고정된 패턴 리스트를 사용한 단백질 2차 구조의 검색

나상준[O] 박상현
연세대학교 컴퓨터과학과
{sjna[O], sanghyun}@cs.yonsei.ac.kr

Searching Secondary Structure of Protein Using Fixed Pattern List

Sangjun Na[0] Sanghyun Park
Dept. of Computer Science, Yonsei University

요 약

단백질의 1차 구조를 통하여 생성되는 단백질 2차 구조는 3가지 타입 E, H, L 을 가지고 있다. 단백질 2차 구조는 선형적인 단백질 1차 구조를 공간적으로 형성한 것이며 단백질 2차 구조에 관한 연구는 단백질 기능 예측에 중요한 부분이다. 단백질 2차 구조는 3가지 타입이 각각 그룹을 이루어 나타나는 특징이 있다. 단백질 2차 구조의 이러한 특징을 이용하면 효과적인 검색이 가능하다. 기존의 연구에서는 시퀀스 전체와 질의를 스트링 기반으로 비교하는 방법과 단백질 2차 구조의 세그먼트 테이블을 이용하는 방법을 사용하였다. 하지만 이러한 방법은 검색 비용이 많이 드는 단점이 있다. 본 논문에서는 효과적인 단백질 2차 구조의 검색을 위하여 고정된 패턴을 정의하고 고정된 패턴을 사용하는 방안을 제시한다.

1. 서 론

단백질의 1차 구조를 통하여 생성되는 단백질 2차 구조는 3가지 타입 (alpha helices (H), beta sheets (E), turns or loops(L)) 을 가지고 있다[3]. 단백질 2차 구조는 선형적인 단백질 1차 구조를 공간적으로 형성한 것이며 단백질 2차 구조에 관한 연구는 단백질 기능 예측에 중요한 부분이다[6]. 단백질 2차 구조는 3가지 타입이 각각 그룹을 이루어 나타나는 특징이 있다. 단백질 2차 구조의 이러한 특징을 이용하면 효과적인 검색이 가능하다. 기존의 연구에서는 시퀀스 전체와 질의를 스트링 기반으로 비교하는 방법[2]과 세그먼트 테이블을 이용하는 방법[3]을 사용하였다. 하지만 이러한 방법은 검색의 비용이 많이 드는 단점이 있다. 이런 문제를 해결하기 위해서는 단백질 2차 구조 시퀀스와 질의를 비교함에 있어 좀 더 효과적인 방법이 필요하다. 이를 위해서 이 논문에서는 고정된 패턴 방법(FPL)을 사용한다. 단백질 2차 구조 에서는 3가지 타입만이 사용됨으로 일정한 길이의 패턴의 집합을 미리 정의하는 것이 가능하며 이를 인덱스로 사용함으로써 검색 시간을 줄일 수 있다. 고정된 패턴은 3가지 타입의 타입 가중치와 타입의 위치에 따른 위치 가중치를 이용한 연산을 통해 단백질 2차 구조 전체 시퀀스와 질의를 신속하게 비교할 수 있다. 또 여기서는 2차 구조 시퀀스를 고정된 패턴으로 구성할 때 슬라이딩 윈도우 방식을 사용함으로써 윈도우간의 연결성을 이용한 검색도 가능하게 한다.

본 논문에서는 고정된 패턴을 구성하여 2차 구조를 검색하는 기본적인 방법론인 FPL을 우선 제시하고 이를 사용하여 질의가 패턴보다 긴 경우와 질의에 와일드 카드가 포함된 경우에 대한 해결책도 아울러 제시한다.

2. 관련연구

단백질 2차 구조의 검색 기법은 [2][3]등에 제시 되었다. 하지만 이러한 방법은 검색에 관한 시간의 비용이 많이 들게 된다. [3]에서는 2차 구조 전체 시퀀스를 세그먼트 단위로 구분한 테이블을 이용하여 검색 하는 방법을 사용하였다. [3]에서 제시한 세그먼트 단위의 구분은 단백질 2차 구조가 3가지 타입으로 이루어져있는 특징을 이용한다는 점에서 장점을 가지고 있다. 하지만 [3]에서 제시된 방법을 이용하여 검색을 하게 되면 테이블을 조인하는 과정에서 검색에 관한 비용이 많이 들게 된다.

이 논문에서는 [2]와 [3]에서 제시한 방법을 개선하기 위하여 고정된 패턴을 사용하며 3가지 타입의 타입 가중치와 타입의 위치에 따른 위치 가중치를 이용한 연산을 통해 단백질 2차 구조 전체 시퀀스와 질의를 신속하게 비교할 수 있는 방법을 제시한다.

3. 고정된 패턴의 생성 방법

정의 1. FPL(Fixed Pattern List)

FPL 이란 지정된 윈도우의 사이즈에 따라 구성되는 고정된 패턴의 집합이다. :

(1)FPL을 구성하기 위해서는 그림1 에서와 같이 2차 구조 전체 시퀀스를 세그먼트 단위로 구분 한다.

(2)세그먼트는 S_k로 표시한다. K는 세그먼트의 위치를 나타내며 윈도우 안에 포함되는 세그먼트의 개수는 윈도우 사이즈와 동일하다.

(3)패턴을 생성하기 위하여 단백질 2차 구조에서 사용되는 3가지 타입 (alpha helices (H), beta sheets (E), turns or loops(L))에 각각 가중치를 부여한다. $E=0$ $H=1$ $L=2$ 를 부여한다.

(4)패턴의 사이즈가 k일 경우 윈도우의 위치에 따라 3^k을 부여한다. 예를 들어 패턴사이즈 4, HLEL의 패턴을 가지게 되면 $(3^4 \cdot 1 + 3^3 \cdot 2 + 3^2 \cdot 0 + 3^1 \cdot 2 = 105)$ 로 인덱스를 생성한다.

(5)해당 인덱스에는 윈도우 위치 (2차 구조 전체 시퀀스에서 패턴과 일치하는 서브 시퀀스가 검색되는 슬라이딩 윈도우의

시작 위치)가 저장된다.

(6)위의 과정을 거쳐 형성된 패턴 리스트는 다음과 같이 분류 되어 저장된다.

Pattern List =< (PI_1,W-List, C),(PI_2, W-List, C),…,(PI_N,W-List, C) >.

PI = 패턴 인덱스

W-List = 패턴인덱스에 속하는 윈도우의 위치 정보.

C = PI 에 속하는 세그먼트 개수 그룹. ■

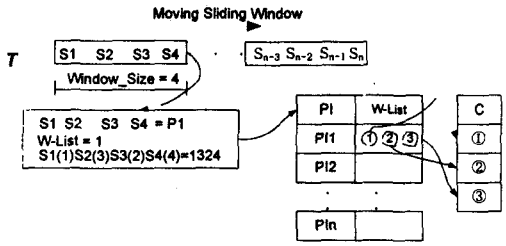

그림1. 고정된 패턴과 윈도우 위치

FPL 방법에서는 고정되는 패턴의 길이가 가변적이므로 2차 구조 전체 시퀀스에서의 타입 세그먼트와 개수를 유지하는 테이블을 생성하여 패턴의 길이가 정해지면 테이블의 행을 그룹화시켜 그림 1의 테이블을 생성하는 방법을 사용한다.

4. FPL를 이용한 검색 기법

정의1에서 말한 FPL방법을 이용하면 효과적인 검색이 가능하다. 기존의 스트링 매칭 알고리즘에서 사용되는 방법은 질의로 들어온 패턴으로 전체 시퀀스를 검색하는 방법을 사용한다. 전체 시퀀스의 길이를 m , 질의로 들어온 패턴의 길이를 n 이라 하면 검색에 걸리는 시간은 $O(mn)$ 의 비용이 들게 된다[2]. 대용량의 데이터베이스에서는 상당히 비효율적이다. 반대로 FPL 방법을 사용하면 사전 고정된 패턴 테이블을 사용하기 때문에 시간 비용을 줄일 수 있게 된다. 또 단백질 2차 구조는 사용되는 문자의 개수가 한정되어있으므로 각각의 문자에 가중치를 두어 타입을 구분할 수 있다.

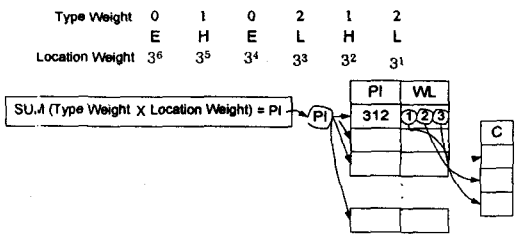

그림2. 패턴 검색 과정.

그림2처럼 질의에 포함된 패턴을 타입에 관한 가중치와 위치에 관한 가중치를 이용한 연산을 통하여 고정된 패턴 인덱스를 숫자를 통한 검색을 하게 된다. 패턴인덱스에는 동일한 패턴이 반복되지 않고 슬라이딩 윈도우로 2차 구조 시퀀스를 분류할 때 윈도우 위치정보가 PI와 연결되어 있으므로 두 개의 가중치를 이용한 결과 값을 가지는 모든 윈도우의 위치를 얻을 수 있게 된다. 또 PI와 연결되어 있는 W-List에 저장되어 있는 윈도우의 위치와 세그먼트의 개수 그룹이 저장되어 있는 테이블이 연결되어 있으므로 [3]에서 제시하였던 <H 1 1><E 1 2><L 2 3>과 같은 범위 질의의 경우 개수 그룹이 저장되어 있는 테이블의 C 값과 질의의 개수 범위의 비교를 통하여 간단하게 개수의 일치 성을 판단할 수 있게 된다. 위의 경우 <HEL, (11) (12)(23)>과 같이 구분할 수 있으며 개수 그룹이 저장되어 있는 테이블의 C값을 (11), (12), (23) 각각의 위치에서 비교하여 범위 안에 들어가는 개수 리스트를 검색할 수 있다.

5. 특정한 경우의 검색

단백질 2차 구조의 검색에서 생각할 수 있는 특정한 경우는 고정된 패턴의 길이보다 질의의 길이가 긴 경우와 특정위치에서 발생하는 와일드 카드의 경우이다. 여기서는 정의1에서 제시한 FPL 방법을 기반으로 이를 해결하고자 한다.

5.1 고정된 패턴의 길이보다 질의의 길이가 긴 경우의 처리

그림 2의 경우는 패턴의 길이와 질의의 길이가 같은 경우의 검색을 나타낸다. 이의 경우는 단순히 타입 가중치와 위치 가중치를 연산하여 결과값을 패턴인덱스와 비교만을 필요로 하는 간단한 경우이다. 하지만 질의의 길이가 패턴과 상이할 경우 다른 방법을 사용해야 한다. 기존의 연구에서는 두 개의 패턴을 단순히 오버랩핑 시키는 방법[2] 을 사용하였다. 하지만 여기서는 2차 구조 시퀀스를 분류할 때 사용한 윈도우가 슬라이딩 윈도우란 점을 착안 하여 패턴과 질의를 비교했을 경우 차이가 생기는 질의의 사이즈와 PI에 연결되어 있는 윈도우의 위치 리스트를 이용하여 문제를 해결한다. 예를 들어, 고정된 패턴의 길이가 10이고 질의의 길이가 12인 경우 질의와 패턴을 매칭시키면 질의 부분에 2만큼의 사이즈가 남게 된다. 그림3에서 보는 바와 같이 질의가 기존의 PI에 저장된 패턴보다 검은 사각형의 길이 만큼 긴 경우이다. 2차 구조 전체 시퀀스는 슬라이딩 윈도우를 사용하므로 그림1에서의 윈도우의 구성은

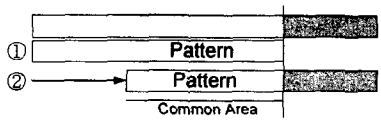

그림3. 긴 질의가 들어온 경우

(S1S2S3S4), (S2S3S4S5), (S3S4S5S6)이 되며 이동범위는 세그먼트 하나가 된다. 따라서 그림 3에서 ① 번의 초기 비교 패턴과 ② 번의 패턴은 공통 구역을 가지게 된다. 따라서 ①번 패턴의 윈도우를 질의의 검은 부분의 사이즈 만큼 이동을 시키면 ②번 패턴의 윈도우의 위치가 된다. 따라서 FPL 방법에서의 패턴 인덱스 테이블을 이용하면 ①번 패턴의 PI에 연결된 윈도우의 위치에서 검은 사각형 사이즈만큼 더해 주면 ②번 패턴의 윈도우의 위치를 얻을 수 있다. 또 검은 사각형안의 타입과 공통된 부분의 타입을 합쳐 타입 가중치와 위치 가중치를 이용해 ②번 패턴의 PI의 값을 연산하면 ①번 패턴의 윈도우에 연결된 질의의 범위 안에 드는 정확한 ②번 패턴의 윈도우를 찾을 수 있게 된다. 결과적으로 ①~②패턴의 윈도우 범위가 시퀀스에서 질의가 속하는 부분이 된다. 그림4에서 ①과 ②번 패턴을 검색한 후 이에 해당하는 윈도우 위치 리스트를 검색한다. 그림 4의 패턴의 윈도우 리스트에는 4개의 윈도우 위치가 저장되어 있다. ②번 패턴에서 검은 부분은 ①번 패턴이 이동한 사이즈와 같은 길이가 된다. 결국 ②번 패턴의 위치 리스트 중 ①번 패턴의 윈도우 위치에서 검은 사각형 사이즈 만큼 이동된 윈도우의 위치가 연결된 윈도우의 위치가 된다.

전체 질의를 α, ①번 패턴을 β, 검은 사각형을 δ , ②번 패턴의 윈도우 위치 값을 ρ, ①번 패턴의 윈도우 위치 값을 σ라 가정 하면 다음과 같은 식을 생각해 볼 수 있다.

질의에 포함되는 ②번 패턴 = (α – β) + Common Area· · · (1)
질의를 나타내는 윈도우의 범위 =
(ρ – σ)의 값이 검은 사각형의 사이즈와 같은 것.· · · (2)

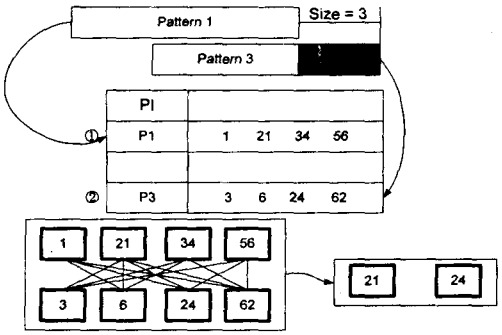

그림4. 연결된 윈도우 탐색

(1)을 통하여 ②번 패턴의 PI 값을 연산할 수 있으며 (2)를 통하여 PI값에 연결된 W-List 중 질의의 범위에 있는 윈도우 위치를 찾을 수 있게 된다. (1), (2)를 통하여 질의가 긴 특정 경우의 문제를 해결할 수 있게 된다.

5.2 질의에 와일드 카드가 속한 경우의 처리

단백질 2차 구조의 검색을 위한 질의에는 * 과 같은 와일드 카드를 사용할 수 경우가 있다. 와일드카드란 * 부분에 단백질 2차 구조의 세가지 타입 중 어느 것이 와도 무관한 경우이다. 하지만 전체 시퀀스의 구조를 세그먼트 단위로 분류해서 보면 같은 타입은 연속해서 나오는 경우가 없다. 예를 들어, *HHE*와 같이 *H* 타입이 연속해서 나올 수 없다. 따라서 이와 같은 특성을 와일드 카드 문제를 해결하는데 사용한다.

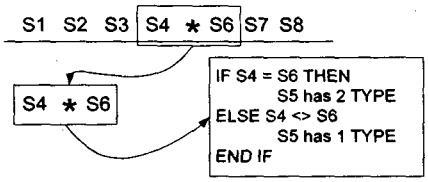

그림 5.와일드 카드의 경우

그림5의 경우처럼 세그먼트 S4 S6사이에 *가 오는 경우 2가지 경우를 생각할 수 있다. 첫 번째 경우는 S4와 S6의 타입이 같은 경우이고 두 번째는 S4와 S6의 타입이 다른 경우이다. 위에서 말한 바와 같이 같은 타입이 연속해서 나올 수 없으므로 전자는 2개의 타입이 후자의 경우는 한 개의 타입만이 올 수 있다. 이 경우 아래와 같은 과정을 거쳐 해당 PI와 W-List을 얻어낸다.

(1)*에 해당하는 위치 가중치는 일정하므로 * 부분을 제외한 S1~S8의 값을 계산한다.
(2)위치에 따른 * 에 속할 수 있는 타입의 위치 가중치와 타입의가중치를 계산한다. (1)의 결과 값에 계산된 값을 더하여 PI의값을 계산한다.
(3)위의 PI값을 이용하여 해당 질의의 범위 안에 속하는 모든 패턴을 검색한다.

논문에서 제시하고 있는 방법을 이용하면 간단하게 와일드카드가 포함된 경우를 해결할 수 있게 된다.

6. 실험 계획

실험에서 2차 구조를 생성하기 위하여 사용한 단백질 1차 구조는 PIR[9]에서 사용된 단백질 1차 구조를 사용하였다. 2차 구조의 변환에는 predator를 사용하였다. Predator를 사용하여 단백질 1차 구조를 2차 구조로 변환하는데 상당한 시간이 걸리므로 PIR의 전체 시퀀스 중에서 총 10,000개의 시퀀스만 2차 구조로 변환을 실시하였다. 논문에서 제시된 방법을 수행하기 위하여 predator를 통하여 생성된 2차 구조를 FPL방법으로 구성하였으며 가변적인 질의의 사이즈를 통한 정확한 검색의 경우를 구현하였다. 질의의 경우는 5가지로 분류하여 10,20,40,80,100의 경우를 FPL로 구성하였다. 현재 구현된 FP를 이용하여 두 가지의 특정 경우의 검색에 관한 내용을 실험 중이며, [3]의 방법과의 성능 비교를 실험할 계획이다.

7. 결론

단백질 1차 구조를 통하여 생성된 단백질 2차 구조의 전체 시퀀스에서 패턴과 매치되는 서브 시퀀스를 찾는 것은 이종의 단백질에서 공통의 부분을 찾아 두 단백질간의 연관 규칙을 발견하는데 있어서 필수적인 연산이다. 기존의 연구에서는 패턴과 전체 시퀀스를 대응시키는 방법과 단백질 2차 구조의 세그먼트 테이블을 통한 조인 방법을 제시하였다. 그러나 패턴과 전체 시퀀스를 대응시키는 방법은 검색에 있어 시간의 비용이 많이 드는 단점이 있다. 또 테이블을 이용하는 방법은 세그먼트 단위로 구분하여 패턴과 일치하는 부분을 검색하는 체계적인 방법을 제시하였지만 다수의 조인이 필요하게 되며 조인 기반의 검색 방법은 인덱스 사용을 용이하게 하지만 조인 자체에 많은 비용이 소요되는 단점이 있다. 논문에서 제시한 FPL 방법은 고정된 패턴과 질의와의 검색을 숫자를 통하여 비교함으로써 기존의 방법보다 시간의 비용 면과 간결성에서 효율적이다. 또 슬라이딩 윈도우를 사용한다는 점에서 질의의 길이 변화에 유동적으로 대처할 수 있다는 장점이 있다. 향후에는 고정된 패턴을 통하여 생성되는 인덱스의 사이즈를 줄이는 것에 관한 연구를 통해 좀 더 효율적인 단백질 2차 구조의 검색을 간결화 시키는 방안을 연구할 예정이다.

참고문헌

[1] O. Çamoğlu, Tamer Kahveci, A K. Singh "towards index-based similarity search for protein structure databases" IEEE Computer Society Bioinformatics Conference (CSB'03)
[2] D.Giladi. Algorithms on Strings, Trees, and Sequence: Computer Science and Computational Biology. Cambridge University Press, 1 Edition, January 1997.
[3] L. Hammel, J M. Patel, "Searching on the Secondary Structure of Protein Sequences" in VLDB, 2002
[4] T. Kahveci and A. K. Singh, "An Efficient Index Structure for String Databases" in VLDB, 2001
[5] D. L. Nelson and M. M. Cox, "Lehinger Principles of BioChemistry", 3rd Edition, WORTH Publishers, 2000
[6] C. A. Orengo, A. E. Todd, and J. M. Thornton. From Protein Structure To Function. In Current Opinion in Structural Biology, 9:374-382, 1999
[7] S.Park, D.Lee, and W. W. Chu. "Fast Retrieval of Similar Subsequences in Long Sequence Databases." In KDEX, 1999
[8] H. Wang, C. Perng, W. Fan, S. Park, and P. S. Yu, "Indexing Weighted Sequences in Large Databases" in ICDE, 2003
[9] C. H. Wu,L-S. L. Yeh, H. Husang, L. Arminski, J. Castro Alvear, Y. Chen, Z-Z. Hu, Robert S.L., P. Kourtesis, B. E. Suzek, C. R. Vinayaka, J. Zhang, and Winona C. Barker. "The Protein Information Resource", in Nucleic Acids Rearch, 31:345-347, 2003