시퀀스 데이터베이스에서 유연 규칙의 탐사 및 매칭

정회원 박 상 현*, Wesley Chu**, 김 상 욱***

Discovering and Matching Elastic Rules in Sequence Databases

Sang-Hyun Park*, Wesley Chu**, Sang-Wook Kim*** *Regular Members*

요 약

유연 패턴은 시간 축으로 확장 및 수축할 수 있는 요소들의 순서화 된 리스트이다. 유연 패턴은 서로 다른 샘플링 비율을 갖는 데이터 시퀀스들로부터 규칙들을 찾아내는데 유용하게 사용된다. 본 연구에서는 헤드(head: 규칙의 왼쪽 부분)와 바디(body: 규칙의 오른쪽 부분)가 모두 유연 패턴으로 구성된 규칙들을 신속하게 찾도록 하기 위하여 데이터 시퀀스로부터 서픽스 트리(suffix tree)를 구성한다. 이 서픽스 트리는 유연 규칙들의 압축된 표현이며, 타깃 헤드 시퀀스와 매치되는 규칙을 찾기 위한 인덱스 구조로서 사용된다. 만일, 매치되는 규칙을 찾을 수 없는 경우에는 규칙 완화(rule relaxation)의 개념을 이용한다. 클러스터 계층(cluster hierarchy)과 완화 오차(relaxation error)를 사용하여 타깃 헤드 시퀀스의 고유한 정보를 대부분 포함하고 있는 최소한으로 완화된 규칙을 찾는다. 다양한 실험을 통한 성능 평가를 통하여 제안한 기법의 우수성을 검증한다.

ABSTRACT

This paper presents techniques for discovering and matching rules with elastic patterns. Elastic patterns are the ordered lists of elements that can be stretched or shrunk along the time axis. Elastic patterns are useful for discovering rules from data sequences with different sampling rates. For fast discovery of rules whose heads (left-hand sides) and bodies (right-hands sides) are elastic patterns, we construct a suffix tree from succinct forms of data sequences. The suffix tree is a compact representation of rules, and is also used as an index structure for finding rules matched to a target head sequence. When matched rules cannot be found, the concept of rule relaxation is introduced. Using a cluster hierarchy and a relaxation error, we find the least relaxed rules that provide the most specific information on a target head sequence. Performance evaluation through extensive experiments reveals the effectiveness of the proposed approach.

I. 서 론

순차적인 데이터로부터 규칙을 탐사하는 것은 경향 예측을 위한 데이터 마이닝의 중요한 응용 분야의 하나이다[3][7]. 데이터 시퀀스에서 빈번히 발생하는 패턴으로부터 유용한 규칙을 탐사하기 위하여 다양한 연구들이 수행되어 왔다[1][6][9][14]. 패턴이란 부분적으로 순서화 된 요소들의 집합으로 정의된다. 패턴은 요소들이 배열 된 상태에 대한 제한에 따라 직렬 패턴(serial pattern)과 병렬 패턴(parallel pattern)으로 분류된다[9].

직렬 패턴의 하나로서 패턴 내의 요소들을 반복시킴으로써 패턴을 시간 축으로 확장시킬 수 있는 유연 패턴(elastic pattern)을 고려할 수 있다[9]. 유연 패턴 AB와 ABC는 정규 식(regular expression)을 사용하여 각각 A^+B^+와 $A^+B^+C^+$로 표현된다. <A, B>와 <A, A, B, B, B>는 유연 패턴 AB에 속하는 반면 <A, B, C>는 그렇지 못하다. 유연 패턴은 샘

* IBM T. J. Watson Research Center, ** UCLA 대학 전산학과, *** 강원대학교 컴퓨터정보통신공학부
논문번호 : 010016-0228, 접수일자 : 2001년 2월 28일
※ 본 연구는 2000년도 한국학술진흥재단 선도연구자 지원사업(KRF-2000-041-E00258)의 연구비 지원에 의하여 연구되었습니다.

플링 비율이 다른 데이터 시퀀스들로부터 규칙을 탐사하는데 매우 유용하게 사용된다. 예를 들어, 환자의 체온이 기록된 의료 데이터 시퀀스들이 있다고 가정하자. 그 중에는 환자의 체온을 매일 기록한 데이터 시퀀스도 있을 것이며, 체온을 매주 기록한 것도 있을 것이다. 뿐만 아니라, 하나의 데이터 시퀀스에서조차 서로 이웃하는 값의 샘플링 간격이 비 선형적으로 서로 다를 수 있다. 이러한 데이터 시퀀스들은 요소들을 시간 축으로 요소들을 늘이거나 줄이지 않고 서로 직접적으로 비교할 수 없다.

유연 규칙(elastic rules)이란 헤드(head: 규칙의 왼쪽 부분)와 바디(body: 규칙의 오른쪽 부분)가 모두 유연 패턴인 규칙으로 정의된다. 주어진 유연 패턴 α 와 β 에 대하여 유연 규칙 '$\alpha \rightarrow \beta$' 는 "만약 α 에 속하는 하나의 시퀀스가 발생하면, 이것은 β 에 속하는 시퀀스가 뒤를 따를 것"임을 의미한다. 여기서 시간 간격은 유연 패턴에 영향을 주지 않는다. 왜냐하면, 이 패턴들은 시간 축에서 유연성 있게 움직일 수 있기 때문이다.

직렬 패턴들을 갖는 규칙을 탐사하기 위하여 다양한 기법들이 제안되어 왔다[1][6][9][14]. 이들 중 대부분은 패턴과 그 서브 패턴 사이의 연관성을 이용하고 있다. 직렬 패턴 AB가 200번 발생하고 ABAC가 150번 발생하는 경우, 기존 기법들은 신뢰도 150/200(=0.75)를 가지고 AB → AC을 추출한다. 발생되는 경우의 수가 일정 수준 이하인 패턴은 중요하게 간주되지 않기 때문에 무시한다. 빈번히 발생되는 패턴을 찾기 위해 먼저 길이가 짧고 빈번히 발생되는 패턴을 찾은 후, 그 짧은 패턴들을 조합함으로써 길이가 더 긴 후보 패턴을 생성한다. 이렇게 생성된 후보 패턴은 그것이 빈번히 발생하는 것인지 아닌지 검사되고, 이런 조합과 검사의 단계가 전체적으로 빈번히 발생되는 패턴을 찾을 때까지 계속된다. 이 결과, 데이터 시퀀스들에 반복적인 액세스가 불가피하다.

데이터 시퀀스로부터 유용한 규칙들이 발견되면, 그 규칙들은 규칙 매칭 처리에 의해 타깃 헤드 시퀀스 q^*에 대한 미래의 경향을 예측하는데 사용된다. 하나의 규칙이 q^*와 매치된다는 것은 q^*의 각 요소와 규칙 헤드의 각 요소가 일치됨을 의미한다. 그러나 많은 규칙들이 존재하는 경우, q^*와 매치되는 규칙을 효율적으로 찾는 작업은 쉬운 일이 아니다.

또한, 타깃 헤드 시퀀스와 매치되는 규칙을 찾는데 실패하는 경우가 발생한다. 이것은 q^*가 빈번한 시퀀스가 아닌 경우에 종종 일어난다. 이와 같이, 빈번히 발생되지 않는 타깃 헤드 시퀀스의 경우를 해결하기 위한 한가지 방법은 규칙 완화(rule relaxation)에 대한 개념을 도입하는 것이다. 클러스터 계층(cluster hierarchy)을 기반으로 하나의 규칙 R은 그 요소 값들을 좀 더 일반적인 개념 혹은 넓은 범위를 나타내는 다른 요소 값으로 대치함으로써 새로운 규칙 R'로 완화된다. 타깃 헤드 시퀀스 q^*와 이것과 매치되지 않는 규칙 R이 있는 경우, 규칙 R은 q^*를 커버(cover)할 수 있는 규칙 R'로 완화될 수 있다. 여기서, 하나의 규칙이 q^*를 커버한다는 것은 이 규칙 헤드의 각 요소가 대응되는 q^*의 요소와 동일한 구간 범위 혹은 더 넓은 범위를 표현함을 나타낸다. 물론, q^*를 커버하는 다수의 완화된 규칙 R'을 만들어 낼 수 있으나, 본 연구에서는 최소한으로 완화된 규칙을 찾아내는 것을 목표로 한다. 이 최소한으로 완화된 규칙이 다른 완화된 규칙들과 비교하여 q^*를 더욱 정확하게 예측하는데 도움을 주기 때문이다.

본 논문에서는 다른 샘플링 비율을 갖는 데이터 시퀀스들로부터 유연 규칙을 탐사하고 매치하는 이슈에 대하여 논의한다. 본 논문에서는 유연 규칙을 탐사하는 알고리즘, 완전 일치 규칙 매칭 알고리즘, 완화 규칙 매칭 알고리즘을 제안한다. 또한, 다양한 실험을 통한 성능 평가를 수행함으로써 제안된 알고리즘들의 우수성을 검증한다.

Ⅱ. 연구 배경

2.1. 서픽스 트리

인서픽스 트리(suffix tree)[13]는 주어진 타깃 시퀀스와 정확하게 매치되는 서브시퀀스의 위치를 빠르게 찾도록 하는데 유용한 인덱스 구조이다. 서픽스 트리는 트라이(trie)와 서픽스 트라이(suffix trie)를 기반으로 하는 트리 구조이다. 트라이는 크기가 다양한 키워드들의 집합을 인덱싱하는데 사용되는 인덱스 구조이며, 서픽스 트라이는 키워드들의 집합이 시퀀스들의 서픽스들로 이루어진 트라이이다. 서픽스 트리는 서픽스 트라이에서 차수(degree)가 1인 모든 노드들을 결합한 형태이다. 여기서, 노드 N의 부모 노드를 PN 표기하고, N_i과 N_j를 연결하는 경로의 라벨을 $label(N_i, N_j)$로 표기한다.

2.2. 타입 요약 계층

타입 요약 계층(type abstraction hierarchy: TAH)[5]

은 다단계 클러스터 계층 구조로서 클러스터를 생성하기 위한 기준으로서 완화 오차(relaxation error)를 사용한다. n개의 요소를 갖는 클러스터 C = { x_1, x_2,....., x_n }에 대하여, C의 완화 오차

$RE(C) = \sum_{i=1}^{n} \sum_{j=1}^{n} P(x_i) P(x_j) | x_i - x_j |$ 로 정의된다. 여기서, $P(x_i)$, $P(x_j)$는 각각 x_i , x_j 가 발생할 확률을 의미한다.

참고문헌 [5]에서는 2차 TAH와 n차 TAH를 생성하기 위한 알고리즘을 제시하고 있다. TAH는 최대 엔트로피 클러스터링(maximum entropy clustering)에 비하여 구현이 쉽고, 등간격 클러스터링(equal-length interval clustering)에 비하여 더욱 정확한 클러스터를 생성한다. 그림 1은 [0, 7.0)의 범위 안의 값들을 요소들로 갖는 데이터 시퀀스들을 위한 TAH의 예를 보여주고 있다. 이 그림에서 완화 오차와 값의 범위는 각 노드 내에 있고, 노드들은 고유한 심볼들로 표현되어 있다.

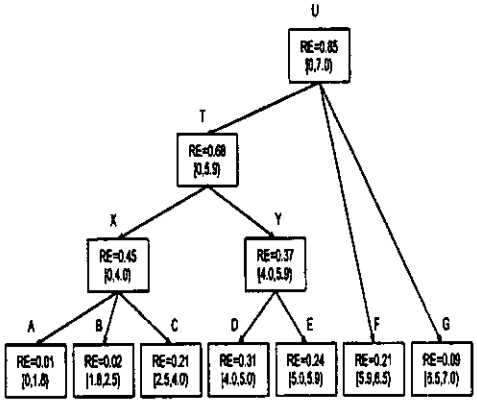


그림 1. TAH의 예.

Ⅲ. 규칙 탐사 방법

이 장에서는 서픽스 트리를 이용하여 데이터 시퀀스로부터 유연 규칙을 찾기 위한 효과적인 방법을 제안하고자 한다.

먼저, 데이터 시퀀스들로부터 TAH가 생성되어 있고, 각 TAH 노드에 대하여 고유의 심볼이 할당되어 있다고 가정하자. 패턴 α 의 지지값(support value)은 α 를 프리픽스(prefix)로 갖는 서픽스(suffix)들의 수로 정의된다. 최소 지지값(minimum support value) SUPmin는 빈번하게 발생되지 않는 패턴들을 필터 아웃하기 위해 사용된다. 또한, 상대 지지값(relative support value) RSUP(α)는 (패턴 α 를 프리픽스로 가지고 있는 서픽스의 수)/(전체 서픽스의 수)로 정의된다. 전체 데이터 시퀀스의 수와 길이가 가변적인 응용에서는 상대 지지도를 사용하는 것이 절대 지지도를 사용하는 것 보다 유용하다.

유연 규칙 탐사 문제는 다음과 같이 정의된다.

M개의 시퀀스 x_1^*, x_2^*, ... , x_M^*로 구성되는 데이터베이스와 최소 지지값 SUPmin가 주어질 때, 최소 지지값이 적어도 SUPmin 이상인 유연 패턴으로 구성되는 규칙을 탐사하라.

이러한 유연 규칙 탐사는 다음의 다섯 단계로 구성된다.

□ 제 1 단계: 심볼 변환 과정

데이터 시퀀스의 각 요소 값들을 대응되는 TAH 내의 리프 노드와 일치하는 심볼로 변환한 다. x^*를 심볼 형태로 변환 된 표현을 S(x^*)로 표기한다. 예를 들어, 데이터 시퀀스 x^* = <3.4, 3.0, 3.7, 2.3, 2.1, 4.3>는 그림 1에 나타난 TAH를 사용하여 심볼로 변환하면 S(x^*) = <C, C, C, B, B, D>가 된다.

□ 제 2 단계: 축약 과정

심볼로 변환 된 데이터 시퀀스 S(x^*)를 간결한 표현 C(S(x^*))로 축약된다. 여기서, C(S x^*)) 는 연속적으로 같은 값을 갖는 요소들을 하나의 값을 갖는 요소로 치환함으로써 구할 수 있다. 이 단계는 유연 패턴의 특성을 고려하기 위한 것이다. 예를 들어, 위의 단계 1에서 S(x^*) = <C, C, C, B, B, D>는 C(S(x^*)) = <C, B, D>로 축약된다. 본 논문에서는 편의상 C(S(x^*))를 X^*로 표기한다.

□ 제 3 단계: 서픽스 트리 구성 과정

M개의 축약된 심볼 데이터 시퀀스 X^*, ... , X_M^*들을 대상으로 서픽스 트리를 구성한다. 서픽스 트리의 구성을 위하여 McCreight의 알고리즘[8] 혹은 점진적 디스크 기반 알고리즘(incremental disk-based algorithm)[4]을 이용할 수 있다.

□ 제 4 단계: 서픽스 트리 정제 과정

서픽스 트리 내 각 노드의 지지값을 계산하고, 지지값이 SUPmin이하인 노드들은 서픽스 트리로부터 제거한다. 내부 노드의 지지값은 그 자식 노드들의 지지값들의 합을 통하여 구할 수 있다. 리프 노드들의 지지값은 리프 노드들에 의하여 표현되는 서픽스들의 수를 의미한다. 본 논문에서는 이렇게 만들어진 정제된 서픽스 트리를 규칙 트리(rule tree)라

정의한다.

□ 제 5 단계: 규칙 추출 과정

각 노드의 신뢰도 값을 계산한 후, 규칙들을 추출한다. 노드 N의 신뢰도 값을 계산하기 위한 식은, confidence(N) = support(N) / support(PN)이며, 여기서 PN은 N의 부모 노드이다. PN으로부터 N까지의 경로 상에 L개의 라벨이 있는 경우, 아래와 같이 L개의 규칙들을 추출 할 수 있다.

R_1 : label(rootNode, PN) → label(PN, N)

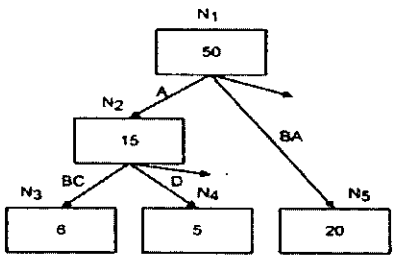
R_2 : label(rootNode, PN) •
(label(PN, N)[1:1]) → label(PN, N)[2:L]

R_3 : label(rootNode, PN) •
(label(PN, N)[1:2]) → label(PN, N)[3:L]

...

R_L : label(rootNode, PN) •
(label(PN, N)[1:L-1])→label(PN, N)[L:L]

여기서, label(PN, N)[i : j]는 i에서 j까지 ($i \le j \le L$) 위치에 있는 요소들을 포함하는 label(PN, N)의 서브시퀀스이다. 또한, '•'는 두 시퀀스들을 접합하기 위한 이진 연산자이다. 만약 N이 루트 노드이면 label(rootNode, PN)는 널 시퀀스<>가 된다. 여기서, R_1의 신뢰값은 confidence(N)이며, R_2 , R_3 , ... , R_L의 신뢰값은 1이다.



(a) 규칙 트리.

```
<> -> A  with confidence 15/50
<> -> BA with confidence 20/50
A  -> BC with confidence 6/15
A  -> D  with confidence 5/15
AB -> C  with confidence 1
B  -> A  with confidence 1
```

(b) 트리에서 규칙 추출.

그림 2. 규칙 트리와 대응되는 규칙들.

그림 2는 규칙 트리의 일부와 트리로부터 추출된 규칙들을 나타낸 것이다. 각 노드 내의 값은 그 노드의 지지값을 나타낸다.

Ⅳ. 완전 일치 규칙 매칭 방법

완전 일치 규칙 매칭 문제는 다음과 같이 정의된다.

규칙 트리, TAH, 그리고 타깃 헤드 시퀀스 q^*가 주어지는 경우, q^*와 헤드가 완전히 일치되는 규칙을 검색하라.

이러한 완전 규칙 매칭은 다음과 같은 두 단계로 구성된다.

□ 제 1 단계: 완전 일치 헤드의 검색

규칙 트리를 인덱스 구조로 사용하여 타깃 헤드 시퀀스와 완전히 일치되는 규칙의 헤드 h^*를 찾는다. 알고리즘 1은 완전 일치 규칙 매칭 알고리즘 RTI-E(Rule Tree Index for Exact matching)를 나타낸다. 여기서 C(S(q^*))의 표현을 위해 Q^*로 표기한다. 알고리즘 1은 규칙 트리를 탐색하여 (N, p)의 쌍을 리턴한다. (N, p)는 매치된 규칙 헤드 h^* = label(rootNode, PN) • (label(PN, N)[1 : p])를 나타낸다. 이 알고리즘에서 첫 번째 호출에서 사용되는 두 인수는 rootNode와 Q^*이다.

□ 제 2 단계: 완전 일치 헤드로부터 규칙의 추출

완전 일치 헤드와 그 이후에 나타나는 서브시퀀스들 사이의 연관성을 사용하여 규칙들을 추출한다. 여기서, RIT-E는 한 쌍의 (N, P)를 리턴하고, label(PN, N)의 길이가 L이라고 가정하자. p < L인 경우, 추출되는 규칙은 'label(rootNode, PN) • (label(PN, N)[1 : p]) → label(PN, N)[p+1 : L]이며, 그 신뢰도는 1이 된다. $p \ge L$인 경우에는 추출되는 규칙들의 수는 N의 자식 노드 수와 같다. N의 각각의 자식 노드 CN에 대하여, 추출되는 규칙은 label(rootNode, N) → label(N, CN)이며, 그 신뢰도는 CN이 갖는 신뢰도와 같다.

Ⅴ. 완화 규칙 매칭 방법

완화 규칙 매칭 문제는 다음과 같이 정의된다.

규칙 트리, TAH, 그리고 타깃 헤드 시퀀스 q^*가 주어지는 경우, q^*를 커버하는 최소한의 완화규칙들을 검색하라.

길이가 $|q^*|$와 같지 않는 규칙 헤드 h^*는 q^*를 커버하기 위하여 완화될 수 있다. 본 논문에서는 h^*와 의 유사성을 측정하기 위한 척도로서 길이가 $|q^*|$와 같지 않는 규칙헤드 h^*는 완화 오차 기반 타임 와핑 거리 함수 DRE(h^*, q^*)를 제안한다.

Algorithm RTI-E:

Input : a node N, a target head sequence Q^*

Output : the child node CN, the length of matched prefix

Visit the node N;

Select the child node CN, where label(N, CN) is matched to the prefix of Q^*;

Remove the matched prefix from Q^*;

If Q^* becomes empty **Then** return a pair (CN, the length of a matched prefix);

Else Call RIT-E(CN, Q^*) recursively;

알고리즘 1. RTI-E: 완전 일치 규칙 매칭 알고리즘

Algorithm RTI-S

Input : node N, cumulative distance table T

Visit the node N;

For each child node CN do

Build a new cumulative distance table *new*T, by adding rows corresponding to label(N, CN) on T;

Find a nearer rule head h^* from *new*T and update *MinDist*;

If futher traverse-down the tree is necessary, **Then** call RTI-S(CN, *new*T) recursively;

알고리즘 2. RTI-S: 완화 규칙 매칭 알고리즘

DRE(h^*, q^*)는 h^*와 q^*의 차이를 최소화하는 최적 매핑을 찾기 위해 h^*와 q^* 각각의 요소들을 비선형적으로 반복시킬 수 있다. h^*[i]와 q^*[j]가 [j]) − RE(h^*[i])로 정의된다. 여기서, RE(h^*[i], q^*[j])는 h^*[i]와 q^*[j]를 모두 포함하는 최하위 노드의 완화 오차이고, RE(h^*[i])는 h^*[i]를 포함하는 최하위 노드의 완화 오차이다. DRE(h^*, q^*)의 설명은 참고 문헌 [10]에 보다 상세히 기술되어 있다.

DRE(h^*, q^*)는 순환 관계(recurrence relation) r(x, y)를 기반으로 하는 동적 프로그래밍 기법에 의해 효과적으로 계산할 수 있다. 여기서, x = 1, ... ,|h^*|이며, y = 1, ... ,|q^*|이다. 최종 누적 거리 r(|h^*|, |q^*|)은 h^*가 q^*를 커버하도록 만들기 위해 요구되는 완화의 정도이다. 그림 3은 그림 1에서 나타난 클러스터 계층을 사용하여 구한 DRE(h^*=<C, A, E, D, A>, q^*=<C, E, A>)의 누적 거리 함수 테이블 T, 타임 와핑과 완화 후의 요소 매핑을 각각 나타낸 것이다.

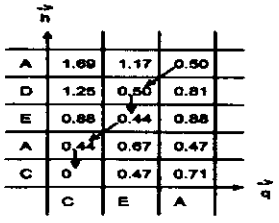
다음은 완화 규칙 매핑의 처리를 위한 두 가지 단계를 구체적으로 설명한다.

□ 제 1단계: 최근접 규칙 헤드의 검색

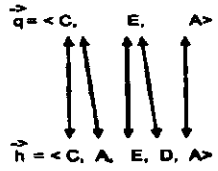
최소한으로 완화된 규칙을 생성하기 위하여 먼저 규칙 트리를 탐색함으로써 최소한의 완화를 통하여 타깃 헤드 시퀀스 q^*를 커버하는 규칙 헤드 h^*를 찾는다. 이 완화 규칙 매칭 알고리즘 RTI-S (Rule Tree Index for Similarity matching)는 알고리즘 2에 나타나 있다. 여기서, 검색 수행을 시작하기 전에 타깃 헤드 시퀀스 q^*는 축약된 표현 Q^*로 변환되어야 한다.

알고리즘은 검색이 수행되는 동안 Q^*, 현재까지 찾은 최근접 규칙 헤드 h^*, Q^*로부터 h^*까지의 거리 MinDist 등 세 가지 변수를 유지한다. 알고리즘의 첫 번째 호출에서 사용하는 두 인수는 rootNode와 emptyTable이다. RIT-S는 가지 치기 방식(branch pruning approach)[11]을 하고, 같은 프

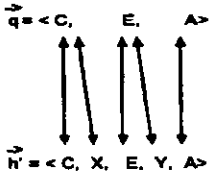
리픽스를 갖는 규칙 헤드들이 누적 거리 테이블을 공유하는 것을 허용함으로써 검색 시간을 크게 줄일 수 있다.



(a) 누적 거리 테이블(T)



(b) 타임 와핑 후의 요소 매핑



(c) 완화 후의 요소 매핑

그림 3. DRE(h^*=<C,A,E,D,A>, q^*=<C,E,A>)의 누적 거리 테이블 및 요소 매핑.

□ 제 2단계: 최근접 규칙 헤드로부터의 규칙 추출 Q^*와 가장 가까운 규칙 헤드 h^*를 찾은 후 h^*와 h^* 직후에 나타나는 서브시퀀스로부터 최소한으로 완화된 규칙들을 추출한다. 이 단계에서는 제4장에서 설명한 것과 동일한 방법을 사용하여 h^*로부터 규칙들을 추출한다. 그런 후, h^*와 Q^*를 매핑한 방식에 따라 규칙의 헤드와 바디의 심볼들을 그들의 완화된 심볼들로 변환한다. 이 결과, 축약된 형태의 규칙들을 얻을 수 있다. 마지막으로 같은 헤드와 바디를 갖는 서로 다른 규칙들은 병합되고, 그 신뢰도는 다시 계산된다.

VI. 성능 평가

본 장에서는 제안하는 기법의 우수성을 평가하기 위하여 랜덤 워크(random walk) 합성 데이터[10]를 사용하여 실험을 수행한다. 찾을 수 있는 규칙들의 수를 조절하기 위해 상대 지지값$RSUP_{min}$를 이용한다.

6.1. 규칙 탐사

규칙 탐사 알고리즘의 성능 평가를 위한 척도로서 전체 수행 시간을 사용한다. 첫 번째 실험에서는 시퀀스의 평균 길이를 200으로 고정시키고, 시퀀스의 수를 100에서 10,000까지 증가시키면서 전체 수행 시간의 변화를 조사한다. 두 번째 실험에서는 시퀀스의 수를 500으로 고정시키고, 시퀀스의 평균 길이를 100에서 10,000까지 증가시키면서 전체 수행 시간의 추이를 살펴본다. 그림 4와 그림 5에서 나타난 바와 같이, 전체 수행 시간은 데이터 시퀀스의 수와 평균 길이 각각에 대하여 선형적으로 비례하는 것으로 나타났다.

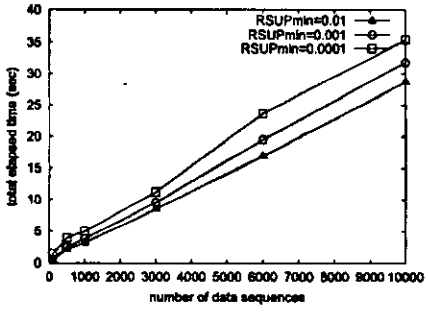


그림 4. $RUSP_{min}$를 이용한 시퀀스 수에 따른 실행 시간의 비교.

또한, 이러한 선형적인 증가 형태는 다양한 RSUPmin 값에 대해서도 그대로를 유지되는 것으로 나타났다. 이 결과는 제안된 기법이 효과적으로 규칙 매칭을 수행하고 있음을 보이는 것이다.

6.2. 규칙 매칭

평균 길이가 400인, 500개의 데이터 시퀀스에 대하여 그림 6은 규칙의 수가 증가함에 따라 RTI-E 알고리즘과 순차 스캔 기반 알고리즘(SS-based algorithm)을 이용하는 각각의 경우 완전 일치 규칙 매칭의 평균 탐색 시간을 비교하고 있다. 또한, 그림 7은 RTI-S 알고리즘과 순차 스캔 기반 알고리즘(SS-based algorithm)을 이용하는 각각의 경우의 완화 규칙 매칭의 평균 탐색 시간을 비교하고 있다. 순차 스캔 기반 알고리즘은 규칙의 수가 증가함에 따라 선형적으로 비례하여 크게 증가하는 반면,

RTI-E와 RTI-S 알고리즘의 탐색 시간은 상대적으로 낮은 일정한 수준을 유지하는 것으로 나타났다. 이 결과는 제안된 기법이 효과적으로 규칙 매칭을 수행하고 있음을 보이는 것이다.

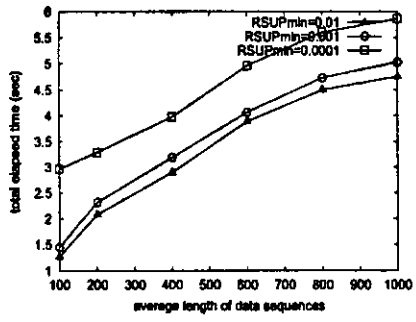


그림 5. $RUSP_{min}$를 이용한 시퀀스 길이에 따른 실행 시간의 비교

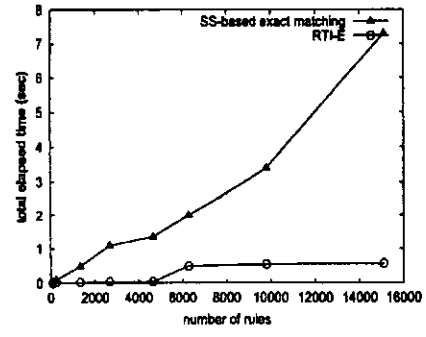


그림 6. RTI-E와 SS-기반 알고리즘을 이용한 규칙 수에 따른 완전 일치 규칙 매칭 실행 시간의 비교

Ⅶ. 결론

본 논문에서는 시퀀스 데이터베이스로부터 유연 규칙들을 탐사하기 위한 기법을 제안하였다. 또한, 최소한으로 완화된 규칙들을 검색하기 위한 효율적인 기법들을 제시하였다. 현 논문에서는 주로 수치적 요소를 갖는 데이터 시퀀스에 초점을 맞추었다. 만약, 비수치적 형태인 경우에는 인코딩 기법을 사용하여 비수치적인 형태의 요소들을 수치적인 형태의 요소로 변환하여 쉽게 적용할 수 있을 것이다.

합성 데이터 시퀀스들을 이용한 실험 결과에 의하면, 1) 제안된 기법의 규칙 탐사 알고리즘은 데이터 시퀀스의 수가 증가하거나 시퀀스의 평균 길이가 증가함에 따라 전체 수행 시간이 이와 선형적으로 비례하는 것으로 나타났으며, 2) 본 논문에서 제시된 완전 일치 및 완화 규칙 매칭 알고리즘은 SS-기반 알고리즘과 비교하여 월등한 성능을 가지는 것으로 나타났다.

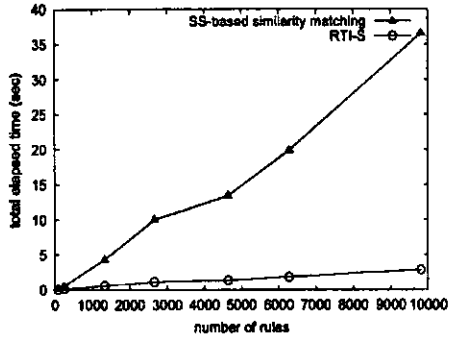


그림 7. RTI-S와 SS-기반 알고리즘을 이용한 규칙 수에 따른 완화 규칙 매칭 실행 시간의 비교

참 고 문 헌

[1] R. Agrawal, and R. Srikant, "Mining Sequential Patterns," *Proc. IEEE ICDE*, 1995.

[2] D. J. Berndt and J. Clifford, "Finding Patterns in Time Series: A Dynamic Programming Approach," *Advances in Knowledge Discovery and Data Mining*, 1996.

[3] P. S. Bradley, U. M. Fayyad, and O. L. Mangasarian, "Data Mining: Overview Optimization Opportunities," *Micorsoft Research Report* MSR-TR-98-04, 1998.

[4] Bartolicci, P. Bieganski, J. Riedl, and J. V. Carlis, "Generalized Suffix Trees for Biological Sequence Data: Applications and Implementation," *Proc. Hawaii Int'l Conf. on System Sciences*, 1994.

[5] W. W. Chu, and K. Chiang, "Abstraction of High Level Concepts from Numerical Values in Databases," *Proc. of AAAI Workshop on Knowledge Discovery in Databases*, 1994.

[6] G. Das, K. Lin, H. Mannila, G. Rengnathan, and P. Smyth, "Rule Discovery from Time Series," *Proc. Int'l. Conf. on Knowledge Discovery and Data Mining*, 1998.

[7] U. M. Fayyad, "Mining Databases: Toward Algorithms for Knowledge Discovery," *IEEE Data Engineering Bulletin*, 21(1), 1998.

[8] E. M. McCreight, "A Space-Economical Suffix

Tree Construction Algorithm," *Journal of the ACM*, Vol. 23, No. 2, 1976.

[9] H. Mannila, H. Toivonen, and A. I. Verkamo "Discovery Frequent Episodes in Sequences," *Proc. Int'l. Conf. on Knowledge Discovery and Data Mining*, 1995.

[10] S. Park and W. W. Chu, Discovering and Matching Elastic Rules from Sequence Databases, *UCLA Technical Report* UCLA-CS-TR-200012, 2000.

[11] S. Park and W. W. Chu, J. Yoon, and C. Hus, "Efficient Searches for Similar Subsequences of Different Lengths in Sequence Databases," *Proc. IEEE ICDE*, 2000.

[12] L. Rabinar, and B. Juang, Fundamentals of Speech Recognition, *Prentice Hall*, 1993.

[13] G. A. Stephen, String Searching Algorithms, *World Scientific Publishing*, 1994.

[14] J. T.-L. Wang, G.-W. Chirn, T. G. Marr, B. Shapiro, D. Shasha, and K. Zhang, "Combinatorial Pattern Discovery for Scientific Data: Some Preliminary Results," *Proc. ACM SIGMOD*, 1994.

박 상 현(Sanghyun Park)

1989년 2월 : 서울대학교 컴퓨터공학과 졸업(학사)
1991년 2월 : 서울대학교 컴퓨터공학과 졸업(석사)
2001년 1월 : UCLA대학교 전산학과 졸업(박사)
1991년 3월~1996년 7월 : 대우통신 연구원
2001년 2월~현재 : IBM T.J. Watson Research Center Post-Doctorial Fellow
<주관심 분야> 시공간 데이터베이스, 멀티미디어 데이터베이스, 데이터 마이닝, 데이터 웨어하우징, XML, 분산 데이터베이스

Wesley Chu

Professor of Computer Science Department at University of California, Los Angeles. He received the B.S. and M.S. degrees from the University of Michigan in 1960 and 1961, and earned the Ph.D. in Electrical Engineering from Stanford University in 1966. He has worked on switching circuits and computer systems at General Electric and at IBM, and did his research in computer communications at Bell Labs, 1966-69. He joined the UCLA Faculty in 1969, and was the Chairman of the Department from 1988-91. He is an IEEE fellow. He was the program co-chairman for the First International Conference on Data Engineering, and the program co-chairman for the 12th International Conference on VLDB in 1986. His current research interests include distributed processing and distributed database systems, knowledge base and database systems, and intelligent information systems.

김 상 욱(Sang-Wook Kim)

1989년 2월 : 서울대학교 컴퓨터공학과 졸업(학사)
1991년 2월 : 한국과학기술원 전산학과 졸업(석사)
1994년 2월 : 한국과학기술원 전산학과 졸업(박사)
1991년 7월~8월 : 미국 Stanford University, Computer Science Department 방문 연구원
1994년 2월~1995년 2월 : 정보전자연구소 Post-Doc.
1999년 8월~2000년 8월 : 미국 IBM T.J. Watson Research Center 방문 교수
1995년 3월~현재 : 강원대학교 컴퓨터정보통신공학부 부교수
<주관심 분야> DBMS, 데이타 마이닝, 멀티미디어 정보 검색, 공간 데이터베이스/GIS, 실시간 주기억장치 데이타베이스, 트랜잭션 관리