 임계치에 의해 유용한 패턴이 추출되지 않을 수 있다. 즉, 적절한 임계치 설정은 도메인 지식을 필요로 하며, 마이닝 결과의 정밀한 제어 불가능으로 인해 쉬운 작업이 아니다. 상위 K 마이닝 [7, 8]은 이러한 문제를 해결하기 위해 제안되었으며, 최소 임계치 설정 없이 상위 K개의 유효 패턴들을 마이닝 한다. 본 논문은 실세계 데이터베이스 특성을 반영하기 위한 하이 유틸리티 패턴 마이닝의 상위 K 그리고 최소 임계치 기반의 알고리즘들에 대한 성능분석을 통해 상위 K 하이 유틸리티 패턴 마이닝의 개선사항 및 발전 방향에 대해 고찰한다.

2. 관련 연구

2.1. 최소 임계치 기반의 패턴 마이닝

$I = \{i_1, i_2, ..., i_m\}$를 서로 구별되는 m개의 아이템 집합 그리고 $D = \{T_1, T_2, ..., T_n\}$를 n개의 트랜잭션 집합이라고 하면, I의 부분 집합인 각 트랜잭션 $T_i = \{i_1, i_2, ..., i_l\}$ ($T_i \supseteq I \wedge 1 \le i \le n \wedge 1 \le l \le m$)는 식별자를 통해 구별된다. 또한, k개의 아이템으로 구성된 패턴 $X_k = \{i_1, i_2, ..., i_k\}$ ($X_k \supseteq I \wedge 1 \le k \le m$)를 길이가 k인 *k-itemset*이라 한다.

정의 1. 최소 임계치. 전체 데이터베이스 D의 기준 값을 $V(D)$ 그리고 사용자 정의 임계치 비를 δ라고 하면, 최소 임계치 min은 다음 식 (1)과 같이 정의된다.

$$min = V(D) \times \delta \tag{1}$$

$V(D)$는 데이터베이스 스캔을 통해 계산되며, 빈발 패턴 마이닝 [3, 9]에서는 전체 트랜잭션 개수 그리고 하이 유틸리티 패턴 마이닝 [5, 6]에서는 데이터베이스의 전체 유틸리티 합을 의미한다.

정의 2. 안티 모노톤 속성. X_k를 유효하지 않은 길이가 k인 패턴, X_s를 임의의 길이가 k+1 이상인 X_k의 상위 패턴 (super pattern)이라고 하면, X_k의 모든 상위 패턴은 다음 식 (2)의 속성을 만족한다.

$$min > V(X_k) \rightarrow \forall X_s \supset X_k \wedge min > V(X_s) \tag{2}$$

즉, 안티 모노톤 속성 (anti-monotone property) [10]을 기반으로 마이닝 과정에서 유효하지 않은 패턴 Xk가 발견되면, Xk의 상위 패턴들에 대한 검색 공간을 탐색하지 않고 삭제함으로써 마이닝 연산이 효율적으로 수행된다. 따라서 마이닝 과정에서 상기 속성을 만족시키는 것은 성능 측면의 중요한 요소이다.

2.2. 상위 K 기반의 패턴 마이닝

상위 K 기반의 패턴 마이닝 [7, 8, 11]은 초기 최소 임계치를 0으로 설정하며, 마이닝 과정에서 효과적으로 임계치를 상승시킨다. 상기 기법은 먼저 데이터베이스 스캔을 통해 각 길이가 1인 패턴, X_1, 의 기준 값, $V(X_1)$, 을 계산한다. 즉, 모든 아이템에 대한 기준 값들, $\{V(\{i_1\}), V(\{i_2\}), ..., V(\{i_m\})\}$, 이 계산된 후 내림차순 또는 오름차순으로 정렬되며, 가장 큰 기준 값부터 K번째 기준 값까지 리스트 $L = \{V_1, V_2, ..., V_k\}$ ($V_1 \ge V_2 \ge ... \ge V_k$) 에 저장된다. 이때, 동일한 기준 값들이 존재하면, K번째로 높은 값을 계산할 수 있도록 모두 삽입한다. 즉, L에는 K개 이상의 기준 값들 또는 동일한 값들이 존재할 수 있다.

L이 생성되면 최소 임계치 min은 리스트 내 가장 작은 기준 값인 V_k로 설정되어 마이닝 과정이 진행되며, 해당 과정에서 min 이상의 기준 값을 가지는 패턴, $V(X) \ge min$, 이 발견될 때마다 X를 후보 패턴으로 추출하고, $V(X)$를 L 내의 가장 작은 기준 값 V_k와 비교한다. 만약 X의 기준 값

이 V_k보다 크면, $V(X)>V_k$, L에 $V(X)$가 추가되고, L에 최소 K개의 기준 값들이 남을 수 있도록 가장 작은 값들이 제거된다. 기준 값 리스트가 갱신될 때마다 최소 임계치는 리스트 내 가장 작은 기준 값으로 변경된다. 마이닝 과정 완료 후, 추출된 후보 패턴들 중 상위 K 기준 값들을 가지는 패턴들이 실제 패턴들로 생성된다.

상위 K 기반의 하이 유틸리티 패턴 마이닝 [11]은 전통적인 패턴 마이닝 [3, 10]과는 달리 빈도수가 아닌 유틸리티를 기준으로 하며, 최소 임계치 기반 하이 유틸리티 패턴 마이닝 [5]과는 달리 임계치 설정이 필요하지 않다. 또한, 빈도수 기반 상위 K 패턴 마이닝 [7, 8]가 마이닝 과정에서 생성된 후보 패턴의 빈도수 정보를 임계치 상승 과정에 바로 적용할 수 있는 것과는 달리, 실제 유틸리티 정보를 적용하기 어렵다는 특성을 가진다.

3. 상위 K 하이 유틸리티 패턴 마이닝 기법

하이 유틸리티 패턴 마이닝 [5, 6]에서, 데이터베이스 내 각 아이템 i_p ($i_q \in I \wedge 1 \le p \le m$)은 판매 이윤과 같은 외부 유틸리티라 불리는 상대적인 중요도 $eu(i_p)$를 가지며, 각 트랜잭션 $T_i = \{i_1, i_2, ..., i_l\}$ ($T_i \supseteq I \wedge 1 \le i \le n \wedge 1 \le l \le m$)는 수량과 같은 내부 유틸리티라 하는 비 이진 값 $iu(i_q, T_i)$ ($i_q \in T_i \wedge 1 \le q \le l$)의 아이템들로 구성된다.

정의 3. 아이템 유틸리티. i_q를 비 이진 트랜잭션 T_i 내 임의의 아이템이라고 하면, T_i에 대한 i_p의 유틸리티, $u(i_q, T_i)$, 는 다음 식 (3)의 수식에 의해 계산된다.

$$u(i_q, T_i) = eu(i_q) \times iu(i_q, T_i) \quad (3)$$

트랜잭션 내 모든 아이템은 유틸리티를 가지며, 마이닝 과정에서 트랜잭션 및 패턴 유틸리티를 계산하는 데 사용된다.

정의 4. 트랜잭션 유틸리티. T_i를 l개의 아이템으로 구성된 비 이진 트랜잭션이라고 하면, 데이터베이스 내 T_i의 유틸리티, $tu(T_i)$, 는 다음 식 (4)와 같이 계산된다.

$$tu(T_i) = \sum_{i_q \in T_i} u(i_q, T_i) \quad (4)$$

정의 5. 데이터베이스 유틸리티. 데이터베이스 D는 n개의 트랜잭션들로 구성되므로 다음 식 (5)와 같이 모든 트랜잭션 유틸리티 합을 통해 데이터베이스 유틸리티, $V(D)$, 가 도출된다.

$$V(D) = \sum_{T_i \in D} tu(T_i) \quad (5)$$

정의 6. 패턴 유틸리티. 길이가 k인 패턴 X_k의 데이터베이스 내 유틸리티, $u(X_k)$, 는 다음 식 (6)의 수식을 통해 정의된다.

$$u(X_k) = \sum_{X_k \subseteq T_i \in D} \sum_{i_q \in X_k} u(i_q, T_i) \quad (6)$$

만약 X_k의 유틸리티가 최소 유틸리티 임계치, *minutil* = $minutil = V(D) \times \delta$, 보다 작지 않으면, 해당 패턴을 하이 유틸리티 패턴이라 한다. 하이 유틸리티 패턴 마이닝에서 안티 모노톤 속성 [10]은 각 패턴 X_k의 유틸리티 상한 (upper-bound)를 의미하는 과추정 유틸리티 $twu(X_k)$에 의해 유지되며, X_k를 포함하는 모든 트랜잭션의 유틸리티 합, $\sum tu(T_i)$ ($X_k \subseteq T_i \in D$), 으로 정의된다.

TKU (*Top-K Utility itemset mining*) [11]는 상위 K 하이 유틸리티 패턴 마이닝 알고리즘으로서 비 이진 데이터베이스로부터 아이템 중요도를 고려하여 최소 유틸리티 임계치 설정 없이 하이 유틸리티 패턴들을 마이닝 한다. 이를 위해 TKU는 초기 *minutil*을 0으로 설정하고, UP-Tree라 하는 트리를 사용하는 최소 임계치 알고리즘인 UP-Growth [5]를 기반으로 마이닝 과정을 수행한다. TKU는 먼저 첫 번째 데이터베이스 스캔을 통해 길이가 2인 패턴들의 유틸리티 하한 (lower-bound)을 계산하고, K번째 하한 값으로 *minutil*을 설정한다 (PE 전략). 다음 데이터베이스 스캔에서, 해당 *minutil*보다 작은 과추정 유틸리티 값을 가지는 아이템들을 제거하고, 전역 UP-Tree를 구축하며, NU 그리고 MD라 하는 두 가지 전략을 적용하여 최소 유틸리티 임계치를 증가시킨다. 다음으로 증가된 *minutil*를 기준으로 UP-Growth를 적용하여 마이닝을 수행하며, MC 전략을 통해 최소 임계치를 지속적으로 증가시킨다. Phase I이라 하는 후보 생성 과정이 완료되면, TKU는 다음 과정인 Phase II에서 생성된 후보 패턴들을 과추정 유틸리티 내림차순으로 정렬시키고, 각 후보 패턴의 실제 유틸리티 값을 계산하며, SE 전략을 통해 *minutil*을 증가시킨다. 즉, TKU는 SE 전략을 기반으로 Phase II에서 최종 상위 K개의 하이 유틸리티 패턴들을 식별한다.

TKU는 트리 기반 하이 유틸리티 패턴 마이닝 알고리즘들 중에서 가장 성능이 뛰어난 UP-Growth를 기반으로 하므로 변화하는 임계치에 대하여 적은 수의 후보 패턴만을 생성하는 장점이 있다. 또한, 마이닝 과정에서 다양한 상승 전략들을 적용함으로써 가능한 많은 패턴 정보를 이용하여 임계치를 상승시키는 장점을 지닌다.

4. 상위 K 및 최소 임계치 기반 기법들의 성능 분석

4.1. 실험 환경

본 논문은 상위 K 기반 하이 유틸리티 패턴 마이닝을 위한 TKU 알고리즘 [11]의 성능을 최소 임계치 기반 하이 유틸리티 패턴 마이닝 알고리즘인 UP-Growth [5]에 대해 비교 분석한다. 모든 알고리즘은 C++로 구현되었으며, 4.0GHz 인텔 프로세서 및 16GB 메모리 환경의 윈도우즈 7 운영체제에서 실행되었다. 또한, 성능분석 실험에는 NU-MineBench 2.0 [12]의 유틸리티 정보가 포함된 마켓 데이터 집합인 Chain-store와 버섯 품종 관련 정보가 포함된 데이터 집합인 Mushroom이 사용되었다. 본 실험을 위해 Mushroom 데이터 집합에 대하여 1과 10사이의 정수 내부 유틸리티 그리고 0과 1사이의 실수 외부 유틸리티가 생성되어 포함되었다. 표 1은 본 실험에서 사용된 상기 데이터 집합들의 특성을 보여주며, $|D|$는 트랜잭션 개수, $|I|$는 아이템 개수, T_{avg}는 평균 트랜잭션 길이, 그리고 *Size*는 데이터 집합의 실제 크기를 의미한다.

(표 1) 실제 데이터 집합의 특성
(Table 1) Characteristics of Real Datasets

Dataset	|D|	|I|	T_{avg}	Size (MB)
Chain-store	1,112,949	468	7.2	60.60
Mushroom	8,124	119	23.0	0.92

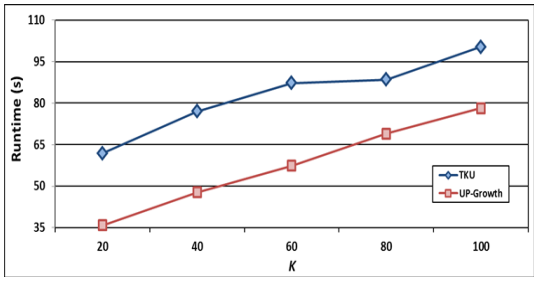
성능분석 실험은 20부터 100까지 변화하는 *K*에 대한 실행 속도, 메모리 사용량, 그리고 생성 및 검증 후보 패턴 수 측면에서 수행되었다. 더욱이, UP-Growth의 최소 유틸리티 임계치는 최적화되어 TKU와 동일한 수의 패턴들을 마이닝 할 수 있도록 표 2와 같이 설정되었다.

(표 2) UP-Growth의 최적화 임계치
(Table 2) Optimal thresholds of UP-Growth

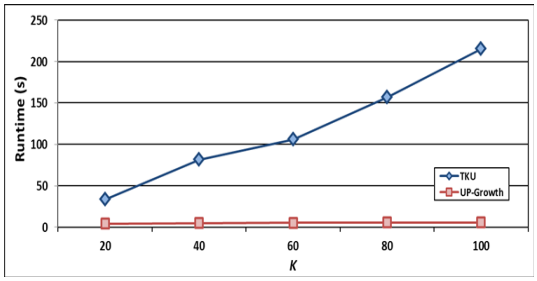
K	Optimal Threshold	
	Chain-store	Mushroom
20	0.2310%	15.9094%
40	0.1620%	15.4705%
60	0.1236%	15.2671%
80	0.1012%	15.0839%
100	0.0875%	14.9202%

4.2. 실행시간 성능분석

그림 1과 2는 20부터 100까지 변화하는 *K*에 따른 비교 알고리즘들의 두 실제 데이터 집합들, Chain-store 그리고 Mushroom, 을 사용한 실행시간 측면에 대한 성능평가 실험 결과이다. 실험 결과에서, TKU는 *K*가 증가함에 따라 더 많은 하이 유틸리티 패턴들을 생성하기 위한 연산을 수행함으로써 실행 속도가 느려지는 결과를 야기하였다. UP-Growth 또한 동일한 수의 패턴들을 마이닝 하기 위해 최소 유틸리티 임계치가 감소됨으로써 더 많은 실행 시간을 필요로 하였으며, 모든 *K*에 대해 TKU보다 더욱 빠른 속도로 실행되었다. 그 이유는 UP-Growth가 상위 K 패턴들을 마이닝 하기 위한 최소 임계치를 초기에 설정한 반면에, TKU는 초기에 0으로 설정하여 더 많은 검색 공간을 탐색했기 때문이다. 실제 응용에서는 그러나 원하는 수의 패턴들을 생성하기 위한 임계치 설정이 쉽지 않으며, 이를 찾기 위한 여러 번의 실행 과정이 필요하다. 즉, 실제 응용에서는 최적화 임계치를 알 수 없으므로 실험 결과보다 더 많은 실행시간을 필요로 한다. 따라서 실제 응용에 적용 시 사용자가 원하는 *K*개의 패턴들을 마이닝 하는 데 TKU보다 UP-Growth가 더 느릴 수 있다.



(그림 1) 실행시간 결과 (Chain-store)
(Figure 1) Runtime result (Chain-store)

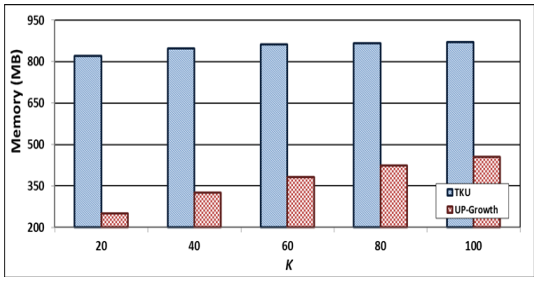


(그림 2) 실행시간 결과 (Mushroom)
(Figure 2) Runtime result (Mushroom)

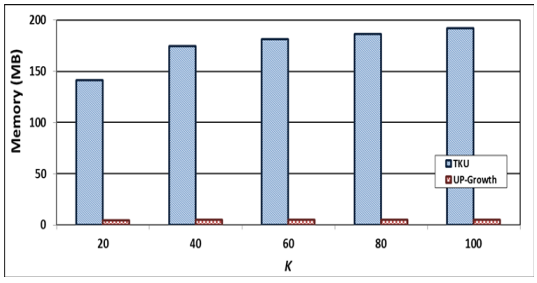
한편, UP-Growth의 실행시간이 Chain-store를 사용한 결과보다 Mushroom을 사용했을 때 더 빠르게 나왔으며, 그 결과 TKU와의 성능 차이가 Mushroom에서 더 크게 벌어졌다. 그 이유는 Mushroom 데이터 집합이 Chain-store에 비해 더욱 밀집한 특성을 보이고, 이로 인해 마이닝 과정에서 더 많은 후보 패턴들이 생성되어 처리하는 데 더 많은 실행시간을 소모했기 때문이다.

4.3. 메모리 사용량 성능분석

그림 3과 4는 Chain-store 그리고 Mushroom에 대해 *K*를 20부터 100까지 변화시킨 성능평가 실험의 메모리 사용량 측면에 대한 결과이다. 두 비교 알고리즘들, TKU 그리고 UP-Growth, 은 실험 결과에서 *K* 값의 증가에 따라 메모리 사용량 또한 증가하였다. 더욱이, TKU는 UP-Growth보다 훨씬 많은 메모리 자원을 필요로 하였는데, 이는 비록 PE, NU, 그리고 MD 전략들에 의해 초기 임계치가 증가하였지만, 표 2의 UP-Growth에서 사용된 초기 임계치보다 낮게 설정되어 더 많은 정보를 자료 구조에 저장해야 했기 때문이다.



(그림 3) 메모리 사용량 결과 (Chain-store)
(Figure 3) Memory usage result (Chain-store)



(그림 4) 메모리 사용량 결과 (Mushroom)
(Figure 4) Memory usage result (Mushroom)

그림 1과 2의 실행시간 실험 결과에서는 Mushroom의 크기가 Chain-store보다 작음에도 *K*의 증가에 따라 TKU의 실행 속도가 더 느려졌지만, 그림 3과 4의 메모리 사용량 측면에 대한 결과에서는 Mushroom와 비교했을 때 TKU는 Chain-store에서 동일한 수의 패턴들을 마이닝 하기 위해 더 많은 메모리 자원을 사용하였다. 그 이유는 더 많은 트랜잭션 및 아이템 정보가 Chain-store에 포함돼 있으며, 마이닝을 위해 해당 정보를 모두 자료 구조에 저장했기 때문이다.

4.4. 후보 패턴 성능분석

표 3과 4는 비교 알고리즘들, TKU 그리고 UP-Growth, 이 실제 데이터 집합들을 사용한 마이닝 과정에서 생성한 후보 패턴 개수 그리고 상위 K개의 하이 유틸리티 패턴들을 식별하기 위해 검증한 후보 패턴 개수 결과이다. 표 3의 Chain-store에 대한 실험 결과에서, TKU는 100보다 작은 *K* 값에 대해 마이닝 과정에서 UP-Growth보다 최소 임계치를 더욱 상승시킴으로써 더 적은 수의 후보 패턴들을 생성하였다. 또한, SE 전략이 적용된 TKU가 그렇지 않은 UP-Growth보다 훨씬 더 적은 후보 패턴 검증 작업을 수행하였다.

(표 3) 생성 및 검증 후보 패턴 개수 (Chain-store)
(Table 3) Number of generated and checked candidate patterns (Chain-store)

K	TKU		UP-Growth	
	#Candidate Patterns	#Valid Patterns	#Candidate Patterns	#Valid Patterns
20	612	20	1,273	1,273
40	1,534	40	2,118	2,118
60	2,507	60	2,994	2,994
80	3,347	80	3,814	3,814
100	4,908	100	4,513	4,513

(표 4) 생성 및 검증 후보 패턴 개수 (Mushroom)
(Table 4) Number of generated and checked candidate patterns (Mushroom)

K	TKU		UP-Growth	
	#Candidate Patterns	#Valid Patterns	#Candidate Patterns	#Valid Patterns
20	4,527,073	20	23,532	23,532
40	13,545,656	40	28,309	28,309
60	18,611,795	60	30,240	30,240
80	27,374,872	80	31,965	31,965
100	39,780,745	100	33,733	33,733

표 3의 Chain-store 데이터 집합을 사용한 실험 결과와는 달리 표 4의 Mushroom 데이터 집합에 대한 결과에서는 모든 K에 대하여 두 알고리즘들은 더 많은 후보 패턴들을 생성하였다. 이는 Mushroom 내 트랜잭션 밀집도가 Chain-store보다 더 높아 전체적으로 과추정 패턴 유틸리티가 증가했기 때문이다. 더욱이, TKU는 초기에 0으로 설정된 임계치로 인해 UP-Growth보다 훨씬 더 많은 후보 패턴들을 생성하였으나, K만큼의 하이 유틸리티 패턴들을 마이닝 하기 위한 검증 횟수는 더 적었다.

전체적으로 TKU가 UP-Growth보다 실행시간 및 메모리 사용량 성능이 떨어지지만, 이는 성능평가 실험에서 UP-Growth가 동일한 패턴 집합을 마이닝 하기 위한 최적의 임계치를 사전에 습득 및 설정했기 때문이다. 또한, TKU는 많은 후보 패턴들을 생성했지만, K와 동일한 수의 후보 패턴들에 대해서만 검증 작업을 수행하였다. 특히, TKU는 밀집도가 높은 Mushroom 데이터 집합에 대해 최악의 성능을 보였으며, 이는 수많은 후보 패턴들을 생성했기 때문이다. 따라서 이를 개선하기 위해서는 후보 패턴 생성이 없는 기법의 개발 및 적용이 필요하다.

5. 결 론

최소 임계치 기반의 하이 유틸리티 패턴 마이닝 알고리즘, UP-Growth, 에 대한 상위 K 기반의 알고리즘, TKU, 의 성능분석 결과는 최적 임계치에 대한 사전 정보를 습득 가능할 때 최소 임계치 기반 기법이 더 나은 성능을 보일 수 있음을 보여주며, 실험에서 사용된 최소 임계치를 통해 이는 어려운 작업임을 알 수 있다. 따라서 앞으로 발전하기 위해서는 마이닝 과정에서 최소 임계치 값을 더욱 효과적으로 높이는 방법이 필요함을 도출할 수 있다. TKU는 마이닝 과정에서 PE, NU, MD, 그리고 MC 전략들을 사용하여 임계치 값을 상승시키며, 효과적인 임계치 상승을 위해서는 더 많은 패턴들에 대한 유틸리티 값들을 도출할 수 있어야 한다. 또한, TKU는 데이터 집합의 밀집도가 높을수록 실행시간, 메모리 사용량, 후보 패턴 수 측면의 성능이 떨어지는 현상을 보였으나, 적은 수의 후보 패턴들만을 검증 작업에 사용하였으므로 마이닝 과정에서 생성되는 후보 패턴 수를 감소시키는 방법은 좋은 연구 주제가 될 수 있음을 예상할 수 있다. 비록 성능평가 결과 UP-Growth가 TKU보다 더 나은 성능을 보였으나, 이는 적절한 임계치가 설정되는 최적의 상황을 가정한 것이므로 실제 환경에 대한 적용에서는 적절한 임계치를 찾기 위한 UP-Growth의 반복 실행으로 TKU의 적용이 더 나은 결과를 보일 것임을 도출할 수 있다.

참 고 문 헌 (Reference)

[1] G. Lee and U. Yun, "Analysis and Performance Evaluation of Pattern Condensing Techniques used in Representative Pattern Mining", Journal of Internet Computing and Services, Vol. 16, No. 2, pp. 77-83, 2015. http://dx.doi.org/10.7472/jksii.2015.16.2.77

[2] G. Pyun and U. Yun, "Performance evaluation of approximate pattern mining based on probabilistic technique", Journal of Internet Computing and Services, Vol. 14, No. 1, pp. 63-69, 2013. http://dx.doi.org/10.7472/jksii.2013.14.63

[3] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without Candidate Generation: A frequent-Pattern Tree Approach", Data Mining and Knowledge Discovery, Vol.8, No.1, pp.53-87, 2004. http://dx.doi.org/10.1023/B:DAMI.0000005258.31418.83

[4] U. Yun and G. Lee, "A Weighted Frequent Graph Pattern Mining Approach considering Length-Decreasing Support Constraints", Journal of Internet Computing and Services, Vol. 15, No. 6, pp. 125-132, 2014. http://dx.doi.org/10.7472/jksii.2014.15.6.125

[5] V.S. Tseng, B.-E. Shie, C.-W. Wu, and P.S. Yu, "Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases", IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 8, 2013, pp. 1772-1786. http://dx.doi.org/10.1109/TKDE.2012.59

[6] U. Yun, H. Ryang, and K. Ryu, "High utility itemset mining with techniques for reducing overestimated utilities and pruning candidates", Expert Systems with Applications, Vol. 41, No. 8, pp. 3861-3878, 2014. http://dx.doi.org/10.1016/j.eswa.2013.11.038

[7] Q. Huynh-Thi-Le, T. Le, B. Vo, and H.B. Le, "An efficient and effective algorithm for mining top-rank-k frequent patterns", Expert Systems with Applications, Vol. 42, No. 1, pp. 156-164, 2015. http://dx.doi.org/10.1016/j.eswa.2014.07.045

[8] G. Pyun and U. Yun, "Mining top-k frequent patterns with combination reducing techniques", Applied Intelligence, Vol. 41, No. 1, pp. 76-98, 2014. http://dx.doi.org/10.1007/s10489-013-0506-9

[9] H. Ryang and U. Yun, "Performance Analysis of Frequent Pattern Mining with Multiple Minimum Supports", Journal of Internet Computing and Services, Vol. 14, No. 6, pp. 1-8, 2013. http://dx.doi.org/10.7472/jksii.2013.14.6.01

[10] R. Agrawal, T. Imilienski, and A, Swami, "Mining association rules between set of items in large databases", ACM SIGMOD, Vol.40, No.2, pp.207-216, 1993. http://dx.doi.org/10.1145/170036.170072

[11] C.-W. Wu, B.-E. Shie, V.S. Tseng, and P.S. Yu, "Mining top-K high utility itemsets", in Proc. of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2012, pp. 78-86. http://dx.doi.org/10.1145/2339530.2339546

[12] J. Pisharath, Y. Liu, B. Ozisikyilmaz, R. Narayanan, W.K. Liao, A. Choudhary, and Memik G, NU-MineBench version 2.0 dataset and technical report, http://cucis.ece.northwestern.edu/projects/DMS/Mineyunei@sejong.ac.kr

◐ 저 자 소 개 ◑

양 흥 모 (Heungmo Ryang)
2011년 충북대학교 컴퓨터공학전공 학사. (공학사)
2013년 충북대학교 대학원 컴퓨터공학 석사. (공학석사)
2013년~현재 세종대학교 대학원 컴퓨터공학 박사과정. (공학박사)
관심분야 : 데이터마이닝, 정보검색, 데이터베이스.
E-mail : ryang@sju.ac.kr

윤 은 일 (Unil Yun)
1997년 고려대학교 이학석사. (이학석사)
1997년~2006년 한국통신 멀티미디어연구소 전임/선임연구원.
2005년 Texas A&M Univ. 공학박사. (공학박사)
2006년~2007년 한국전자통신연구원, 선임연구원.
2007년~2012년 충북대학교 전자정보대학 컴퓨터공학부 조교수.
2012년~2013년 충북대학교 전자정보대학 소프트웨어학과 부교수.
2013년~현재 세종대학교 컴퓨터공학과 부교수.
관심분야 : 데이터마이닝, 정보검색, 데이터베이스.
E-mail : yunei@sejong.ac.kr

김 철 홍 (Chulhong Kim)
1993년 성균관대학교 대학원 정보공학 석사. (공학석사)
1983년~현재 한국전자통신연구원, 책임연구원.
관심분야 : 데이터 마이닝, 데이터분석, 데이터베이스.
E-mail : kch@etri.re.kr