J Korea Industr Inf Syst Res Volume 17 Number 6
http://dx.doi.org/10.9723/jksiis.2012.17.6.059

EPs-TFP 마이닝 기법을 이용한 단백질 Disorder/Order 지역 분류†

(Protein Disorder/Order Region Classification Using EPs-TFP Mining Method)

이 헌 규*, 신 용 호**
(Heon Gyu Lee and Yong Ho Shin)

요 약 단백질은 서열의 disorder 구역이 생물학적 반응을 일으켜 order로 변하는 과정에서 그 기능을 하게 되므로 서열 데이터에서 disorder 구역과 order 구역을 분리하는 것은 단백질의 3차 구조 및 특성을 예측하는데 반드시 필요하다. 따라서 이 논문에서는 효율적인 disorder와 order 구역 분류를 위해서 단백질의 특정 특징에 치우치지 않는 분류 결과를 얻으면서, 분류 속도를 향상 시킬 수 있도록 서열 데이터를 이용한 분류/예측 기법을 제안한다. 출현패턴 기반의 EPs-TFP 기법은 중복 출현패턴이 제거된 필수 출현패턴만을 이용하는 분류/예측 기법이다. 이 분류 기법은 disorder 구역의 서열 출현패턴들을 발견하며, 이러한 서열 출현패턴은 disorder 구역에서는 빈발하지만 order 구역에서는 상대적으로 빈발하지 않는 패턴들이다. 또한 제안 알고리즘의 성능 향상을 위해서 기존의 P-tree, T-tree 개념의 TFP 기법을 확장하여 분류/예측 기법으로 적용하였다. EPs-TFP 기법의 성능평가를 위해서 Disprot 4.9와 CASP 7 데이터를 활용하였고, disorder/order 구역을 분류한 결과, 민감도 73.6, 특이도 69.5, 정확도 74.2를 보였다.

핵심주제어 : 출현패턴, TFP-트리, 단백질 Disorder/Order 구역, 데이터 마이닝

Abstract Since a protein displays its specific functions when disorder region of protein sequence transits to order region with provoking a biological reaction, the separation of disorder region and order region from the sequence data is urgently necessary for predicting three dimensional structure and characteristics of the protein. To classify the disorder and order region efficiently, this paper proposes a classification/prediction method using sequence data while acquiring a non-biased result on a specific characteristics of protein and improving the classification speed. The emerging patterns based EPs-TFP methods utilizes only the essential emerging pattern in which the redundant emerging patterns are removed. This classification method finds the sequence patterns of disorder region, such sequence patterns are frequently shown in disorder region but relatively not frequently in the order region. We expand P-tree and T-tree conceptualized TFP method into a classification/prediction method in order to improve the performance of the proposed algorithm. We used Disprot 4.9 and CASP 7 data to evaluate EPs-TFP technique, the results of order/disorder classification show sensitivity 73.6, specificity 69.51 and accuracy 74.2.

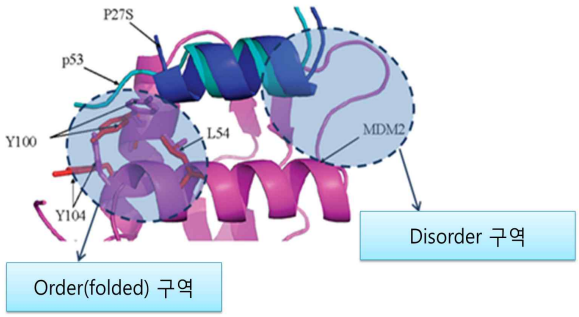
Key Words : Emerging Patterns, TFP-tree, Protein Disorder/Order region

† 이 논문은 지식경제부 우정사업본부의 우정기술연구개발사업 (2006-X-001-02, 실시간 우편물류 운영기술 개발) 지원으로 수행되었음
* 한국전자통신연구원 융합기술연구부문 선임연구원, 제1저자
** 영남대학교 경영학부, 교신저자(yhshin@ynu.ac.kr)

1. 서 론

생물정보학 분야에서 생명체의 기능과 진화 과정을 이해하는데 있어 단백질 기능 규명은 중요한 연구 분야이다. 아직 알려지지 않은 단백질의 기능을 알아내는 방법은 단백질 서열 분석법과 폴딩된 3차 구조 접근법으로 나뉜다. 서열 분석은 기존의 알려진 단백질 서열과 서로 비교함으로서 기능과 진화 정도를 알아내는 방법이나 서열의 상동성이 낮을 경우 유사성 비교가 불가능하다. 단백질의 3차 구조(폴드) 분석은 폴드의 유사성 검색을 통해 그 기능을 예측하나 전체적인 3차 구조가 유사하더라도 다른 기능을 가질 수 있고 3차 구조가 다르더라도 핵심 영역의 구조가 유사할 경우 유사 기능을 가질 수 있다 [1], [2].

최근 단백질의 기능 예측에 있어 서열 데이터의 disorder 구역과 order 구역을 분류하는 연구가 진행되고 있다. disorder 구역이란 단백질 서열이 풀려 있는 부분을 의미하고 order 구역은 생화학적 반응을 일으켜 접힌 부분을 말한다(그림 1).



<그림 1> 단백질 서열의 disorder/order 구역

그림 1의 단백질 disorder 구역을 분류/예측해야 하는 중요한 이유는 다음과 같다.

- Disorder 구역은 생화학적 반응을 일으켜 order 구조로 변한다. 이 과정에서 단백질은 특정 기능이 결정되기 때문에 disorder 구역을 알아내면 해당 단백질의 기능을 예측 할 수 있다. 또한 disorder 구역은 암 단백질을 제외한 대부분의 hub_protein(반응을 일으켜 결합된 단백질 파트너가 많은 단백질)보다도 결합 반응을 잘하므로 disorder 구역 예측이 단백질 기능 예측에서의 중요성을 준다 [3].
- 단백질 서열 분석 방법의 정렬 비교에 있어서 disorder와 order 구역이 잘 분리되어 있을 경우, 유사성 분석 단계에서 두 구역이 서로 정렬 비교되는 것을 막아 유사성 비교 결과가 더 좋게 된다.
- 모티프 단백질에서 짧은 선형의 펩티드 구역인 ELMs(Eukaryotic Linear Motifs)는 단백질의 서열이나 구조와 관계없이 독자적 기능을 가지는 구역이다. 이 ELMs의 70%가 disorder 구역에 위치하고 있으므로 ELMs을 발견하는 데에도 disorder 구역이 중요한 역할을 한다 [4].

이러한 disorder 구역 예측 필요성에 따라 현재까지의 연구는 단백질의 구조 데이터보다 이미 많이 알려진 서열 데이터를 이용하는 방법과 단백질의 특정 성질을 나타내는 특징(feature)을 추출하여 분류/예측 알고리즘을 통해 구역을 예측하는 방법들이 제안되었다. 단백질 서열 데이터로부터 disorder 구역 예측을 하는 기존의 분류/예측 기법은 주로 서열 데이터에 슬라이딩 윈도우(sliding window) 적용과 특징선택(feature selection) 방법을 사용하여 생성된 패턴들을 이용하는 것이다. 다음으로 이러한 패턴들은 이진 분류/예측 모델에 입력으로 하여 disorder 구역을 예측하며, 주로 SVM (Support Vector Machine), NNs (Neural Networks), Regression 등이 사용된다. DISOPRED2 [5], VSL2 [6], POODLE_L [7], PrDOS [8]에서는 분류 모델로서 SVM을 사용하였고, RONN [9], NORSp [10], VL3 [11], DisEMBL [12], PONDR [13], DISpro [14]은 NNs을 적용 하였다. 이외에 regression 기법을 사용한 분류/예측 모델은 VL2 [10]와 VSL1 [6], 슬라이딩 윈도우 방법 적용은 FoldIndex [15], Random Forest 모델은 DRaai-S [16]이 있다. 이러한 많은 예측 프로그램이 제안되어 사용되었지만 서열 데이터만을 사용한 구역 분류모델은 원 단백질 서열들을 입력으로 하여 예측하므로 모델 구성 및 예측 속도가 느린 단점을 가진다. 또한 단백질의 친수성, B-factor 포함량, position-specific score 프로파일 등 단백질의 특정 특징을 추출할 경우, 프로그램 마다 분류 모델을 구축하는데 쓰이는 데이터가 다르므로 예측되는 disorder 구역도 차이가 있다 [12].

이 논문에서는 단백질의 특정 성질을 나타내는 특징들을 사용하지 않고 단백질 서열 자체를 사용하며, 구역 분류 속도도 향상시킬 수 있는 출현패턴 기반의

EPs-TFP (Emerging Patterns - Total form Partial) 기법을 제안한다.

EPs-TFP 알고리즘은 1차 서열 데이터로부터 disorder 구역의 서열에 대한 출현패턴(*EPs*)을 발견하며, 이러한 패턴은 disorder 구역(Target class) 에서는 빈발하지만 order 구역(Background class)에서는 상대적으로 빈발하지 않는 서열 패턴들을 사용한다. 또한 기존의 패턴기반 방식의 단점인 분류모델 구축 속도를 향상시키기 위해서 대표적 빈발패턴 마이닝 기법인 *P*-tree, *T*-tree 개념의 *TFP-growth* [17] 알고리즘을 확장, 개선하여 출현패턴 기반의 분류/예측 기법으로 적용한다. 따라서 1차 서열에서의 disorder 구역을 빠르게 발견할 수 있으며, 트리 구조 알고리즘으로 메모리의 효율적 관리가 가능하다[18].

2. Disorder 구역 예측을 위한 EPs-TFP

2.1 출현패턴(EPs)

출현패턴이란 성장률(growth-rate)의 적절한 분류 기준을 적용하여 서로 다른 클래스에 해당되는 데이터 집합의 분명한 변화와 차이를 보이는 속성값들의 조합으로 지지도를 증가시키는 항목집합을 출현패턴이라고 한다 [19]. 즉, 출현패턴은 두 개의 분할된 데이터 집합을 명확하게 구분해 주는 패턴을 말하는 것으로, 이러한 패턴들은 하나의 데이터 집합에서 다른 클래스를 갖는 데이터 집합 사이에 명확한 차별 점을 가진다. 일반적으로 연관(association) 분석에서 자주 발생하는 패턴과는 달리 출현 패턴은 높은 구별력(discriminating power)으로 분류 문제에 적용되어 더욱 유용하다고 증명되어 있다. 출현패턴에 대한 문제 정의는 다음과 같다.

[정의 1] 성장률 (growth rate) : 두 개의 서로 다른 클래스에 해당되는 두 집합 D_1, D_2에 대해, 패턴 X의 D_1에 대한 D_2의 성장률은 다음과 같이 정의 된다 [18].

$$GrowthRate(X) = GR(X) = \begin{cases} 0 & If\ sup_1(X)=0\ and\ sup_2(X)=0 \\ \infty & If\ sup_1(X)=0\ and\ sup_2(X)>0 \\ \sup_2/\sup_1 & otherwise \end{cases} \quad (1)$$

여기서, D_1을 배경(Background) 데이터 집합, D_2를 목표(Target) 데이터 집합이라고 하며, 출현패턴은 배경 데이터로부터 목표 데이터 집합에 대해 높은 성장률을 가지는 패턴을 의미한다. 또한 성장률 임계값 $\rho > 1$에 대해서 패턴 X가 $GrowthRate(X) \gg \rho$의 성장률을 가질 때, 패턴 X를 ρ-Emerging Pattern ($\rho - EP$)라 한다.

예를 들어 임계값인 성장률(*minimum growth rate*) $\rho = 2$라 하고, disorder 및 order 두 클래스에 해당되는 출현패턴은 다음과 같다.

- EP_1: 부분 서열이 {K,L,C}인 값을 가진 패턴의 지지도가 disorder에 대해 2/9, order에 대해 3/5를 가질 경우, 성장률(*GR*)=2.7이므로 {K,L,C}는 order 구역에 대한 출현패턴이다.
- EP_2: 부분 서열 {C,S}의 지지도가 disorder에 대해 4/9, order에서 1/8일 경우, 성장률(*GR*)=3.5이므로 {C,S}는 disorder에 대한 출현패턴이다.

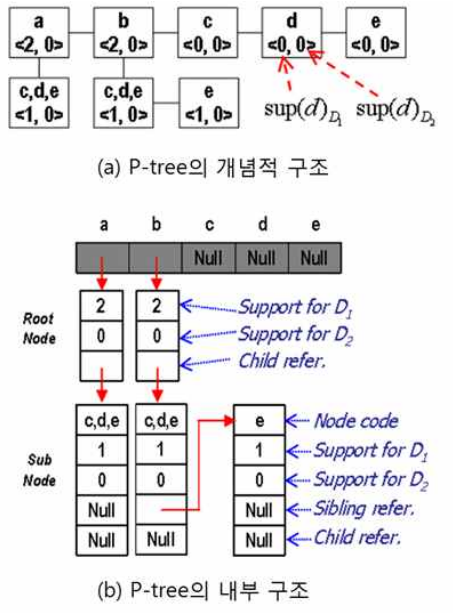
2.2 EPs-TFP를 이용한 서열 출현패턴 추출

단백질 서열 데이터로부터 효율적인 disorder 구역의 출현패턴 마이닝을 위해서 기존의 *Apriori-TFP* [17] 방법의 *P*-tree, *T*-tree 구조를 유지하는 기법을 기술한다. 패턴 발견 과정은 유사하나 순서화된 단백질 서열의 목표 클래스(disorder 구역)에 대한 출현패턴 발견을 위해 추가적인 파라미터 정의 및 알고리즘을 확장한다.

2.2.1 EPs-TFP의 P-tree, T-tree 구조

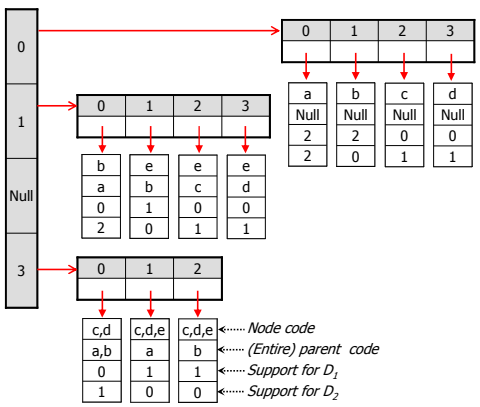
기존 *Apriori-TFP* [17] 알고리즘의 *P*-tree, *T*-tree 및 *P*-tree 테이블을 분류기법인 출현패턴 마이닝 적용을 위한 구조적 개선 사항은 다음과 같다.

1) *P*-tree (Partial support tree) : 데이터 항목의 압축된 열거트리로서 노드에는 분류 클래스 분포를 고려한 카운트 값(support value)이 저장되며, 그 구조는 그림 2와 같다.



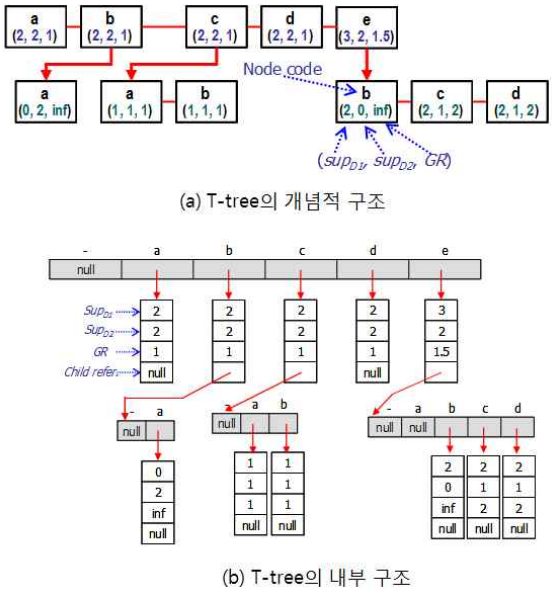
<그림 2> P-tree의 구조

- 상위(top) 레벨은 노드 배열로서 자식 노드의 참조하는 1-*itemset* 인덱스이다. 또한 상위 레벨은 각 클래스의 카운트 값을 저장하고 하위 노드의 링크로 구성된다.
- 그림 3의 *P*-tree로부터 구축된 *P*-tree 테이블이며, 이 테이블에는 *P*-tree의 모든 정보가 저장되어 있기 때문에 메모리에 저장하여 사용할 경우, 알고리즘의 계산 시간이 상당히 단축 될 수 있다. 그림 3에서 오른쪽 인덱스(0, 1, Null, 3)는 *P*-tree로부터 생성된 패턴에 포함된 항목의 수를 나타낸다.



<그림 3> P-tree 테이블의 구조

2) *T*-tree (Total support tree)는 P-tree 테이블을 스캔하여 구성되는 열거 트리 구조로서 각 노드는 클래스 분포를 가지며, 출현패턴 추출을 위한 식 1의 성장률, *GR*값 저장을 위한 필드가 추가된다.



<그림 4> T-tree의 구조

- *T*-tree는 배열 구조의 열거 트리로서 각 노드는 두 클래스의 카운트 값을 저장하고, 성장률(*GR*), 자식 노드를 참조하는 레퍼런스로 구성된다.

2.2.2 P-tree 및 P-tree 테이블 생성

서열 데이터로부터 *P*-tree 생성을 위한 생성 규칙은 다음과 같다.

[**정의 2**] *P*-tree 생성 규칙 : *P*를 현재 노드에 저장된 패턴이고, *P*'를 트리에 삽입할 새로운 패턴이라고 한다.

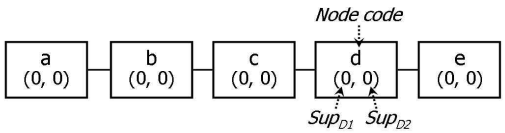
- 규칙 1 : $P=P'$ 일 때, 패턴이 동일하므로 단순히 각 클래스의 카운트 값을 증가시킨다.
- 규칙 2 : $P \supset P'$ 일 때, *P*에 대한 새로운 노드를 생성하고, *P*를 새로운 노드의 자식 노드로 대치한다.
- 규칙 3 : $P \supset P'$ 일 때, 새로운 *itemset*은 현재 노드의 상위 형제 노드이다.

- 규칙 4 : $P \subset P'$ 일 때, 새로운 *itemset*은 현재 노드의 자식이다.
- 규칙 5 : $P \subset P$일 때, 새로운 *itemset*은 현재 노드의 하위 형제이다.

예를 들어 표 1의 두 클래스의 데이터에 대한 *P*-tree를 구성한다고 할 경우, 데이터에 포함된 항목의 가지 수는 5이므로 크기 5의 배열로 표현할 수 있다(그림 5).

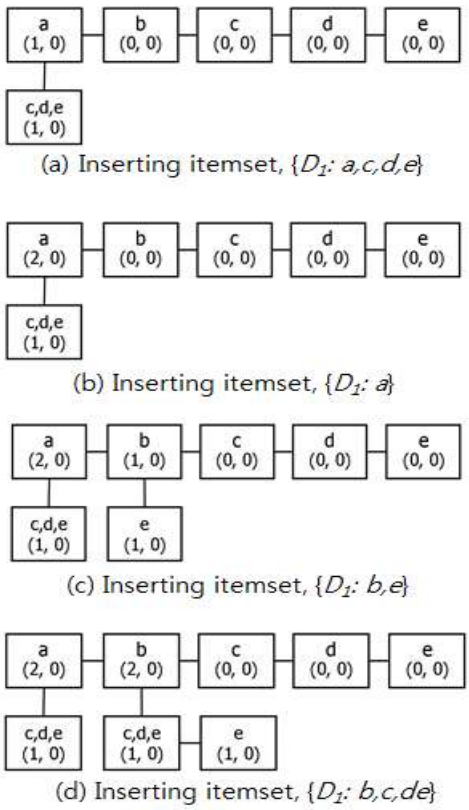
<표 1> P-tree 구성을 위한 데이터 항목

ID	Class	Itemset
1	D_1	*a c d e*
2	D_1	*a*
3	D_1	*b e*
4	D_1	*b c d e*
5	D_2	*a b*
6	D_2	*c e*
7	D_2	*a b c d*
8	D_2	*d e*



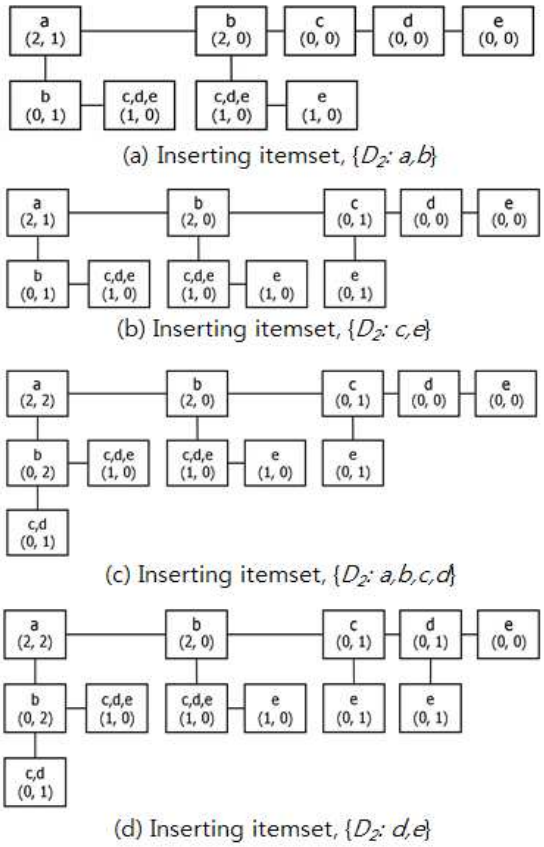
<그림 5> <표 1>의 데이터에 대한 초기 P-tree 배열 구조

첫 번째 데이터 {D_1: *a,c,d,e*}은 *a*와 연결되는 itemset *c,d,e*를 위한 노드를 생성하는 것으로 *P*-tree에 저장되며, 클래스 D_1쪽 카운트 값을 1로 설정한다(그림 6(a)). 두 번째 데이터 {D_1: *a*}는 이미 트리에 값이 저장되어 있으므로 단순히 클래스 D_1에 대한 카운트 값만을 증가시킨다(그림 6(b)). 데이터 {D_1: *b,e*}는 *b*에 연결되는 항목 *e*에 대한 노드를 생성하여 삽입시킨다(그림 6(c)). 마지막 {D_1: *b,c,d,e*}는 항목 b가 이미 트리에 저장되어 있으므로 *b*의 카운트를 증가시키고, itemset *c,d,e*에 대한 새로운 노드(dummy 노드)를 추가시키며, 이전의 노드인 *e*와는 형제 노드가 된다(그림 6(d)).



<그림 6> 클래스 D_1에 대한 P-tree 생성과정

또 다른 클래스 D_2에 대한 *P*-tree 생성은 현재까지 생성된 트리에 추가적으로 클래스 분포를 고려하여 진행된다. {D_2: *a,b*}는 항목 *a*의 D_2 클래스쪽 카운트를 1로 하고 *b*에 대한 더미 노드를 추가한다(그림 7(a)). {D_2: *c, e*}는 *c*에 연결되는 *e*에 대한 새로운 노드를 추가하는 것으로 삽입된다(그림 7(b)). 데이터 {D_2: *a,b,c,d*}는 prefix인 항목 *a*와 *b*에 대한 카운트를 증가시키고 itemset *c,d*에 대한 노드를 생성한다(그림 7(c)). 마지막 {D_2: *d, e*}는 *d*에 연결되는 *e* 노드를 생성하여 트리에 삽입한다(그림 7(d)).



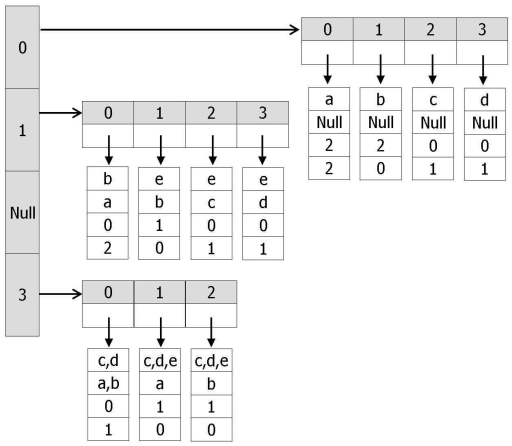
<그림 7> 클래스 D_2에 대한 P-tree 생성과정

표 1에 대한 *P*-tree 생성 후에 그림 8과 같은 *P*-tree 테이블을 생성한다. 인덱스 '0'은 1-*itemset*에 대한 노드들의 클래스 분포를 저장하고, 인덱스 '1'은 2-*itemset*, 인덱스 '3'은 4-*itemset* 노드들의 클래스 분포 값을 저장한다. 3-*itemset* 노드는 존재하지 않으므로 인덱스를 'Null'로 지정한다. *P*-tree 테이블의 내부 구조는 그림 8과 같다.

P-tree 테이블은 *T*-tree 구성을 위해서 사용되며, 메모리에 로딩되어 *T*-tree 구축의 성능 향상을 기대할 수 있다.

2.2.3 T-tree 생성 및 EPs-TFP 기법

기존의 *EPs* 기법들은 많은 출현패턴들을 생성하며, 이 패턴들에는 서로 다른 클래스간 중복 패턴들을 포함한다. 따라서 *P*-tree 테이블로부터 *T*-tree 구성 방법 전에 분류를 위한 필수 출현패턴(*eEPs*: essential Emerging Patterns)만을 추출하기 위한 다음과 같은 조건을 정의 한다.



<그림 8> 생성된 P-tree 테이블

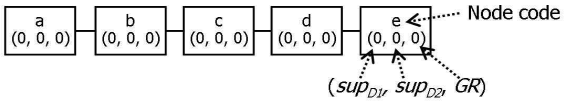
[**정의 3**] 중복패턴 제거를 위한 조건 정의, *eEPs* 선택을 위한 조건;

- $sup(X) \geq \sigma$, 최소지지도 σ 이상의 출현패턴
- $GR(X) \geq \rho$, 최소성장률 ρ 이상의 출현패턴
- 조건 $\forall Y \subset X$, $GR(Y) < GR(X)$ 을 항상 만족하는 출현패턴 X

위의 정의에서 첫 번째 조건은 모든 출현패턴은 사용자가 지정한 최소 지지도(support)를 만족해야 한다는 것이고, 두 번째 조건은 최소지지도를 만족한 패턴들이라도 최소 성장률(growth rate) 또한 만족해야 한다는 것이다. 세 번째 조건은 Y가 X에 부분 패턴일 경우, 기존 출현패턴 마이닝 알고리즘은 X 패턴을 제거 한다. 그러나 만약 최소성장률이 X가 더 클 경우, X 패턴은 필수 출현패턴이므로 제거 하지 않고 분류규칙으로 사용한다는 조건이다.

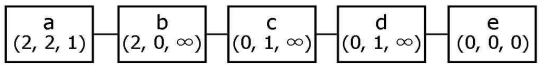
T-tree 역시 *P*-tree와 같은 열거 트리이고, 메모리에 저장된 *P*-tree 테이블을 스캔하여 클래스 카운트 및 성장률을 노드에 저장한다. 패턴 생성은 하위 노드로부터 역으로 상위 노드로의 운행으로 패턴들을 생성한다. 다음은 그림 8의 *P*-tree 테이블로부터 *EPs-TFP*를 적용하여 *eEPs*를 추출하는 예이다.

먼저, 사용자가 최소지지도를 σ=2, 최소성장률을 ρ=2로 설정할 경우, *P*-tree 생성과 유사하게 *T*-tree도 그림 9와 같이 총 항목의 개수인 5개 배열로 구성된다.



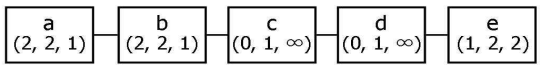
<그림 9> 초기 T-tree의 5개 배열 구조

메모리의 *P*-tree 테이블로부터 *T*-tree는 *1-itemset*의 각 클래스 카운트 값과 성장률을 포함하여 구성한다. 예를 들어 *P*-tree 테이블의 인덱스 '0'인 항목 *a*는 두 클래스 카운트 2, 2를 저장하고 성장률 2/2=1를 *a*(2,2,1) 형식으로 *T*-tree의 첫 노드에 삽입한다. 같은 방식으로 인덱스 '0'에 항목 *b, c, d*까지의 *T*-tree 삽입 결과는 그림 10이다.



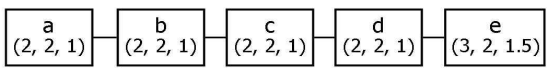
<그림 10> 인덱스 '0'의 1-*itemset*에 대한 *T*-tree 삽입 결과

또한, 1-*itemset*은 *P*-tree 테이블의 인덱스 '1'을 다시 스캔하여 항목 *b*(0,2)의 카운트 값을 저장하고 항목 *b*의 전체 성장률을 다시 계산하여 갱신한다(이전 단계의 항목 *b*(2,0) + 현재 삽입하려는 항목 *b*(0,2) = *b*(2,2), 따라서 갱신된 성장률은 2/2=1 이므로 노드에는 *b*(2,2,1)가 저장된다). 같은 방식으로 항목 *e*가 *P*-tree 테이블 인덱스 '1'의 세 노드에 저장되어 있으므로 이를 합친 *e*(1,2,2)를 저장한다(그림 11).



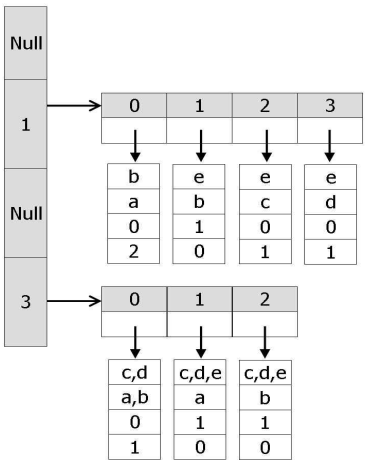
<그림 11> 인덱스 '1'의 1-*itemset*에 대한 *T*-tree 삽입 결과

1-*itemset*에 대한 마지막 인덱스 '3'에서의 항목 *c*(2,1), *d*(2,1), *e*(2,0)에 대한 클래스 카운트를 고려하여 성장률을 계산하여 노드를 그림 12와 같이 갱신한다.



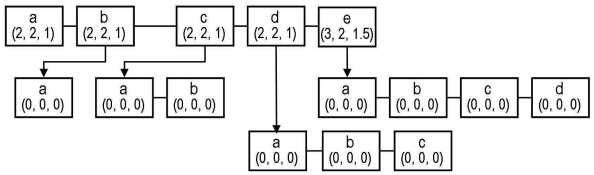
<그림 12> 인덱스 '3'의 1-*itemset*에 대한 *T*-tree 삽입 결과

P-tee 테이블을 이용하여 모든 1-*itemset*을 노드에 저장한 후에, 더 이상 1-*itemset*을 고려하지 않으므로 *P*-tree 테이블의 인덱스 '0' 레벨을 제거한다. 다음으로 2-*itemset*에 대한 노드 삽입이 진행되며, 이때 참조되는 *P*-tree 테이블은 다음과 같다.



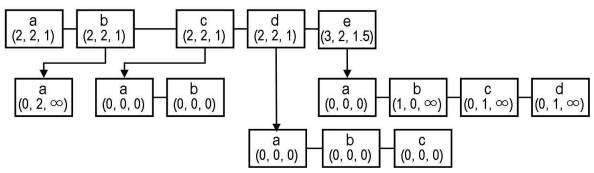
<그림 13> 인덱스 '0'이 제거된 P-tree 테이블

2-*itemset*에 대한 모든 가능한 조합을 전 단계의 트리에 그림 14와 같이 생성한다.



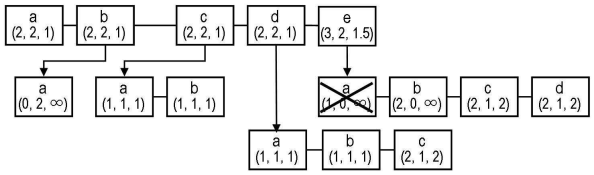
<그림 14> 2-*itemset*을 위한 T-tree 구조

먼저, 인덱스 '1'의 2-*itemset*인 {*a,b*}(0,2), {*b,e*}(1,0), {*c,e*}(0,1), {*d,e*}(0,1)의 클래스별 카운트값과 성장률을 계산하여 저장 한다(그림 15).



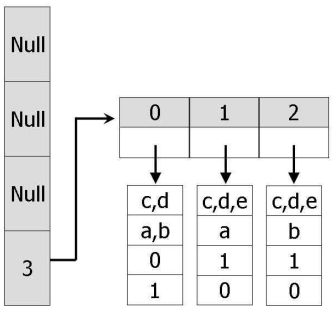
<그림 15> 인덱스 '1'에 대한 T-tree 노드 생성

다음으로 인덱스 '3'의 첫 레코드인 항목 *a,b,c,d*의 가능한 2-*itemset* 조합인 {*a,c*}(0,1), {*b,c*}(0,1), {*a,d*}(0,1), {*b,d*}(0,1), {*c,d*}(0,1)에 대해 성장률을 계산하여 이전 단계 트리에 갱신한다. 같은 방법으로 두 번째, 세 번째 레코드의 2-*itemset* 조합을 생성하여 카운트 및 성장률을 그림 16과 같이 *T*-tree에 저장한다. 또한 이 단계에서 항목 {*a,e*}(1,0, ∞)는 두 클래스의 카운트를 합한 값이 1이므로 정의 3의 첫 번째 조건인 최소지지도 $\rho = 2$를 만족 못하므로 삭제시킨다.



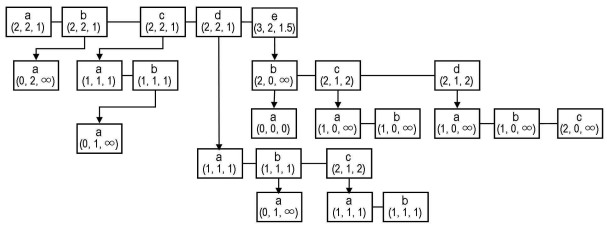
<그림 16> 2-*itemset*에 대한 T-tree의 생성결과

2-*itemset*까지 *T*-tree를 구성한후 인덱스 '1'에 레코드를 제거한 *P*-tree 테이블은 다음과 같다.

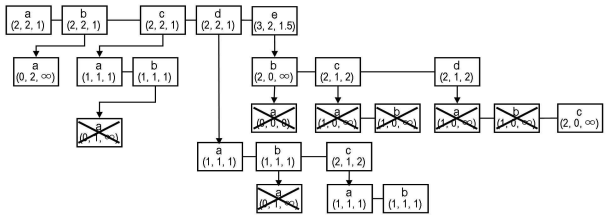


<그림 17> 인덱스 '1'이 제거된 P-tree 테이블

인덱스 '3'에 대한 3-*itemset*의 노드가 삽입된다. 첫 번째 레코드에 대해서, {*a,b,c*}(0,1), {*a,b,d*}(0,1), {*a,c,d*}(0,1), {*b,c,d*}(0,1)를 삽입하고. 두 번째 레코드인 {c,d,e} 노드에 대해서 {*a,c,d*}(1,0), {*a,c,e*}(1,0), {*a,d,e*}(1,0), {*c,d,e*}(1,0)를 삽입한다. 세 번째 레코드 역시 노드 {*c,d,e*}와 부모 {*b*} 노드를 스캔하여, {*b,c,d*}(1,0), {*b,c,e*}(1,0), {*b,d,e*}(1,0), {*c,d,e*}(1,0)를 저장한다. 모든 3-*itemset*에 대한 *T*-tree 구성 후에 현재 트리에 대한 전지(pruning)를 필수 출현패턴 조건을 참고하여 수행한다. 3-*itemset* 저장 후의 트리 구조는 그림 18이고 필수 출현패턴 조건에 대한 노드 제거 후의 *T*-tree는 그림 19와 같다.

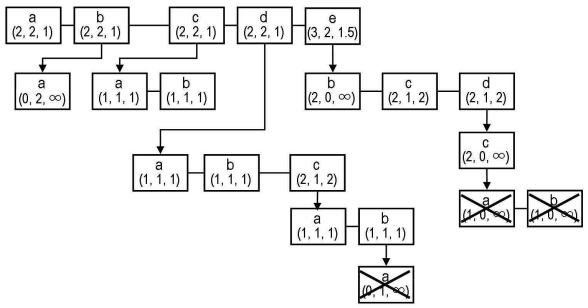


<그림 18> 3-itemset 항목 삽입 후의 T-tree



<그림 19> 필수 출현패턴을 위한 트리 전지

마지막으로 인덱스 '3'의 노드는 각각 {*c,d*}, {*c,d,e*}, {*c,d,e*}를 포함한다. 따라서 4-*itemset* 구성이 가능하며, 트리에 {*a,b,c,d*}(0,1), {*a,c,d,e*}(1,0), {*b,c,d,e*}(1,0)를 삽입할 수 있다. 그러나 모든 4-*itemset*의 조합은 최소지지도 조건을 만족하지 못하므로 제거된다.



<그림 20> 4-*itemset* 삽입 및 전지 후의 T-tree

T-tree의 구성이 완료 후에 정의 3에 따라 최소성장률, ρ=2를 만족하지 않는 모든 패턴들을 제거한다. 표 2는 *EPs-TFP*를 이용한 필수 출현패턴(*eEPs*) 추출의 결과이다. 또한 표 2에서 패턴 {*c,d,e*}는 부분 패턴인 {*c,d*}, {*c,e*}, {*d,e*} 보다 더 높은 성장률을 가지므로 정의 3의 세 번째 조건을 만족하는, 중복패턴이 아닌 필수 패턴으로 추출된다.

<표 2> *EPs-TFP*를 통한 필수 출현패턴

	eEPs	σ	ρ	Class
1	{c,d}	2	2	D_1
2	{b,e}	2	∞	D_1
3	{c,e}	2	2	D_1
4	{d,e}	2	2	D_1
5	{c,d,e}	2	∞	D_1
6	{a,b}	2	∞	D_2

3. eEPs를 이용한 Disorder 구역 분류

이 절에서는 *EPs-TFP* 기법을 이용하여 추출된 disorder 구역의 출현 패턴과 order 구역 출현 패턴으로 단백질 1차 서열을 disorder/order 구역으로 분류하는 방법을 기술한다.

Disorder 및 order 구역에서의 필수 출현패턴들은 분류하고자 하는 단백질 1차 서열을 정규화된 score 기반의 예측 과정과 최종 목적 클래스인 disorder 구역의 예측 결과에 대한 후처리(post-processing) 단계로 구성된다.

3.1 eEPs를 이용한 단백질 1차 서열 분류

모든 필수 출현패턴 생성 후, 새로운 데이터에 대한 분류는 [20]에서 제안한 score를 계산하여 가장 높은 score 값을 가지는 클래스로 분류하게 된다. 분류를 위한 score 계산식은 다음과 같다.

$$score(s, C) = \sum_{e \subseteq s, e \in E(C)} support_c(e) \cdot \frac{growth\ rate(e)}{growth\ rate(e)+1} \quad (2)$$

여서기 s는 분류될 서열 데이터 인스턴스이고, $E(C)$는 클래스 C에서 발견된 필수 출현패턴이다. 예를 들어 두 클래스 집합 D_1, D_2에 대한 출현 패턴이 각각 D_1={(a,e):(50%:25%), (d,e):(50%:25%)}, D_2={(a,d):(25%:50%)이고, 분류될 데이터가 s={a,d,e}이라고 가정하면, 두 클래스의 출현패턴이 s를 포함하므로 각각의 score를 계산한다. D_1, D_2에 해당되는 출현 패턴의 score들은;

$$score_1(s, D_1) = 0.5 \times \frac{2}{2+1} = 0.67,$$

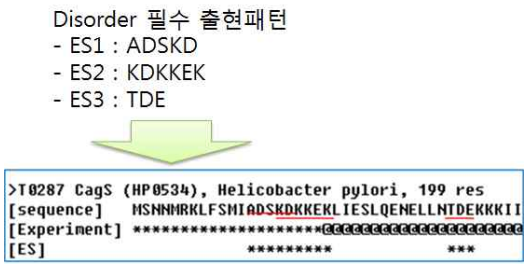
$$score_2(s, D_2) = 0.5 \times \frac{2}{2+1} \approx 0.33$$ 이며, $score_1 > score_2$ 이므로 s는 D_1의 클래스로 분류된다.

그러나 한 클래스의 *eEPs*의 개수가 편중되게 많을 경우, 상대 클래스 보다 score가 현저하게 더 높게 되는 경우가 발생되며, 이는 예측의 정확도에 영향을 주게 된다. 따라서 클래스 별 출현패턴의 균형을 고려하기 위해 기존 score식을 정규화한 *nSore*(normalized score)를 적용한다.

$$nScore = \frac{score}{median\_score} \quad (3)$$

*nSore*는 score 값을 해당 클래스에 속하는 중간 값(median)으로 나누었을 때의 값을 말한다.

단백질 서열에서 disorder/order 구역의 예측은 슬라이드 윈도우를 이용하여 서열 비교로서 찾아낸다. 예를 들어 disorder 필수 출현패턴을 이용한 구역 예측의 경우, 그림 21에서처럼 *eEPs*가 "ADSKD", "KDKKEK", "TDE"라면 슬라이딩 윈도우로 찾아낸 disorder 구역은 *ES*로 심볼 "*"로 표현한다.



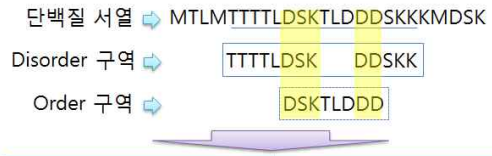
<그림 21> Disorder 구역 예측 결과 예

또한, Experiment라는 서열은 실제 실험을 통해 얻어진 결과로서 "@"는 order 구역이다.

3.2 Disorder/order 중첩 구역의 후처리

그림 21에서 disorder 구역의 패턴인 "TDE"는 실제

실험을 통해 얻어진 결과 서열에서는 “@”로 order 구역에 속하는 것으로 표현되었다. 이렇듯 두 구역으로 예측한 부분이 겹칠 수 있다. 이 논문의 주목적은 disorder 구역을 예측 하는 것이므로 disorder 구역 예측의 정확도를 보장 하는 것이 가장 중요하다. 따라서 disorder 구역 예측의 정확도를 보장하기 위하여 disorder쪽 *eEPs*로 예측된 부분 외에 두 클래스에 겹치는 부분에 대해서는 3.1절에의 식 3의 *nScore*를 계산하여 그 값이 큰 클래스쪽으로 예측/분류한다. 두 클래스에 겹쳐진 부분에 대한 간단한 계산 및 예측 과정을 예제로 설명하면 그림 22와 같다.



Disorder 구역		Order 구역	
eEPs	*nSore*	*eEPs*	*nScore*
TTTLDSK	0.5	DSKT	0.6
LDS	0.8	KT	0.8
DDSKK	0.5	TLDDD	0.3

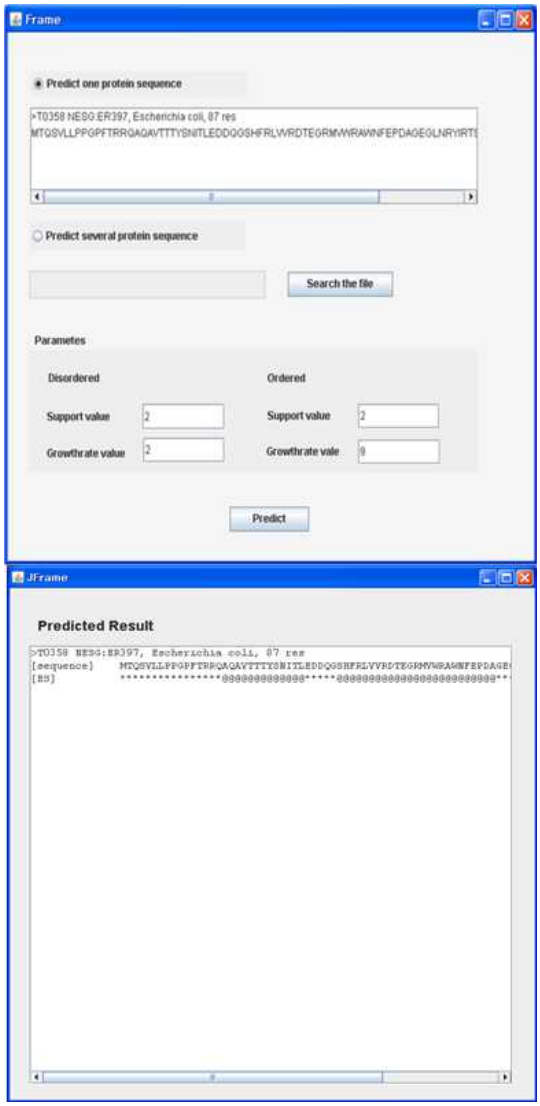
<그림 22> 중첩 구역을 포함한 분류 예제

위 그림에서 disorder와 order에서 중첩 서열은 부분 서열인 “DSK”와 “DD”두 부분이 있다. 이 부분 서열에 대해서 *nScore*의 값은 다음과 같다.

- 첫 번째 중첩 부분 : “DSK”
  · nScore(Disorder)=0.5+0.8=1.3 (“TTTLDSK” 와 “LDS”),
  · Score(Order)=0.6+0.8=1.4 (“DSKT”와 “KT”) 이므로 “DSK”는 order구역으로 분류한다.
- 두 번째 중첩 부분 : “DD”
  · nScore(Disorder)=0.5 (“DDSKK”),
  · nScore(Order)=0.3 (“TLDDD”) 이므로 “DD”는 disorder 구역이다.

그림 23은 이 논문에서 제안한 *EPs-TFP* 기법을 구현한 화면이다. 그림에서 첫 번째 윈도우 창은 초기 서열 데이터의 입력 화면이며, 아래의 윈도우 창은 *EPs-TFP* 수행 결과를 보여준다.

분류 하려는 서열 데이터 입력 시에는 기존의 구역 분류 프로그램들처럼 하나의 서열 또는 파일을 이용한 다량의 서열을 정렬시켜 분류 할 수 있다. 또한 사용자는 임계값인 최소지지도와 최소성장률을 조절하여 휴리스틱하게 여러 실험이 가능하다.



<그림 23> *EPs-TFP* 알고리즘 구동 화면

4. 실험 평가

4.1 데이터 수집 및 전처리

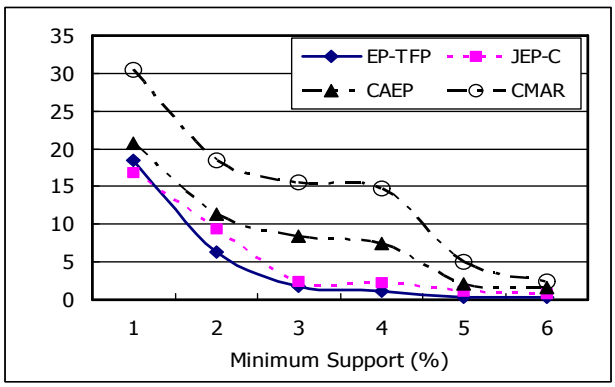
단백질 disorder 분류/예측 모델의 실험 평가를 위한 훈련 데이터(training data)로 disorder 서열 데이터, order 서열 데이터를 사용한다. Disorder 데이터는

Disprot(version 4.9) [21]에서 추출하였고 order 데이터는 PDB [22]에서 추출하였다. Disorder dataset와 order dataset의 모든 서열 데이터는 중복되지 않으며 25% 보다 높은 pairwise identity을 가진다. Disorder는 523개 서열 데이터, 1,195개 구역(region), 67,555개 잔기를 포함하고 있으며, order는 구조가 명확하고 길이가 최소 80개 잔기인 서열들을 추출하였고 290개 구역, 67,555개 잔기를 포함한다. 두 클래스 데이터의 편중 문제에 대해서는 평가 기준치가 정확하게 표현되지 않는 것을 방지하기 위하여 테스트 데이터로 CASP 7의 96개 단백질 서열 중 36개 서열과 Disprot (Ver. 4.9)에서의 10% 샘플 데이터인 40개의 단백질 서열을 사용한다.

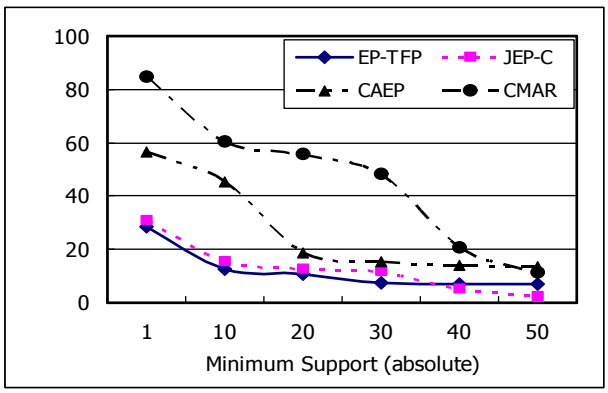
CASP 7 [23], [24]는 단백질 disorder 구역 예측에 테스트 데이터로 많이 사용되나 CASP 7의 단백질 서열에는 disorder 서열이 너무 적게 포함되어 있으므로 긴 disorder 구역을 많이 포함하고 있는 Disprot에서 샘플 데이터로 40개의 추가 데이터를 추출하여 총 76개 서열이 포함되도록 한다.

4.2 EPs-TFP 성능 평가 및 분류 결과

EPs-TFP 분류 모델의 성능은 최소지지도, 최소성장률의 임계값에 대한 확장성(scalability) 테스트를 수행한다. 단백질 서열의 데이터 분석의 효율적 메모리 관리와 빠른 출현패턴 발견에 대한 실험으로 기존의 패턴-기반 분류 기법인 *JEP-C* [25], *CAEP* [26], *CMAR* [27]의 알고리즘들을 상대적, 절대적 지지도 변화에 대한 실행시간을 비교한다. *JEP-C*, *CAEP*는 기존의 출현패턴 기법으로 EPs-TFP 알고리즘의 수정으로 구현 가능하다. 먼저 *P*-tree, 및 *T*-tree 구조를 적용하지 않으면서, *JEP-C*를 위해서는 성장률이 무한대값을 가지는 jumping 출현패턴만을 사용하는 것으로 수정가능하고, *CAEP*는 중복패턴 제거 조건을 제거하여 구현 가능하다. 연관적 분류기법인 *CMAR*은 [28]에서 제공하는 오픈소스를 활용하여 실험에 사용한다.



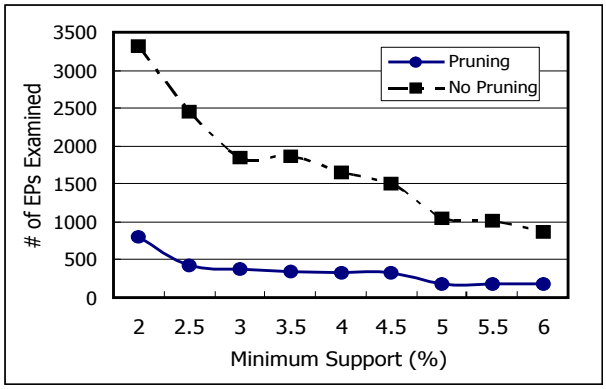
<그림 24> 상대적 지지도 변화에 대한 실행시간



<그림 25> 절대적 지지도에 변화에 대한 실행시간

상대적, 절대적 지지도 변화에 대한 실행시간 비교 결과는 기존의 출현패턴 마이닝 보다 *P*-tree 테이블을 활용한 *T*-tree 구성이 빠른 탐색 시간을 가짐을 알 수 있다. 그러나 *FP*-tree 방식의 CMAR의 경우, 출현패턴 마이닝과는 다르게 신뢰도 및 DB coverage라는 추가적인 임계값을 가지므로 이 실험에서는 간접적인 비교만이 가능할 뿐이다.

그림 26은 정의 3의 조건들에 따라 중복패턴을 제거 할 경우와 기존 유지방식과의 차이를 보여준다. 이는 긴 서열의 단백질 구역 분류에 있어 예측의 정확도와 모델 생성 속도에 영향을 주는 중복패턴 제거가 효율적이라는 것을 보여준다.



<그림 26> 중복패턴 제거의 출현패턴 개수 비교

*EPs-TFP*의 확장성 테스트에 이어 필수 출현패턴을 이용한 분류 모델의 정확성을 평가하기 위해서 민감도(sensitivity), 특이도(specificity) 및 정확도(precision)를 기반으로 평가한다. 민감도는 단백질 서열에서의 실제 disorder 구역의 잔기들에서 disorder 구역으로 정확하게 예측 되었는가에 대한 비율이며, 특이도(specificity)는 단백질 서열에서의 실제 order 구역의 잔기들에서 order 구역으로 정확하게 예측된 잔기가 차지하는 비율을 말하는데 이 기준은 전체 단백질 서열에서 order구역을 예측해내는 정도를 보여준다. 정확도는 disorder 구역으로 예측된 잔기들 가운데서 실제로 disorder 구역의 잔기가 차지하는 비율을 말하며 이는 disorder 잔기들에서의 예측 정확성을 나타낸다.

$$\text{민감도} = \frac{TP}{TP+FN} \quad (4)$$

$$\text{특이도} = \frac{TN}{TN+FP} \quad (5)$$

$$\text{정확도} = \frac{TP}{TP+FP}, \quad (6)$$

TP(True Positive), TN(True Negaitive), FP(False Positive, FN(False Negative)

분류모델의 성능평가는 [12]에서 소개된 기존의 예측 프로그램들과 같은 데이터를 이용하여 얻은 결과와 *EPs-TFP* 기법을 비교한다.

설험 결과 표 3은 단백질 아미노산 서열을 기준으로 disorder 구역을 분류하였을 때의 혼잡매트릭스 이며, 표 4 민감도, 특이도, 정확도 실험 결과이다.

<표 3> Disorder/order 분류의 혼잡매트릭스

단위	구역	예측된 구역	
		Disorder	Order
%	Disorder	73.8	24.4
	Order	30.2	71.5

<표 4> Disorder 분류/예측 결과 비교(%)

	민감도	특이도	정확도
EMBL_hot	42.4	81.9	63.5
EMBL_coil	72.5	52.2	53
EMBL_remark	27.9	75.8	64.8
EPs-TFP	73.6	69.5	74.2

*EPs-TFP*를 통한 disorder 구역 분류 결과, 표 3과 표 4로부터 단백질 아미노산에 대해서, 민감도와 정확도는 가장 높으면서 신뢰할 수 있을 정도의 특이도를 가진다는 것을 알 수 있다.

5. 결 론

이 논문에서는 *P*-tree와 *T*-tree 개념을 단백질 기능 예측을 위한 분류 기법으로 활용하기 위해서 출현패턴 마이닝 방법으로 확장 및 개선한 *EPs-TFP*를 제안하였다. *EPs-TFP* 알고리즘은 단백질 서열 데이터로부터 단백질 기능 예측에 중요한 요소인 disorder 구역을 분류해 낸다. 또한 제안된 기법은 서열 기반 단백질 기능 예측 방식의 단점인 패턴 발견 속도 문제를 해결하고 *P*-tree 테이블의 활용으로 효율적인 메모리 관리가 가능하며, 중복패턴 제거 단계를 거치므로 필수 출현패턴만을 분류 모델 생성에 이용한다. 따라서 기존 출현패턴 분류 기법보다 더 정확한 결과를 얻을 수 있다. 그러나 제안한 *EPs-TFP* 방법도 단백질의 필수 출현패턴 기반 분류 모델 구축에 있어서, 단백질 잔기들을 두 클래스 서열 데이터에서의 발생빈도에 근거하여 분류하므로 휴리스틱 파라미터인 최소지지도 및 최소성장률과 같은 임계값의 최적 값을 찾아야 하는 문제를 가지고 있다.

참 고 문 헌

[1] J.F. Gibrat, T. Madej, and S.H. Bryant, “Surprising similarities in structure comparison,” Curr. Opin. Struct. Biol., vol. 6, pp.377-385, 1996.

[2] 안명상, 고정환, 유재수, 조완섭, “단백질 상호작용 네트워크에서 연결노드 추출과 그 중요도 측정,” 한국산업정보학회논문지, vol. 12, no. 5, pp.1-13, 2007.

[3] S. Maslov, and K. Sneppen, “Specificity and stability in topology of protein networks,” Science, vol. 296, pp.910-913, 2006.

[4] F. Ferron, S. Longhi, B. Canard, and D. Karlin, “A practical overview of protein disorder prediction methods,” Proteins: Structure, Function, and Bioinformatics, vol. 5, pp.1 - 14, 2006.

[5] DT. Jones, and JJ. Ward, “Prediction of disordered regions in proteins from position specific score matrices,” Proteins, vol. 53, pp.573-578, 2003.

[6] K. Peng, P. Radivojac, S. Vucetic, AK. Dunker, et al., “Length dependent prediction of protein intrinsic disorder,” BMC Bioinformatics, vol. 7 online, 2006.

[7] S. Hirose, and K. Shimizu, “POODLE-L: a two-level SVM prediction system for reliably predicting long disordered regions,” Bioinformatics, vol. 23, pp.2046-53, 2007.

[8] T. Ishida, and K. Kinoshita, “PrDOS: prediction of disordered protein regions from amino acid sequence,” Nucleic Acids Research, vol. 35, pp.460-464, 2007.

[9] ZR. Yang, R. Thomson, P. McNeil, and RM. Esnouf, “RONN: the bio-basis function neural network technique applied to the detection of natively disordered regions in proteins,” Bioinformatics, vol. 21, pp.3369-3376, 2005.

[10] J. Liu, H. Tan, and B. Rost, “Loopy proteins appear conserved in evolution,” Mol. Biol., vol. 322, pp.53-64, 2002.

[11] Z. Obradovic, K. Peng, S. Vucetic, P. Radivojac, et al., “Predicting intrinsic disorder from amino acid sequence,” Proteins, vol. 53, pp.566 - 572, 2003.

[12] R. Linding, LJ. Jensen, F. Diella, P. Bork, TJ. Gibson, RB. Russell, “Protein disorder prediction: implications for structural,” Proteomics, vol. 11, pp.1453-1459, 2003.

[13] R. Linding, LJ. Jensen, F. Diella, P. Bork, et al., “Protein disorder prediction: implications for structural proteomics,” Structure. vol. 11, pp.1453-1459, 2003.

[14] J. Cheng, M. Sweredoski, P. Baldi, “Accurate prediction of protein disordered regions by mining protein structure data,” Data Mining and Knowledge Discovery, pp.213-222, 2005.

[15] J. Prilusky, C.E. Felder, T. Mordehai, E.H. Rydberg, et al., “FoldIndex: a simple tool to predict whether a given protein sequence is intrinsically unfolded,“ Bioinformatics, vol. 21, pp.3435 - 3438, 2005.

[16] P. Han, X. Zhang, ZP. Feng, “Predicting disordered regions in proteins using the profiles using amino acid indices,” BMC Bioinformatics, vol. 10 online, 2009.

[17] F. Coenen, P. Leng, and G. Goulbourne, “Tree Structures for Mining Association Rules,” Data Mining and Knowledge Discovery, vol. 15, pp.391-398, 2004.

[18] 최해원, “대용량 DNA서열 처리를 위한 서픽스 트리 생성 알고리즘의 개발,” 한국산업정보학회논문지, vol. 15, no. 1, pp.37-46, 2010.

[19] G. Dong, X. Zhang, L. Wong, J. Li, "Classification by aggregating emerging patterns," Int'l Conf. on Discovery Science, pp.30-42, 1999.

[20] 이헌규, 노기용, 류근호 정두영, “심혈관계 질환 진단을 위한 출현 패턴 기반 분류 기법,” 한국정보처리학회 16-D, pp.11-26, 2009.

[21] S. Vucetic, Z. Obradovic, V. Vacic, P. Radivojac, et al., “DisProt: A Database of Protein Disorder,”Bioinformatics, vol. 21, pp.137-140, 2005.

[22] U. Hobohm, C. Sander, “Enlarged representative

set of protein structures," Protein Science, vol. 3, p.522, 1994.
[23] J. Moult, K. Fidelis, A. Zemla, T. Hubbard, "Critical assessment of methods of protein structure prediction (CASP)-round 5," Proteins, vol.53, pp.334-339, 2003.
[24] J. Moult, K. Fidelis, B. Rost, T. Hubbard, et al., "Critical assessment of methods of protein structure prediction (CASP)-round 6," Proteins, vol. 61, pp.3-7, 2005.
[25] J. Li, G. Dong, and K. Ramamohanarao, "Making use of the most expressive jumping emerging patterns for classification," Knowledge and Information Systems, vol. 3, no. 2, pp.131-145, 2001.
[26] G. Dong, X. Zhang, L. Wong, and J. Li, "Classification by aggregating emerging patterns," Int'l Conf. on Discovery Science, Japan, pp.30-42, 1999.
[27] W. Li, J. Han and J. Pei, "CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules," ICDM 2001, pp.369-376, 2001.
[28] F. Coenen, "LUCS-KDD group, Dept. of Computer Science," The University of Liverpool, UK, "http://www.cSc.liv.ac.uk/~frans/KDD/," 2004.

이 헌 규 (Heon Gyu Lee)

- 정회원
- 경기대학교 전자계산학과 이학학사
- 충북대학교 전자계산학과 이학석사
- 충북대학교 전자계산학과 공학박사
- 한국전자통신연구원 융합기술연구부문 선임연구원
- 관심분야 : 데이터마이닝, 기계학습, 패턴인식, 바이오인포매틱스, 바이오매디컬, 데이터베이스, GIS 등

신 용 호 (Yong Ho Shin)

- 정회원
- 서울대학교 산업공학과 공학학사
- 한국과학기술원 산업 및 시스템공학과 공학석사
- 한국과학기술원 산업 및 시스템공학과 공학 박사
- 영남대학교 경영학부 조교수
- 관심분야 : 경영과학, 데이터마이닝, Machine Learning, 정보시스템, Formal Model 등

논 문 접 수 일 : 2012년 09월 25일
1차수정완료일 : 2012년 10월 22일
2차수정완료일 : 2012년 11월 20일
게 재 확 정 일 : 2012년 11월 30일