정렬된 잔기 사이의 최대거리와 유사도 그래프에 기반한 단백질 구조 정렬

(Protein Structure Alignment Based on Maximum of Residue Pair Distance and Similarity Graph)

김 우 철 † 박 상 현 †† 원 정 임 †††
(Woo-Cheol Kim) (Sang-Hyun Park) (Jung-Im Won)

요 약 최근 인간 게놈 프로젝트를 통해서 인간의 DNA가 해석된 이후 유전자가 생성하는 단백질의 기능에 대한 관심이 높아지고 있다. 단백질의 기능은 서열의 유사도보다는 진화과정 상에서 잘 보존되는 구조의 유사도에 더 연관되어 있다. 이를 통해 두 개의 단백질 간에 구조 유사성이 관찰되면 이로부터 이들이 유사한 생물학적 기능을 가질 것을 기대할 수 있다.

따라서 유사한 단백질 구조를 가진 단백질을 찾기 위한 방법으로 단백질 구조 정렬에 대한 많은 연구들이 진행되었다. 하지만 기존의 연구들은 유사도로 주로 RMSD(Root Mean Square Deviation)를 사용했기 때문에 두 단백질의 정렬 결과가 유사한지 혹은 유사하지 않은지를 직관적으로 판단하기 쉽지 않다. 또한 대부분의 기존 연구들은 정렬 결과로 최적의 정렬 결과 하나만을 찾기 때문에 서로 다른 목적을 가지는 사용자들을 만족시키기 어렵다. 따라서 본 논문에서는 새로운 유사도인 MRPD(Maximum of Residue Pair Distance)와 다수의 정렬 결과를 하나의 그래프로 표현하는 SG(Similarity Graph)을 기반으로 여러 가지 정렬 결과를 한 번에 생성하는 단백질 구조 정렬 방식을 제안한다. 단백질 정렬에 MRPD를 유사도로 사용하면 RMSD를 사용하는 경우에 비해서 유사 정도를 직관적으로 이해할 수 있을 뿐 아니라 신속하게 결과를 얻을 수 있다. SG는 사용자가 다양한 후보 정렬 결과들 중에서 자신이 원하는 정렬 결과를 신속히 검색할 수 있도록 지원한다. 따라서 본 논문에서 제안한 단백질 구조 정렬 알고리즘은 다양한 길이에 따른 다수의 최적 정렬들을 제시하여 사용자의 만족도를 향상시킬 수 있었으며, 다수의 정렬 결과 검색임에도 불구하고 정렬 시간은 기존 방법들과 거의 비슷하다는 장점이 있다.

키워드 : 단백질 구조, 구조 정렬, 구조 유사도

Abstract After the Human Genome Project finished the sequencing of a human DNA sequence, the concerns on protein functions are increasing. Since the structures of proteins are conserved in divergent evolution, their functions are determined by their structures rather than by their amino acid sequences. Therefore, if similarities between two protein structures are observed, we could expect them to have common biological functions. So far, a lot of researches on protein structure alignment have been performed. However, most of them use RMSD(Root Mean Square Deviation) as a similarity measure with which it is hard to judge the similarity level of two protein structures intuitively. In addition, they retrieve only one result having the highest alignment score with which it is hard to satisfy various users of different purpose. To overcome these limitations, we propose a novel protein structure alignment algorithm based on MRPD(Maximum of Residue Pair Distance) and SG (Similarity Graph). MRPD is more intuitive similarity measure by which fast filtering of unpromising pairs of protein pairs is possible, and SG is a compact representation method for multiple alignment results with which users can choose the most plausible one among various users' needs by providing multiple alignment results without compromising the time to align protein structures.

Key words : Protein structure, Structure alignment, Structure similarity measure

· 본 논문은 2005년 정부재원(교육인적자원부 학술연구조성사업비)으로 한국학술진흥재단(KRF-2005-206-D00015)과 2006년도 정부(과학기술부)의 재원으로 한국과학재단(No. R01-2006-000-11106-0)의 지원을 받았습니다.
† 학생회원 : 연세대학교 컴퓨터과학과
twelvepp@cs.yonsei.ac.kr
†† 종신회원 : 연세대학교 컴퓨터과학과 교수
sanghyun@cs.yonsei.ac.kr
(Corresponding author임)
††† 정 회 원 : 한양대학교 정보통신대학 컴퓨터전공 교수
jiwon@hanyang.ac.kr
논문접수 : 2006년 11월 14일
심사완료 : 2007년 7월 4일

1. 서 론

단백질의 구조 예측, 기능 분석, 유전자와 단백질의 상호 관계 발견 등의 생물학적 문제 해결은 생명체의 생명 현상을 규명하는 중요한 열쇠이다. 미국 국립보건원(NIH)을 주축으로 1988년부터 진행되고 있는 인간 게놈 프로젝트(Human Genome Project)[1]는 2003년 인간의 유전자 서열 분석을 완료하였으며, 유전자 지도가 공개되면서 이를 이용하여 유전자 혹은 유전자의 산물인 단백질의 구조와 기능, 그리고 상호 작용을 밝히려는 연구, 특히 질환 유전자(또는 단백질)의 위치를 검색하거나 기능을 분석하려는 연구가 매우 중요한 분야로 자리잡고 있다. 지난 수십 년 동안 생물학자들을 중심으로 단백질의 기능 분석에 관한 많은 연구들이 진행되어 왔다. 이는 대부분 생물학자들의 실험을 통한 분석 방식으로 많은 시간과 노력을 필요로 한다는 한계가 있었으나, 최근에는 컴퓨터공학과 정보기술을 도입하여 다수의 단백질 기능을 동시에 빠른 속도로 검색, 분석하려는 연구들이 보고되고 있다

두 개의 단백질 간에 매우 높은 서열 혹은 구조 유사성이 관찰되면 이들이 공통적인 진화 단계를 가진다고 예측할 수 있으며, 이로부터 이들이 유사한 생물학적 기능을 가질 것을 기대할 수 있다. 그러나 공통 조상을 가졌음에도 불구하고 서열은 진화 과정 중에 돌연변이로 인하여 치환이나, 삽입, 삭제 등의 변화가 생길 수 있으므로 서열 정보에만 의존한 단백질의 기능 예측은 오류가 있을 수 있다. 한편, 단백질의 구조는 서열에 비해 훨씬 더 보존성이 높다[2]. 이러한 특징을 이용하여 역 단백질 폴딩(inverse-protein-folding) 방법[3]에서는 기존의 단백질과 기능은 유사하지만 서열 간의 유사도는 낮고, 3차원의 입체 구조는 비슷한 단백질을 인공적으로 생성할 수 있음을 보였다. 따라서 새롭게 발견된 단백질의 기능을 예측하기 위해서는 기존의 기능이 밝혀진 단백질들 중에서 구조적으로 유사한 단백질을 찾아내는 단백질 구조 검색 방식을 사용해야 한다.

단백질의 기능은 단백질의 입체 구조에서의 아미노산 곁사슬(Side chain)에 의하여 결정된다. 단백질을 구성하는 각 아미노산의 원자들 중에서 C_α원자만 연결한 것을 골격(Backbone)이라 하고, C_α원자를 잔기(Residue)라고 한다. 골격은 단백질 구조로부터 얻을 수 있는 가장 기본이 되는 정보이다. 이러한 단백질 구조 정보를 이용하여 두 개의 단백질 구조간의 유사도를 최대로 하는 정렬을 찾아내고, 그 정렬에 참여하는 잔기들의 중첩(Superposition)과 변환(Transformation)을 구하는 것을 구조 정렬 문제(Structure alignment problem)라 한다. 여기에서 중첩은 정렬되는 잔기의 짝을 의미한다. 반면, 구조 중첩 문제(Structure superposition problem)는 중첩을 이미 알고 있는 상태에서 단지 정렬에 사용되는 변환만을 찾는 것으로서 더 단순한 문제라고 할 수 있다[4]. 구조 중첩 문제의 경우 잔기 개수의 선형적인 시간 안에 RMSD(Root Mean Square Deviation)를 최소화하는 회전(Rotation)과 이동(Translation) 변환을 찾을 수 있음이 알려져 있다[5]. 하지만 구조 정렬 문제는 NP-hard한 문제이므로 단백질의 구조를 정렬하는 거의 모든 방법들은 휴리스틱 접근 방식을 사용한다[4,6].

대부분의 단백질 구조 정렬 방법은 다음과 같은 4단계로 구성되어 있다[4]. 1) 정렬하려는 두 개의 단백질 구조를 공간상의 좌표로 표현한다. 2) 두 개의 단백질 구조를 정렬한다. 3) 정렬 결과를 최적화한다. 4) 정렬된 결과가 얼마나 통계적으로 유의한 결과인지 Z-score 등을 사용하여 평가한다. 이러한 단계를 통해 통계적으로나 분석적으로 의미 있는 정렬 결과를 얻을 수 있다. 그러나 때로 생물학적 의미를 가지는 정렬 결과를 놓치는 경우가 발생할 수 있다. 두 개의 단백질 구조를 정렬할 때 수백 만 개의 다른 정렬이 가능하므로, 이들 후보 정렬 결과 중에서 가장 높은 유사도를 갖는 하나의 정렬 결과만을 찾아내는 방식은 생물학적으로 의미 있는 정렬 결과를 놓칠 가능성이 더욱 더 크다. 따라서, 몇 개의 후보 정렬 결과를 제시하고 그 중에서 두 단백질간의 생물학적 유사성을 가장 잘 대표할 수 있는 정렬을 생물학자가 직접 선택할 수 있게 하는 정렬 알고리즘이 필요한 실정이다.

또한 대부분의 단백질 구조 정렬 방법에서는 유사도 지표(Similarity measure)로 정렬된 잔기와 잔기 사이의 평균 거리인 RMSD를 사용한다[7,8]. 그러나 RMSD로는 직관적인 유사성을 판단하기 어렵다는 단점이 있다. 예를 들어 두 개의 단백질 구조에 대하여 정렬된 잔기 사이의 거리가 각각 (0.5, 0.5, 4, 0.5, 0.5)와 (1.73, 2, 1.73, 2, 1.73)인 경우 RMSD는 모두 1.843로 같다. 그러나 시각적으로 두 개의 정렬 결과가 동일한 유사도를 나타낸다고 보기는 어렵다. 즉, 단순히 RMSD만으로는 두 단백질의 정렬 결과가 유사한지 혹은 유사하지 않은지를 직관적으로 판단하기는 쉽지 않다.

본 논문은 이러한 기존 방식의 문제점을 해결하기 위해 1) 하나의 정렬 결과만을 제시하는 기존 방법과는 달리 다양한 여러 정렬 결과를 내포하는 새로운 유사도 표현 방식인 SG(Similarity Graph)를 제안한다. 제안하는 SG는 사용자가 다양한 후보 정렬 결과들 중에서 자신이 원하는 정렬 결과를 신속히 검색할 수 있도록 지원한다. 2) RMSD에 비해 보다 직관적인 유사도 지표인 MRPD(Maximum of Residue Pair Distance)를 제안한다. 또한 정렬되지 않는 두 단백질 구조가 정렬 후에 특정 임계값을 만족하는 MRPD를 갖는지를 신속하게 판별할

수 있도록 하여 RMSD를 사용하는 경우에 비하여 매우 빠른 정렬을 가능하도록 한다. 3) SG를 구성하는데 필요한 여러 정렬 결과들을 빠르게 찾아내는 새로운 단백질 구조 정렬 알고리즘을 제안한다. 제안한 알고리즘은 다양한 길이에 따른 다수의 최적 정렬 들을 찾아냄으로써 사용자의 만족도를 높일 수 있을 뿐 아니라 전체 정렬 시간이 단지 하나의 정렬만을 찾는 기존 알고리즘의 정렬 시간과 거의 비슷하다는 장점을 갖는다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 단백질 구조 정렬 알고리즘에 대해서 설명하고, 3장에서는 본 논문에서 사용하는 용어들을 정의한다. 4장에서는 새로운 단백질 구조간의 유사도 표현 방식인 SG를 제안하고, 5장에서는 유사도 지표로 사용하는 MRPD를 설명한다. 6장에서는 본 논문에서 제안하는 단백질 구조 정렬 알고리즘을 설명하고, 7장의 실험을 통해서 제안하는 알고리즘의 성능을 검증한다. 마지막으로 8장에서 본 논문을 요약하고, 결론을 제시한다.

2. 관련 연구

단백질 구조를 정렬하는 문제는 NP-hard 하다는 것이 알려져 있다. 따라서 대부분의 단백질 구조 정렬 알고리즘에서는 전체 구조를 한번에 정렬하는 대신 구조 중첩 문제로 단순화하여 해결하는 방식을 채택한다. 즉, 연산량이 작은 부분 구조들을 먼저 정렬한 후 그것들을 중첩으로 이용해 정렬에 사용된 변환을 찾는 방식을 많이 이용한다.

단백질 구조 정렬 알고리즘은 사용하는 데이타에 따라서 3가지로 구분할 수 있다[9]. 첫 번째는 골격을 구성하는 C_α원자를 이용하는 방법이며, 두 번째는 단백질 2차 구조SSE(Secondary Structure Element)를 이용하는 방법이다. 마지막은 기하학적 해싱(Geometric hashing)을 이용하는 방법이다.

C_α원자를 이용하는 가장 간단한 방법은 다이나믹 프로그래밍 기법을 사용하는 것이다. 가장 대표적인 방식으로는 DALI[2]와 CE[10]가 있다. 다이나믹 프로그래밍을 여러 번 수행하여 정렬 결과를 얻는 방식으로는 Double Dynamic Programming[11], Iterative Dynamic Programming[12], MINRMS[7] 등이 있다.

DALI는 거리 행렬(Distance matrix)를 이용해 단백질 구조를 정렬한다. 거리 행렬은 단백질 구조의 내부 잔기 거리(Intra-residue distance)를 표현하는 n*n의 행렬이다. 즉 거리 행렬의 i번째 행의 j번째 열에는 해당 단백질 구조의 i번째 잔기와 j번째 잔기 사이의 거리를 기록한다. 거리 행렬의 중요한 특징은 단백질 간의 구조가 유사한 경우 거리 행렬 또한 유사하다는 것이다. DALI는 이러한 거리 행렬의 특징을 단백질 구조 정렬에 이용한다. 단, 정렬 시간을 줄이기 위해서 크기가 큰 전체 거리 행렬을 한번에 비교하지 않고 여러 개의 작은 부분 거리 행렬로 나누어 비교함으로써 유사한 부분 거리 행렬의 쌍들을 찾아낸다. 다음, 찾아낸 유사 부분 거리 행렬의 쌍들을 연결해감으로써 점점 더 큰 거리 행렬을 만들어간다.

CE는 거리 행렬 대신 단백질 구조를 일정 길이로 나눈 부분 구조(Fragment)을 정렬에 이용한다. DALI의 경우는 부분 거리 행렬을 비교하여 유사한 부분 구조를 찾지만 CE는 부분 구조 사이의 유사도를 측정해서 유사한 부분 구조의 짝인 AFP(Aligned Fragment Pair)를 찾는다. 그런 다음 DALI와 마찬가지로 AFP를 확장해서 단백질 구조를 정렬한다.

DALI와 CE같이 다이나믹 프로그래밍 방식을 이용하는 경우, 정렬 시간은 데이타의 길이에 많은 영향을 받는다. 따라서 DALI와 CE 모두 단백질 구조의 유사 정도에 상관없이 일정 개수의 C_α를 하나의 데이타로 변환해 연산량을 줄이는 휴리스틱을 적용한다. 그러나 유사도가 높은 두 단백질의 구조를 비교하는 경우에도 유사한 부분을 강제로 여러 개의 부분 구조로 나누어 비교하기 때문에 연산 시간이 많이 걸리는 단점이 있다.

최근에 Itay Lotan 과 Fabian Schwarzer는 [13]에서 C_α원자를 일정한 개수로 줄여서 정열 시간을 최적화하는 알고리즘을 제안하였다. C_α원자를 이용하는 다른 알고리즘과는 달리, 이 알고리즘은 단백질 구조의 길이와 상관없이 정렬에 참여하는 C_α원자를 일정한 개수로 줄임으로써 빠른 정렬을 수행할 수 있는 장점이 있다. 그러나 정렬이 완료된 후 정렬된 구조의 유사도만 확인할 수 있을 뿐 실제로 어떤 잔기가 정렬에 참여했는지는 찾을 수 없다는 단점이 있다.

SSE를 이용하는 대표적인 방법으로는 VAST[14]가 있다. VAST는 단백질의 2차 구조인 SSE를 이용하는 계층적 정렬 알고리즘이다. 이 알고리즘은 먼저 C_α원자보다 상위 구조인 SSE를 정렬해 정렬 결과를 얻은 후 그 결과를 중첩으로 사용해 C_α원자 단계로 정렬을 확장해 나간다. C_α원자 대신 SSE를 이용하여 정렬을 수행하므로 중첩을 좀 더 빨리 찾을 수 있으며 이로 인해 단백질 구조 정렬에 소요되는 전체 시간을 줄일 수 있다는 장점이 있다. 그러나 아직 많은 단백질들은 SSE 정보가 밝혀지지 않았으므로 이 알고리즘을 모든 단백질에 적용할 수 없다는 단점이 있다. 참고로 C_α원자에 대한 정보는 NMR(Nuclear Magnetic Resonance)이나 X-ray crystallography를 통해서 얻을 수 있으며, C_α원자로부터 상위 정보인 SSE를 찾기 위한 연구로는 DSSP[15], STRIDE[16] 등이 있다.

마지막으로 기하학적 해싱 방법에서는 먼저 기준틀

(Reference frame)을 해쉬값으로 변환하여 해쉬 테이블에 저장한 후, 기준틀 자체보다는 기준틀의 해쉬값을 이용하여 정렬을 수행한다. 이 방식을 이용하는 대표적인 단백질 구조 정렬 알고리즘인 3D-lookup[17]은 SSE를 기준틀로 이용한다. 또한, Nussinov와 Wolfson은 [18]에서 C_α원자를 기준틀로 사용하는 단백질 구조 정렬 알고리즘을 제안하였다. 이 방법은 미리 결정된 기준틀만을 정렬에 이용하므로 정렬 결과의 정확도에 한계가 있다.

또한 최근에 Michael[19]은 매듭 이론(Knot theory)과 기하학적 회선(Geometric convolution)을 통하여 각 단백질 구조의 꼬임과 교차를 찾아 단백질 구조를 정렬하는 방법을 제시하였다. 그러나 이 방식은 전체적인 구조의 유사성이 떨어지는 경우에도 유사한 꼬임이나 교차를 가지고 있다면 이를 유사한 구조로 판단하는 문제점이 있다.

3. 용어 정의

본 장에서는 본 논문에서의 논의 전개에 필요한 용어 및 기호를 정의한다.

정의 1. 단백질 구조 S

$$S = < R_1, R_2, \cdots, R_r, \cdots, R_n >,$$
$$R_r = (x_r, y_r, z_r), |S| = n$$

단백질 구조 S는 3차원의 공간 좌표로 표현된 잔기 R_r ($1 \le r \le n$)들의 리스트이며, S내에 포함된 잔기의 개수 n을 |S|로 표현하고, S의 길이라 정의한다. □

정의 2. 단백질 부분 구조 SS

$$SS_i = < R_{i(1)}, R_{i(2)}, \cdots, R_{i(r)}, \cdots, R_{i(k)} >,$$
$$1 \le i(1) < i(2) < \cdots < i(k) \le n$$

단백질 구조 S의 부분 구조 SS는 S에서 추출한 k개의 잔기들을 연결해서 만든 새로운 구조로, S로부터 추출 가능한 SS의 개수는 S를 구성하는 잔기 집합 {R_1, R_2, …, R_n}의 부분 집합의 개수에서 공집합의 개수를 뺀 것과 같으므로 2^n-1이다. □

정의 3. 변환(Transformation) T

단백질 구조를 정렬하기 위해서는 대상이 되는 두 개의 단백질 구조 중에서 임의의 단백질구조의 위치를 고정하고 다른 나머지 구조를 3차원 공간상에서 회전(Rotation) 변환과 이동(Translation) 변환을 여러 번 수행해야 한다. 이러한 회전 변환과 이동 변환을 합해서 변환 T라고 정의한다. □

정의 4. 변환된 단백질 구조 TS

$$TS = < TR_1, TR_2, \cdots, TR_r, \cdots, TR_n >$$

단백질 구조 S를 변환 T를 이용하여 변환한 것을 TS라고 정의한다. 예를 들어, 임의의 단백질 구조 S의 부분 구조 중 하나인 SS_3를 변환 T_4를 사용하여 변환한 것은 T_4SS_3로 표현한다. □

정의 5. 단백질 구조의 RMSD(Root Mean Square Deviation)

$$RMSD(S^A, S^B) = \sqrt{\frac{\sum_{i=1}^{n} \left| R_i^A - R_i^B \right|}{n}}, \ |S^A| = |S^B| = n$$

두 개의 단백질 구조 S^A와 S^B의 길이가 n으로 같을 때 두 단백질 구조간의 RMSD는 정렬된 잔기 사이의 평균 거리로 정의한다. □

정의 6. 단백질 구조의 유사도 SScore

$$SScore(S^A, S^B)$$

두 개의 정렬된 단백질 구조 S^A와 S^B의 유사도를 SScore로 정의한다. 일반적으로 많이 사용하는 유사도 지표로는 RMSD와 Z-score, p-value등이 있다. 유사도의 최대값은 $SScore_{MAX}$, 최소값은 $SScore_{MIN}$으로 표현한다. 예를 들어, 유사도 지표로 RMSD를 사용하는 경우 $SScore_{MAX}$는 0Å이고 $SScore_{MIN}$는 ∞Å이다. □

정의 7. 단백질 구조 정렬

단백질 구조 정렬은 단백질 구조의 일정 잔기 부위에 대응하는 다른 단백질 구조의 잔기를 찾는 것이다. 이때 정렬 방법에 따라 다양한 정렬 결과가 나온다. 그러나 일반적으로 정렬은 비교하려는 두 단백질 구조 중에서 유사도가 높은 부분 구조의 잔기 쌍을 찾는 것이다. 따라서 임의의 두 개의 단백질 구조 S^A와 S^B를 정렬시키기 위하여는 모든 부분 구조의 쌍들을 비교하여 유사도가 높은 부분 구조의 쌍을 찾아야 한다. 즉, 단백질 구조 정렬은 모든 i, j, k에 대해서 가장 큰 SScore ($T_iSS_j^A$, SS_k^B)를 보이는 변환 T_i와 부분 구조 SS_j^A, SS_k^B를 찾는 것으로 정의된다. □

4. 유사도 그래프(Similarity Graph)

본 장에서는 여러 개의 정렬 결과를 함축적으로 표현할 수 있는 새로운 유사도 표현 방식인 SG(Similarity Graph)를 제안한다. 4.1절에서는 하나의 정렬 결과만을 제시하는 기존 방식의 문제점을 지적하고, 4.2절에서는 제안하는 SG의 기본 개념을 설명한 후 SG가 기존의 문제점을 어떻게 해결할 수 있는지를 설명한다. 마지막으로 4.3절에서는 다수의 SG를 보다 쉽게 비교, 분석할 수 있도록 SG를 하나의 숫자 값으로 표현하는 SG-Score에 대해 기술한다.

4.1 하나의 정렬 결과만을 제시하는 기존 방식의 문제점

기존의 단백질 구조 정렬 방법들은 대부분 하나의 정렬 결과만을 보여준다. 예를 들어 그림 1(a)의 두 단백질 구조 S^A와 S^B를 정렬시킨 경우 기존의 방법들은 최종 결과로서 그림 1(b)나 그림 1(c) 중 하나만을 사용자에게 제시한다. 따라서 최종 결과로서 그림 1(b)만을 사

(a) 정렬하려는 단백질 구조 S^A와 S^B

(b) 정렬된 결과 1 (n: 100, RMSD: 0.2Å)

(c) 정렬된 결과 2 (n: 160, RMSD: 1.0Å)

그림 1 단백질 구조 정렬의 예

용자에게 제시할 경우 사용자는 그림 1(c)의 정렬 결과를 확인할 수 없게 된다. 만약 사용자가 얻고자 했던 정렬 결과가 그림 1(c)였다면 이로 인해 잘못된 판단을 내릴 수도 있다는 위험이 존재한다.

이러한 문제점을 해결하기 위해서는 하나의 정렬 결과 보다는 여러 개의 후보 정렬 결과를 제시하고 그 중에서 가장 의미 있는 결과를 생물학자가 직접 선택할 수 있게 하는 정렬 방법이 필요하다. 이 때 여러 개의 정렬 결과들을 리스트 형태로 나열하는 방법 대신에 이들을 하나의 그래프로 함축하여 표현할 수 있다면 원하는 정렬 결과를 보다 쉽게 선택할 수 있을 것이다.

4.2 SG(Similarity Graph)

하나의 정렬 결과만을 제시하는 기존 방식의 문제점을 해결하기 위해서 본 논문은 다양한 정렬 결과를 함축적으로 표현할 수 있는 새로운 유사도 표현 방식인 SG를 제안한다. 먼저, SG의 기본 개념을 살펴보자.

두 개의 단백질 구조를 정렬할 때 그림 1(b)와 그림 1(c)처럼 정렬된 길이가 서로 다른 경우에는 어떤 정렬 결과가 더 좋은 것인지 판단하기 힘들다. 그러나 정렬된 길이가 같다면 유사도가 높은 것이 더 좋은 정렬 결과라고 할 수 있다. 따라서, 여러 개의 정렬 결과를 유사도와 정렬된 길이를 함께 표시하는 것이 바람직하다. 본 연구에서는 정렬 결과의 가독성을 높이기 위하여 (정렬 길이, 유사도)의 쌍으로 이루어진 2차원의 정렬 결과를 단순히 리스트 형태로 나열하기 보다는 2차원의 공간 좌표에 점으로 표시하는 방법을 사용한다. 그림 2에 (n, SScore)으로 이루어진 정렬 결과를 2차원 공간 상의 (x, y) 점으로 표현한 예를 보인다. 이 때, n의 최대 크기는 정렬하려는 두 단백질 구조 중에서 길이가 작은 단백질 구조의 길이와 같다. 따라서 x는 0부터 min

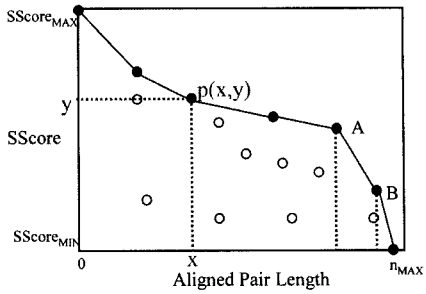


그림 2 SG의 예

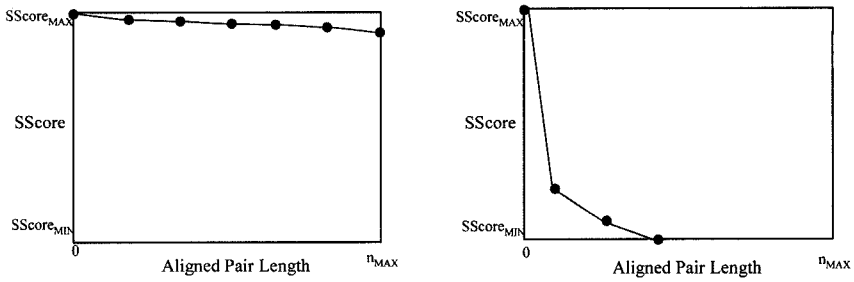
($|S^A|$, $|S^B|$)의 값을 가지며, y는 유사도 점수의 최소값부터 최대값의 범위를 갖는다.

이 때 그림 2의 흰 점들과 같이 다른 점들에 비해 유사도가 낮으면서 길이도 짧은 결과들은 최종 정렬 결과에 포함시키지 않는다. 예를 들어 두 개의 정렬 결과 p_{17}(100, 0.2Å)과 p_5(89, 0.5Å)가 있을 때 p_5는 p_{17}에 비해 정렬된 길이가 짧고(89<100), 유사도도 낮으므로 (RMSD의 경우 클수록 유사도는 낮기 때문에 유사도(0.5Å)<유사도(0.2Å)) 최종 정렬 결과에 포함시키지 않는다. 다음, 불필요한 정렬 결과를 제거한 후 남은 최종 정렬 결과에 해당하는 점들을 선분으로 연결한다. 이렇게 생성된 그래프를 두 단백질 구조의 SG라고 정의한다.

이와 같이 여러 개의 정렬 결과를 SG를 이용해 하나의 그래프에 함축하여 표현함으로써 다음과 같은 두 가지 장점을 얻을 수 있다. 첫째, 그래프의 모양으로 두 단백질 구조간의 전체적인 유사성을 쉽게 판별할 수 있다. 예를 들어 그림 3(a)와 같이 그래프가 높은 유사도 값을 가지면서 X축과 평행한 경우는 두 단백질이 전체적으로 유사한 구조를 가짐을 나타낸다. 반대로 그림 3(b)와 같이 길이가 증가함에 따라 유사도가 급격하게 떨어지는 그래프는 두 단백질 구조가 거의 유사하지 않음을 나타낸다. 둘째, 유사하다고 판단할 수 있는 부분 구조의 최대 길이를 쉽게 판별할 수 있다. 예를 들어, 그림 2의 SG를 다시 살펴보면 점 A로부터 점 B로 이동하면서 유사도가 급격히 감소함을 알 수 있다. 이를 통해 점 A에 해당하는 정렬 길이까지는 두 단백질이 유사한 부분 구조를 가지지만, 그 길이 이상이 되면 두 단백질이 유사한 부분 구조를 가지지 않음을 알 수 있다. 이러한 결과는 생물학자가 최적의 정렬 결과를 판단하는데 도움을 줄 수 있다.

4.3 SG-Score

SG를 이용하면 두 개의 단백질 구조간의 다양한 정렬 결과를 한 눈에 확인할 수 있다. 그러나, 단백질 데이타베이스를 검색하여 질의 단백질과 유사한 구조를



(a) 유사한 구조를 정렬한 결과 (b) 유사하지 않은 구조를 정렬한 결과

그림 3 여러 가지 형태의 SG

가지는 단백질을 찾아내는 응용을 고려한다면 다수의 SG를 손쉽게 비교할 수 있도록 SG를 하나의 숫자로 표현하는 것이 바람직하다. 본 절에서는 SG가 차지하는 면적을 이용하여 SG를 하나의 숫자로 표현하는 SG-Score에 대해 기술한다.

SG에서 그래프의 아래쪽은 정렬을 통해서 얻을 수 있는 결과를 의미하고 위쪽은 정렬을 통해서도 얻을 수 없는 결과를 나타낸다. 따라서 그림 3(a)와 같이 SG에서 그래프 아래쪽 영역이 넓을수록 비교 대상이 되는 두 단백질 구조가 유사하다고 해석할 수 있다. 이러한 해석을 바탕으로 SG-Score를 (그래프의 아래쪽 면적) / (그래프의 전체 면적)로 정의할 수 있다. 이 때, SG-Score의 최소값과 최대값은 각각 0과 1이 된다. 예를 들어 그림 3(a)는 1에 가까운 SG-Score를 가지며 그림 3(b)는 0에 가까운 SG-Score를 갖는다.

그러나 SG-Score를 위와 같이 정의한다면, n이 커질수록 두 단백질 구조가 유사할 확률이 낮아져서 더 작은 SG-Score를 갖게 되는 문제점이 생기게 된다. 예를 들어, 길이가 각각 100, 2, 300인 세 개의 단백질 구조 S^A, S^B, S^C에 대하여, S^A와 S^B의 정렬 결과를 표현하는 SG로부터 SG-Score=1을 얻었고, S^A와 S^C의 정렬 결과를 표현하는 SG로부터 SG-Score =0.95를 얻었다고 가정하자. 이 경우, S^A와 S^C의 정렬 결과가 S^A와 S^B의 정렬 결과보다 확률적으로 더 의미 있는 것임에도 불구하고, 더 작은 SG-Score를 갖게 되는 문제점이 발생한다.

이러한 문제점은 두 단백질 구조에서 우연히 일어나는 구조의 유사성을 제거함으로써 해결할 수 있다. 두 단백질 구조가 우연에 의해 유사해질 수 있는 가능성은 정렬하려는 단백질 구조와 같은 길이를 갖는 임의의 두 단백질 구조를 정렬한 결과로부터 얻을 수 있다. 즉, 정렬하려는 단백질 구조와 같은 길이를 갖는 두 개의 단백질을 임의로 생성하여 이들의 정렬 결과를 동일한 평면 상에 그래프로 나타낸 후, 이 그래프부터 얼마나 유사도가 향상되었는가를 기준으로 SG-score를 결정하면 된다.

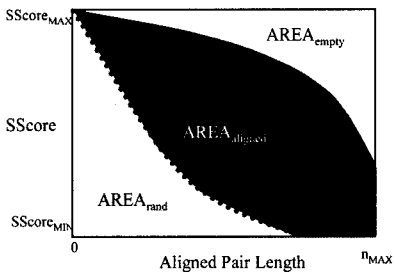


그림 4 SG-Score 계산을 위한 영역의 분할

정의 8. SG-score

단백질 구조 S^A와 S^B를 정렬한 결과는 실선으로, S^A, S^B와 동일한 길이를 갖는 임의로 생성된 두 단백질 구조를 정렬한 결과는 점선으로 동일 SG 평면상에 표시한다. 이 때 좌하단에서 점선까지의 영역을 $AREA_{rand}$, 점선에서 실선까지의 영역을 $AREA_{aligned}$, 실선에서 우상단까지의 영역을 $AREA_{empty}$라고 하자. SG-score는 다음과 같이 정의된다.

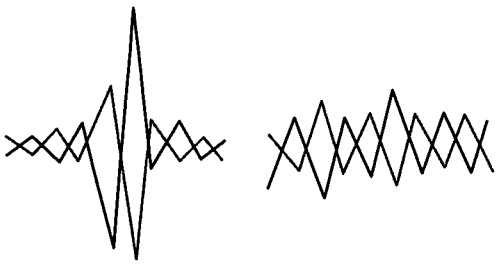
$$SG-score = \frac{\text{실제 정렬을 통해서 늘어난 영역}}{\text{정렬을 통해서 늘어날 수 있는 최대 영역}} = \frac{AREA_{aligned}}{AREA_{aligned} + AREA_{empty}}$$

□

5. MRPD(Maximum of Residue Pair Distance)

기존의 단백질 구조 정렬 알고리즘은 유사도를 나타내기 위해서 주로 기하학적 거리인 RMSD를 사용한다. 그리고 Z-score와 p-value는 보조적인 수단으로 사용한다. RMSD는 각 정렬된 잔기 사이의 평균 거리를 의미하지만 이러한 평균값만으로는 '전체적인 구조의 유사도를 직관적으로 파악하기 힘들다는 단점이 있다.

3차원 구조를 공간 상의 노드(Node)와 에지(Edge)로 구성한 경우, "유사하다"라는 의미는 두 개의 구조가 정렬되었을 때 정렬된 노드 사이의 거리가 작다는 것을 의미한다. 이는 "노드 사이의 평균 거리가 작다"라는 의



(a) 정렬결과 (b) 정렬결과

그림 5 RMSD를 이용한 정렬의 단점

미보다는 "모든 노드 사이의 거리가 일정한 값을 넘지 않는다"라는 의미에 가깝다.

예를 들어 정렬된 노드 사이의 거리가 (0.1, 0.1, 0.2, 0.2, 0.8, 1.0, 0.2, 0.2, 0.1, 0.1)인 결과(그림 5.(a))와 (0.40, 0.40, 0.50, 0.42, 0.42, 0.50, 0.42, 0.42, 0.40, 0.40)인 결과(그림 5.(b))가 있다고 가정하자. 이 때 두 경우 모두 RMSD가 0.429이므로 두 개의 정렬 결과는 같은 유사도를 갖는 것으로 판단되지만 실제로는 후자의 경우가 시각적으로 더 유사해 보인다. 그 이유는 시각적으로 유사 여부를 판단할 때 정렬된 노드 사이의 거리 중 최대값인 1.0과 0.50의 영향을 많이 받기 때문이다. 이와 같이 RMSD는 평균값이므로 전자의 경우 0.8과 1.0의 거리가 다른 값들에 의해 묻혀져서 유사도 판단에 오류가 생길 수 있다. 이러한 특성은 n이 클수록 더욱 심하게 나타나는 경향이 있다. 그러므로, 두 단백질 구조의 전체적인 유사도를 직관적으로 판단하기 위해서는 RMSD보다 정렬된 잔기 사이의 거리 중에서의 최대값을 유사도 지표로 사용하는 것이 더욱 바람직하다. 이러한 분석을 바탕으로 본 논문은 새로운 유사도 지표인 MRPD를 다음과 같이 정의한다.

정의 9. MRPD

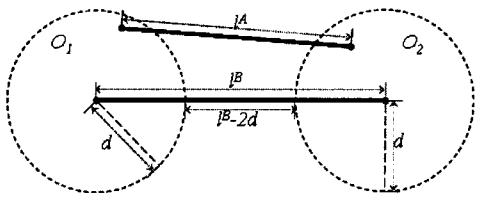
$$MRPD(S^A, S^B) = \max_{i=1}^{n} \left| R_i^A - R_i^B \right|, \quad \left| S^A \right| = \left| S^B \right| = n$$

길이가 n인 두 개의 단백질 구조 S^A와 S^B가 정렬되어 있을 때 MRPD는 정렬된 잔기 사이의 거리 중에서 가장 큰 값으로 정의된다. 만일, 두 단백질의 구조가 동일하면 그들의 MRPD는 0이 된다 (즉, MRPD(S^A, S^A) = 0). □

MRPD를 계산하기 위해서는 RMSD와 마찬가지로 두 단백질 구조가 정렬되어 있어야만 한다. 그러나, MRPD는 정리 1과 정리 2를 통해서 정렬되지 않은 상태에서도 정렬된 결과의 MRPD가 어떤 값보다 큰지 작은지 미리 판단할 수 있다. 따라서 유사도로 MRPD를 이용하면 실제로 정렬을 수행하지 않고도 정렬 결과의 MRPD가 일정한 값을 넘는 단백질 구조의 짝을 제외할 수 있기 때문에 RMSD를 사용하는 경우에 비해서 정렬 속도가 빠르다.

정리 1. 두 단백질 구조 S^A와 S^B가 각각 두 개의 잔기로 구성되어 있고, 잔기 사이의 거리가 각각 l^A, l^B 라고 하자. 만약 $|l^A - l^B| > 2d$ 라면 MRPD(T_iS^A, S^B) < d를 만족시키는 변환 T_i는 존재하지 않는다. □

증명 1.



먼저 정렬하려는 두 개의 단백질 구조 중 길이간 긴 구조를 S^B라 하고 짧은 구조를 S^A라 하자. 즉 $l^B > l^A$ 이다. 이 때 S^A와 S^B를 정렬한 후에 MRPD가 d보다 작기 위해서는 S^B의 양 끝 점(잔기)를 중심으로 반지름이 d인 구 O_1과 O_2를 그렸을 때 위의 그림과 같이 T_iS^A의 두 점(잔기)는 각각 O_1내부와 O_2내부에 위치해야 한다. 따라서 이러한 T_iS^A가 존재하기 위해서는 T_iS^A의 길이(=l^A)가 구 사이의 가장 가까운 거리인 l^B-2d보다 커야 한다. 그러나 $|l^A - l^B| > 2d$을 만족하는 경우에는 아래 식. 1과 같이 l^A는 항상 l^B-2d보다 작기 때문에 MRPD(T_iS^A, S^B) < d를 만족하는 T_i가 존재하지 않는다.

$$\begin{aligned} & |l^A - l^B| > 2d \\ \Leftrightarrow\ & l^B - l^A > 2d \\ \Leftrightarrow\ & l^B - 2d > l^A \end{aligned} \quad (1)$$

□

정리 2. S^A와 S^B가 n개의 잔기로 구성되어 있고 S^A의 각 잔기 R_r^A와 R_{r+1}^A사이의 거리를 l_r^A, 그리고 S^B의 각 잔기 R_r^B와 R_{r+1}^B사이의 거리를 l_r^B고 할 때 $|l_r^A - l_r^B| > 2d$을 만족하는 r이 1개 이상 존재하면 MRPD (T_iS^A, S^B) < d를 만족하는 변환 T_i는 존재하지 않는다. □

증명 2. MRPD(T_iS^A, S^B) < d 가 성립하기 위하여는 정렬된 모든 잔기의 짝 사이의 거리가 d를 넘지 않아야 한다. 그러나 정리 1에 의해서 $|l_r^A - l_r^B| > 2d$을 만족하는 r이 1개 이상 존재하면 적어도 하나 이상의 잔기의 짝 사이의 거리는 d 보다 크다. 따라서 정리 2를 만족한다. □

정리 1과 정리 2를 통해서 단백질 구조를 정렬하지 않은 상태에서도 MRPD가 임의의 특정값 보다 작은지를 쉽게 판별할 수 있다. 이를 이용해 C4MRPD를 다음과 같이 정의한다. 이 후 C4MRPD의 만족 여부에 따라서 단백질 구조 짝을 실제로 정렬할지 하지 않을지 필

터링 할 수 있다.

정의 10. C4MRPD

$C4MRPD(S^A, S^B, MRPD_{Threshold})$ = true (if $\exists T_i$, $MRPD(T_iS^A, S^B) < MRPD_{Threshold}$)
= false (otherwise)

C4MRPD가 true인 경우에는 S^A와 S^B의 모든 잔기가 정렬에 참여한 정렬 결과 중에서 MRPD가 $MRPD_{Threshold}$를 넘지 않는 정렬 결과가 있음을 의미한다. □

6. 세부 알고리즘

그림 6은 본 논문에서 제안하는 단백질 구조 정렬 알고리즘의 개념도이다. 1) 먼저 정렬하려는 단백질 QS와 TS의 구조 정보를 이용하여 유사도가 높고 연속된 잔기만으로 구성된 CFP(Continuous Fragment Pair)들을 찾는다. 2) 찾은 CFP들을 연결해서 CFP보다 유사도는 낮지만 연속되지 않은 잔기들도 포함하는 SPP(Super Position Pair)들을 찾는다. 3) 찾은 SPP들을 중첩으로 사용해서 정렬된 구조를 확장한 ASP(Aligned Sub-structure Pair)들을 생성하는 과정을 반복적으로 수행한다. 4) 마지막으로 모든 정렬 결과를 이용해 SG를 생성한다.

6.1 CFP 찾기

정렬의 뼈대가 되는 중첩을 찾기 위해서 일정 MRPD를 넘지 않는 모든 SS_i^{QS}와 SS_j^{TS}의 쌍들을 구해야 한다. 하지만 모든 SS_i^{QS}와 SS_j^{TS}의 개수가 각각 $(2^{|QS|}-1)$과 $(2^{|TS|}-1)$이기 때문에 정렬 가능한 잔기 쌍의 조합의 개수는 $(2^{|QS|}-1)\times(2^{|TS|}-1)$개이다. 따라서 모든 잔기 쌍에 대해서 MRPD를 구할 경우 정렬 시간이 오래 걸린다.

이러한 문제점을 해결하기 위해서 중첩을 2단계로 나누어 찾는다. 그 첫 번째 단계는 정렬된 잔기들 사이에 갭(Gap)이 존재하지 않는 중첩인 CFP를 찾는 과정이다. CFP를 찾기 위한 알고리즘을 알고리즘 1에 기술한다. 먼저 단계1에서는 연속된 잔기로만 구성된 부분 구조만을 이용해서 연속된 부분 구조의 쌍인 CFP을 찾는다. 즉 임의의 유사성 $MRPD_{CFP}$를 만족하면서 갭이 없는 부분 구조의 짝인 CFP을 찾는다(줄 2-6). 이렇게 찾아낸 CFP들은 길이가 2나 3과 같이 확률적으로 유사할 가능성이 매우 높은 잔기 쌍들도 포함하고 있다. 따라서 길이가 긴 순서로 CFP들을 정렬하여 상위 $CUTOFF_{CFP}$개의 CFP만을 중첩으로 사용한다(줄 7-8).

알고리즘 1. CFP찾기

```
1 CFPSET = Ø /* set of sub-structure pair */
2 for each(SS_i^QS that consists of continuous
  residues in S^QS)
3   for each(SS_j^TS that consists of continuous
    residues in S^TS)
4     if(C4MRPD(SS_i^QS, SS_j^TS, MRPD_CFP))
5       CFP = pair of (SS_i^QS, SS_j^TS)
6       insert CFP into CFPSET
7 sort CFPSET by length
8 CFPSET = select CUTOFF_CFP of CFP from
  CFPSET
```

6.2 SPP 찾기

앞 절에서 찾아낸 CFP는 연속된 잔기만으로 구성되어 있다. 따라서 이 단계에서는 연속하지 않은 잔기들도 포함하는 중첩인 SPP를 찾는다. 일반적으로 단백질 구조를 정렬한 결과는 $<R_6^{QS}, R_{10}^{QS}, R_{14}^{QS}, R_{16}^{QS}, R_{20}^{QS}, R2_{23}^{QS}>$, $<R_3^{TS}, R_5^{TS}, R_9^{TS}, R_{12}^{TS}, R_{14}^{TS}, R2_{18}^{TS}>$와 같이 이산적인 잔기들로 구성되기 보다는 $<R_4^{QS}, R_5^{QS}, R_6^{QS}, R_{16}^{QS}, R_{17}^{QS}, R2_{18}^{QS}>$, $<R_{13}^{TS}, R_{14}^{TS}, R_{15}^{TS}, R_{18}^{TS}, R_{19}^{TS}, R2_{20}^{TS}>$와 같이 부분적으로 연속된 잔기들로 구성된다. 따라서 본 논문은 앞 절에서 찾아낸 CFP중에서 유사한 변환 T를 사용하여 서로 정렬 가능한 CFP들을 연결해 SPP를 생성한다.

SPP를 찾기 위해서는 모든 CFP 조합간의 정렬 가능 여부를 C4MRPD를 통해 확인해야 한다. 하지만 CFP개수가 m인 경우 CFP의 조합의 개수는 CFP를 2개 조합한 개수 $_mC_2$, 3개 조합한 개수 $_mC_3$, …, m개 조합한 개수 $_mC_m$을 모두 더한 것이 되므로 총 (2^m-1-m)이다. 따라서 모든 조합을 검사하려면 매우 많은 시간이 걸린다.

이 문제점을 해결하기 위해 본 논문은 모든 개수의 조합에 대해 검사하지 않고 CFP를 2개씩 조합한 결과만을 검사한 후 이 결과를 이용해 조합의 개수를 확장시켜 나가는 방법을 사용한다. 확장시키는 원리는 그림 7과 같이 {①, ②, ③}의 CFP가 조합 가능하기 위해서는 {①, ②}, {②, ③}, {③, ①}가 조합 가능해야 한다는 것이다. 이러한 원리를 통해서 CFP를 2개씩 조합한 결

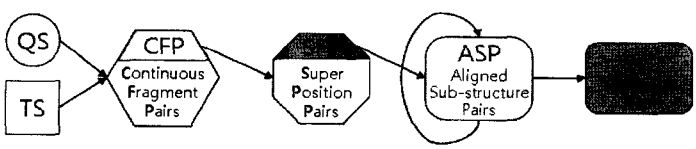


그림 6 제안하는 단백질 구조 정렬 알고리즘의 개념도

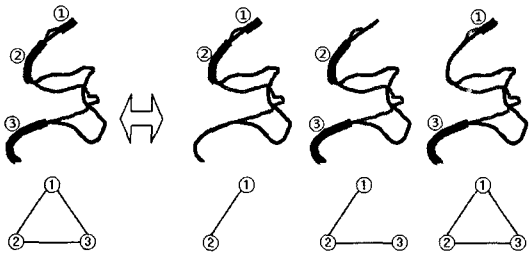


그림 7 CFP를 확장하여 SPP를 만드는 과정

과만을 이용해 여러 개의 CFP로 구성된 SPP를 찾을 수 있다. 이 때 SPP는 그래프 이론 중에서 최대 클리크(*Maximal clique*)를 찾는 방법들[20, 21, 22, 23]을 통해서 찾을 수 있다. 여기서 클리크는 그래프의 모든 노드가 서로 연결된 부분 그래프로 정의되며 최대 클리크는 어떠한 다른 클리크에도 속하지 않는 클리크로 정의된다. 예를 들어 그림 7에서 {①, ②}, {②, ③}, {③, ①}, {①, ②, ③}은 각각 클리크이지만 오직 {①, ②, ③}만이 최대 클리크이다.

CFP를 2개씩 선택해서 C4MRPD를 검사하기 위한 알고리즘을 알고리즘 2에 기술한다. 먼저 모든 CFP중에서 서로 다른 두 개의 CFP_i와 CFP_j를 선택한다(줄 3-4). 그런 다음 CFP_i와 CFP_j를 구성하고 있는 QS의 모든 잔기들을 포함하는 새로운 부분 구조 S'^{QS}를 생성한다(줄 5). 같은 방법으로 S'^{TS}도 생성한다(줄 6). 생성된 S'^{QS}와 S'^{TS}가 $MRPD_{SPP}$의 유사도를 만족하는지의 여부를 C4MRPD를 이용해 검사한다(줄 7). C4MRPD를 만족하는 경우 CFP_i와 CFP_j를 노드로 생성하여 이들을 에지로 연결해 그래프를 생성한다(줄 8-9). 이 후 최대 클리크를 찾고, 클리크를 구성하는 CFP들을 이용해서 SPP를 구성한다(줄 10-11). 마지막으로 길이가 작은 SPP를 제거하기 위해서 SPP들을 길이로 정렬하여 상위 $CUTOFF_{SPP}$개의 SPP만을 중첩으로 사용한다(줄 12-13).

알고리즘 2. SPP찾기

```
1  SPPSET = Ø /* set of sub-structure pairs */
2  GRAPH = Ø /* graph */
3  for each(CFPi in CFPSET)
4    for each(CFPj in CFPSET)
5      S'^QS = CFPi^QS∪CFPj^QS
6      S'^TS = CFPi^TS∪CFPj^TS
7      if(C4MRPD(S'^QS, S'^TS, MRPD_SPP))
8        construct nodes and an edge using CFPi
         and CFPj
9        insert them into GRAPH
10 find maximal cliques from GRAPH
11 insert maximal cliques into SPPSET
12 sort SPPSET by length
13 SPPSET = select CUTOFF_SPP of SPP from
   SPPSET
```

6.3 ASP 찾기

SPP를 중첩으로 사용하면 구조 중첩 문제를 해결하는 알고리즘 통해 유사도를 최대화 하는 변환 T'을 찾을 수 있다. QS를 T'을 통해 변환 후, TS와 정렬하면 SPP를 구성하는 잔기뿐만 아니라 그 외의 잔기 또한 정렬에 포함함으로써 더 좋은 정렬 결과를 얻을 수 있다. 이러한 잔기들을 포함함으로써 SPP를 ASP(Aligned Sub-structure Pair)로 확장해 나간다.

정렬 결과를 확장하기 위해서 본 논문은 MRPD를 증가시켜가면서 정렬에 참여 가능한 잔기의 쌍을 기존의 정렬 결과에 포함시키는 방법을 사용한다. 이때 한번에 증가시키는 MRPD의 증가값을 ITERMRPD라 하고 MRPD를 증가시키는 회수를 ITERCOUNT라 한다. 이 때 ITERMRPD와 ITERCOUNT의 조합을 어떻게 하는가에 따라서 확장 결과의 정확도와 속도가 결정된다. 즉 어떤 SPP의 MRPD가 0.5Å일 때, MRPD를 1.0Å까지 확장해 ASP를 구성하는 방법은 2가지로 나눌 수 있다. 1) 먼저 MRPD를 0.5Å에서 1.0Å로 바로 확장하는 경우가 있다. 2) 이와 달리 0.5Å에서 0.6Å, 0.7Å, 0.8Å, 0.9Å, 1.0Å과 같이 여러 번의 중간 단계를 거치면서 확장할 수도 있다. 후자가 전자에 비해서 더 좋은 결과들을 얻을 수 있는 반면에 중간 단계를 여러 번 거침으로써 정렬 시간이 오래 걸린다.

본 논문에서 사용하는 확장 알고리즘을 알고리즘 3에 기술한다. 먼저 확장하려는 유사도인 MRPD'을 구한다. MRPD'는 확장전의 유사도인 MRPD와 ITERMRPD를 더해서 결정한다(줄 6). 그리고 ASP를 이용해서 정렬에 사용된 변환 T'을 찾는다(줄 7). 그런 다음 찾은 T'를 이용해서 S^{QS}를 $T'S^{QS}$로 변환한다(줄 8). $T'S^{QS}$와 S^{TS}의 모든 잔기를 비교해서 MRPD'을 넘지 않는 잔기들을 추가해서 새로운 ASP'를 찾는다(줄 9-10). 찾은 ASP'를 다음 확장에 사용하기 위해서 ASPSET'에 추가한다(줄 11). 이 과정을 ITERCOUNT만큼 반복적으로 수행하여 최종적인 ASP를 찾는다.

알고리즘 3. ASP찾기

```
1  ASPSET = SPPSET /* set of sub-structure
   pair */
2  iter = 0
3  while(iter < ITERCOUNT)
```

```
4   ASPSET' = Ø /* set of sub-structure pair */
5   for(ASPi in ASPSET)
6     MRPD' = MRPD of ASPi + ITERMRPD
7     T' = transformation of ASPi
8     T'S^QS = structure transformed from S^QS by T'
9     ASP' = extended sub-structure pair of ASP
      whose MRPD is not over MRPD'
10    insert ASP' into ASPSET'
11  ASPSET = ASPSET'
12  iter = iter + 1
```

최종적으로 ASP들을 이용하여 4장에서 설명한 SG를 구성한다.

7. 실험

본 장에서는 본 논문에서 제안한 단백질 구조 정렬 알고리즘의 효율성을 다양한 실험들을 통해서 검증한다.

7.1 실험 환경

실험을 위해서 2006년 6월 PDB[24]로부터 다운 받은 PDB형식의 단백질 구조 데이타를 사용하였다. 정렬 결과로 사용한 유사도는 SG-Score이고 오류를 줄이기 위해서 매 실험마다 1000번씩 실시한 평균을 사용하였다. 실험에서는 펜티엄 4 2.4GHz인 CPU와 1Gbyte의 메인 메모리를 갖는 하드웨어를 사용하며 Fedora Core 5을 운영체제로 사용하였다.

7.2 파라미터 결정

본 논문에서 사용하는 파라미터는 $MRPD_{CFP}$, $CUTOFF_{CFP}$, $MRPD_{SPP}$, $CUTOFF_{SPP}$, ITERCOUNT, ITERMRPD이다. 이 중에서 ITERCOUNT와 ITERMRPD는 최적화 단계에서 사용하는 파라미터이므로 다른 파라미터를 모두 결정한 후 값을 정한다. 또한 $CUTOFF_{CFP}$와 $CUTOFF_{SPP}$는 각 알고리즘의 마지막 단계에서 사용되는 샘플의 개수이므로 값이 클수록 좋은 결과가 나온다. 따라서 $MRPD_{CFP}$와 $MRPD_{SPP}$을 결정하는 실험을 먼저 진행한 후에 $CUTOFF_{CFP}$와 $CUTOFF_{SPP}$를 결정하고 마지막으로 ITERCOUNT와 ITERMRPD를 결정한다.

유사도가 아주 높거나 낮은 단백질 구조의 쌍들은 파라미터를 결정하는 실험에서 제외하였다. 유사도가 아주 높거나 낮은 단백질들을 정렬하는 경우 파라미터의 값에 무관하게 정렬결과가 비슷하기 때문이다. 따라서 파라미터를 결정하기 위한 가장 좋은 데이타는 파라미터 변화에 민감한 중간 정도의 유사도를 가지는 단백질 구조들이다. 이를 위해 단백질의 구조에 따른 분류 방식인 SCOP(Structural Classification Of Proteins)[25]에 의해서 분류수준 중에서 도메인(Domain)보다는 유사도가 적고 슈퍼패밀리(Super-family)보다는 유사도가 큰 패밀리(Family) 수준의 유사도를 갖는 단백질 구조의 쌍을 실험에 사용하였다.

실험 1. $MRPD_{CFP}$와 $MRPD_{SPP}$의 결정

$MRPD_{CFP}$와 $MRPD_{SPP}$를 결정하기 위해서 먼저 $CUTOFF_{CFP}$와 $CUTOFF_{SPP}$, ITERCOUNT, ITERMRPD를 충분히 크거나 작은 값인 각각 500, 300, 200, 0.1Å으로 설정한 후 $MRPD_{CFP}$는 0.2Å부터 1.2Å까지 $MRPD_{SPP}$는 0.4Å부터 2.4Å까지 변화시켰다. 실험 결과, 표 1과 같이 $MRPD_{CFP}$와 $MRPD_{SPP}$가 각각 0.4Å과 1.6Å일 때 가장 좋은 SG-Score 0.59를 얻었다. 따라서 이후 실험에는 $MRPD_{CFP}$와 $MRPD_{SPP}$를 각각 0.4Å와 1.6Å을 사용한다.

실험 2. $CUTOFF_{CFP}$와 $CUTOFF_{SPP}$를 결정

다음 $CUTOFF_{CFP}$과 $CUTOFF_{SPP}$를 결정하는 실험을 수행한 결과를 표 2에 보인다. 이때 $MRPD_{CFP}$와 $MRPD_{SPP}$, ITERCOUNT, ITERMRPD의 값은 각각 0.4Å와 1.6Å, 200, 0.1Å 을 사용한다. 표 2에서 $CUTOFF_{CFP}$과 $CUTOFF_{SPP}$가 각각 200, 40을 넘어서면 정렬 결과가 거의 같은 것을 알 수 있다. 이것은 CFP와 SPP의 개수가 각각 200과 40을 넘은 후에는 더

표 2 $CUTOFF_{CFP}$과 $CUTOFF_{SPP}$의 변화에 따른 SG-Score

$CUTOFF_{CFP}$ \ $CUTOFF_{SPP}$	20	**40**	60	80	100
100	0.550	0.550	0.550	0.550	0.550
150	0.546	0.558	0.558	0.558	0.558
200	0.557	**0.566**	0.566	0.566	0.566
250	0.551	0.563	0.563	0.566	0.566
300	0.539	0.562	0.566	0.566	0.566

표 1 $MRPD_{CFP}$와 $MRPD_{SPP}$의 변화에 따른 SG-Score

$MRPD_{CFP}$ \ $MRPD_{SPP}$	0.4Å	0.8Å	1.2Å	**1.6Å**	2.0Å	2.4Å
0.2Å	0.370	0.497	0.536	0.538	0.533	0.483
0.4Å	0.363	0.520	0.577	**0.590**	0.585	0.556
0.6Å	0.453	0.522	0.567	0.582	0.581	0.568
0.8Å	0.482	0.501	0.539	0.574	0.576	0.568
1.0Å	0.499	0.507	0.560	0.572	0.574	0.566
1.2Å	0.498	0.505	0.541	0.559	0.567	0.558

많은 CFP와 SPP를 사용하더라도 더 좋은 정렬 결과를 찾기 어렵다는 것을 의미한다. 따라서 이후 실험에서는 $CUTOFF_{CFP}$과 $CUTOFF_{SPP}$를 각각 200과 40으로 결정한다.

실험 3. ITERCOUNT와 ITERMRPD를 결정

두 개의 파라미터 ITERCOUNT와 ITERMRPD는 서로 밀접하게 연관되어 있다. ITERCOUNT가 크고, ITERMRPD가 작을 때 가장 좋은 검색 결과를 얻을 수 있다. 하지만 ITERCOUNT가 커지는 만큼 정렬 시간은 증가한다. 따라서 정렬 시간과 유사도의 트레이드오프(tradeoff)를 고려해서 값을 찾는다. 표 4에서 ITERCOUNT가 증가함에 따라 정렬 시간이 선형적으로 증가하며, 표 3에서 ITERCOUNT가 30회 이상, ITERMRPD가 1.2Å 이상일 때 SG-Score의 증가량이 감소함을 알 수 있다. 따라서 추후 실험에서는 ITERCOUNT와 ITERRPD를 각각 30과 1.2Å로 사용한다.

표 3 ITERCOUNT와 ITERMRPD의 변화에 따른 SG-Score

ITERMRPD \ ITERCOUNT	10	20	**30**	40	50
0.4Å	0.378	0.498	0.550	0.571	0.578
0.8Å	0.459	0.559	0.577	0.582	0.583
1.2Å	0.509	0.572	**0.580**	0.581	0.581
1.6Å	0.535	0.576	0.579	0.579	0.579
2.0Å	0.549	0.575	0.577	0.577	0.577

표 4 ITERCOUNT와 ITERMRPD의 변화에 따른 정렬시간

ITERMRPD \ ITERCOUNT	10	20	**30**	40	50
0.4Å	0.434	0.722	1.039	1.349	1.618
0.8Å	0.490	0.767	1.025	1.273	1.510
1.2Å	0.474	0.768	**1.078**	1.345	1.642
1.6Å	0.506	0.760	0.999	1.230	1.385
2.0Å	0.426	0.658	0.885	1.113	1.334

7.3 CE와의 정렬 결과 비교 실험

제안하는 기법에서는 다수의 정렬 결과를 찾기 때문에 하나의 정렬 결과만을 찾는 타 논문과의 직접적인 비교는 어렵다. 대신 같은 RMSD를 갖는 경우의 정렬된 잔기의 개수인 N을 비교함으로써 간접적으로 정확도를 비교할 수 있다.

표 5, 6, 7은 CE에서 제시한 정렬 결과와 본 논문에서 제안한 방법의 정렬 결과를 비교한 것이다. 여기서 Similarity level은 정렬 하려는 단백질 구조 쌍의 SCOP방식에 따른 유사 수준을 나타내고 CE와 Proposed는 각각 CE와 제안한 알고리즘의 정렬 결과를 나타낸다. 정렬 대상인 단백질 패밀리는 모두 랜덤하게 선택하였다. 표 5의 정렬된 RMSD에 따른 N을 비교한 결과, CE와 비교하여 제안한 방식이 20개의 정렬 결과 중에서 2개는 적고 6개는 같고 나머지 12개에서 더 많은 정렬 결과를 얻을 수 있었다.

표 5 1ATP(E)와 cAMP-protein kinaze 패밀리를 이용한 정렬 결과

Target Protein			Similarity Level	RMSD	CE		Proposed	
PID	CID	n			N	Z-score	N	SG-score
1APM	A	350	Domain	0.33	336	7.9	336	0.93
1CDK	E	350	Domain	0.38	336	7.9	336	0.92
1YDR	E	350	Domain	0.46	336	7.9	336	0.92
1CTP	E	350	Domain	1.50	303	7.4	316	0.81
1PHK	_	298	Family	1.50	255	7.2	255	0.76
1KOA	_	491	Family	2.70	258	7.1	274	0.51
1KOB	A	387	Family	2.78	260	7.1	280	0.56
1AD5	A	438	Family	2.53	237	7.0	250	0.48
1CKI	A	317	Family	2.75	260	6.9	260	0.59
1CSN	_	298	Family	2.42	249	6.8	249	0.60
1ERK	_	364	Family	2.60	254	6.8	267	0.53
1FIN	A	298	Family	2.25	253	6.8	257	0.58
1GOL	_	364	Family	2.60	254	6.8	265	0.52
1JST	A	298	Family	2.45	253	6.7	251	0.57
1IRK	_	306	Family	3.31	244	6.5	256	0.46
1FGK	A	310	Family	3.50	251	6.2	255	0.52
1FMK	_	452	Family	2.82	245	6.2	251	0.45
1WFC	_	366	Family	3.06	240	5.6	256	0.47
1KNY	A	253	-	4.29	112	3.9	100	0.09
1TIG	_	94	-	4.15	54	3.9	80	0.45

표 6과 7은 다른 단백질 패밀리를 랜덤하게 선택한 후 CE와 비교한 실험 결과이다. 표 5와 동일하게 본 논문에서 제안한 방법이 대부분의 경우에 CE보다 정렬 결과가 더 좋음을 알 수 있다.

7.4 여러 유사도 레벨에 따른 SG-Score

다음은 SG-Score의 유용성을 검증하기 위해서 두 개의 단백질 구조 폴드(Fold) Ubiquitin-like과 Protein kinaze like에 대하여 SCOP의 분류 방식에 따라 도메인(Domain), 패밀리(Family), 슈퍼 패밀리(Super-family) 사이의 SG-Score를 측정해 보았다. 동일한 도메인, 패밀리, 슈퍼 패밀리에 속해 있는 단백질 구조들과 비교한 결과 각각 0.916~0.505, 0.754~0.415, 0,441~0.271정도

표 6 1MZN(A)와 NTF2-like 패밀리간을 정렬한 결과

Target Protein			Similarity Level	RMSD	CE		Proposed	
PID	CID	n			N	Z-score	N	SG-score
1jb2	A	127	Family	0.71	120	6.7	122	0.872
1oun	A	127	Family	0.29	124	6.8	125	0.926
1ask	A	127	Family	0.36	124	6.7	126	0.863
1ar0	A	127	Family	0.29	124	6.8	125	0.932
1qma	A	127	Family	0.77	123	6.7	123	0.907
1a2k	A	127	Family	0.71	121	6.7	121	0.897
1jkg	A	140	Family	1.43	118	6.2	120	0.782
1jn5	A	140	Family	1.45	118	6.2	120	0.745
1of5	A	221	Family	1.69	112	4.6	116	0.696
1q40	B	219	Family	1.80	114	4.1	111	0.682
1q42	A	201	Family	1.92	110	5.0	110	0.643

표 7 1NKS(A)와 Nuclear receptor ligand-binding domain 패밀리간을 정렬한 결과

Target Protein			Similarity Level	RMSD	CE		Proposed	
PID	CID	n			N	Z-score	N	SG-score
1sqn	A	261	Family	1.92	206	6.8	208	0.685
1sr7	A	259	Family	1.95	207	6.8	208	0.683
1a28	A	256	Family	2.02	210	6.7	209	0.678
1l2i	A	261	Family	1.77	203	6.6	203	0.681
3ert	A	261	Family	2.48	201	6.5	203	0.659
3erd	A	261	Family	2.00	210	6.8	210	0.697
1sj0	A	248	Family	1.89	191	6.3	193	0.682
1gwq	A	248	Family	1.85	207	6.8	208	0.699
1qkm	A	255	Family	2.26	205	6.6	206	0.674
1l2j	A	271	Family	1.97	192	6.2	195	0.644
1i7g	A	287	Family	2.48	218	6.2	216	0.651
1s9q	A	251	Family	1.56	183	6.8	195	0.690
1tfc	A	251	Family	1.65	210	6.9	210	0.742
1kv6	A	230	Family	1.58	208	7.0	209	0.749

의 SG-Score의 범위를 보였다. 즉, 비교하려는 단백질 구조 쌍의 유사도가 슈퍼 패밀리, 패밀리, 도메인 순으로 높아질수록 SG-Score가 높아짐을 알 수 있다. 이를 통해 SG-Score가 정렬하려는 단백질 구조의 유사도를 결정할 수 있는 지표로 사용될 수 있음을 확인할 수 있다.

7.5 시간 측정 실험

마지막으로 단백질 구조의 길이에 따른 정렬 시간을 실험하였다. 실험에 사용된 데이타는 동일 패밀리에 속한 단백질 구조의 쌍 중에서 랜덤하게 선택된 것을 이용하였다. 그림 8과 같이 CE와 Proposed 방식 모두 정렬하려는 단백질 구조의 길이가 커질수록 정렬 시간이 선형으로 증가함을 알 수 있다. 그러나 제안하는 방식에서는 다수의 정렬 결과를 제시함에도 불구하고, 정렬 시간은 단지 하나의 정렬 결과만을 제시하는 CE와 거의 비슷함을 알 수가 있다.

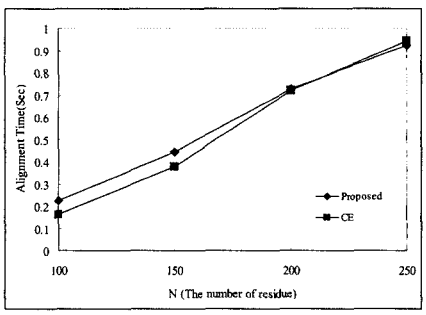


그림 8 단백질 구조의 길이에 따른 정렬 시간

8. 결 론

새로이 발견된 단백질의 기능을 예측하는 것은 생명공학에서 매우 중요하다. 따라서 최근에 단백질 기능을 유추하는데 사용되는 많은 단백질 구조 정렬 알고리즘이 발표되었다. 하지만 기존의 방법에서 사용되는 유사도가 실제 정렬 결과의 유사 정도와 잘 부합하지 않고 최종적으로 하나의 정렬 결과만을 찾기 때문에 사용자의 만족도를 높이는데 한계가 있었다. 따라서 본 논문은 이러한 기존 방식의 문제점들을 해결하기 위해서 다음과 같은 3가지 방법을 제안하였다.

첫째, 새로운 유사도 측정 방식인 MRPD를 제안하였다. 제안한 MRPD는 다음의 2가지 장점을 갖는다. 1) RMSD에 비해서 유사 정도를 직관적으로 이해할 수 있다. 2) 또한 정렬이 되지 않은 상태에서 주어진 단백질 구조들의 MRPD가 미리 지정된 임계 값을 넘는지의 여부를 판별할 수 있어, 정렬 시간을 크게 감소시킬 수 있다.

둘째, 다수의 정렬 결과들을 하나의 그래프로 함축하여 표현하는 SG 그래프 표현 방식과 그에 기반한 유사도 SG-Score를 제안하였다. 제안한 SG는 1) 정렬 길이에 따라서 정렬 결과를 표현하기 때문에 그래프의 모양을 통해서 단백질 구조간의 전체적인 유사도를 한눈에 파악할 수 있다. 2) 그래프의 기울기를 통해서 최적의 정렬 길이를 사용자가 보다 쉽게 판단할 수 있도록 한다. 3) MRPD뿐만 아니라 RMSD를 비롯한 다른 유사도들도 SG를 구성하는데 이용할 수 있어 확장성이 높다.

마지막으로 MRPD와 SG를 이용하는 새로운 단백질 구조 정렬 알고리즘을 제안하였다. 제안한 알고리즘은 하나의 정렬 결과를 찾는 기존의 방법들과 달리 다양한 정렬 길이에 따른 다양한 정렬 결과들을 찾는 알고리즘이다. 실험 결과에 따르면 제안된 방법은 다수의 정렬 결과를 제시하여 정렬 결과에 따른 사용자 만족도를 향상시킬 수 있었으며, 다수의 정렬 결과 검색에도 불구하고 정렬 시간은 기존의 방법들과 거의 비슷함을 알 수 있었다.

참 고 문 헌

[1] F. S. Collins, A. Patrinos, E. Jordan, A. Chakravarti, R. Gesteland, L. Walters, and the members of the DOE and NIH planning groups, "New

표 8 동일 슈퍼 패밀리를 구성하는 단백질 구조들을 이용한 도메인, 패밀리, 슈퍼 패밀리간의 SG-Score를 측정한 결과

	Ubiquitin-like			Protein kinaze like		
	Average	Minimum	Maximum	Average	Minimum	Maximum
Super-family	0.441	0.441	0.441	0.271	0.271	0.271
Family	0.581	0.415	0.754	0.598	0.598	0.598
Domain	0.773	0.505	0.890	0.885	0.816	0.916

Goals for the U.S. Human Genome Project: 1998-2003," Science, Vol.282, No.5389, pp. 682-689, 1998.
[2] L. Holm and C. Sander, "Protein structure comparison by alignment of distance matrices," Journal of Molecular Biology, Vol.233, pp. 123-138, 1993.
[3] B. Dahiya and S. Mayo, "De Novo protein design: fully automated sequence selection," Science, Vol. 278, pp. 82-87, 1997.
[4] P. E. Bourne and H. Weissig, Structural Bioinformatics, John Wiley & Sons Inc, 2003.
[5] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squaresfitting of two 3-D point sets," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-9, No.5, pp. 698-700, 1987.
[6] L. Chen, T. Zhou and Y. Tang, "Protein structure alignment by deterministic annealing," Bioinformatics, Vol.21, No.1, pp. 51-62, 2005.
[7] I. A. Jewett, C. C. Huang and T. E. Ferrin, "MINRMS: an efficient algorithm for determining protein structure similarity using root-mean-squared-distance," Bioinformatics, Vol.19, No.5, pp. 625-634, 2003.
[8] O. Camoglu, T. Kahveci and A. K. Singh, "Index-based Similarity Search for Protein Structure Databases," Journal of Bioinformatics and Computational Biology, Vol.2, No.1, pp. 99-126, 2004.
[9] I. Eidhammer and I. Jonassen, "Protein structure comparison and structure patterns - an algorithmic approach," ISMB tutorial, 2001.
[10] I. N. Shindyalov and P. E. Bourne, "Protein structure alignment by incremental combinatorial extension (CE) of the optimal path," Protein Engineering, Vol.11, No.9, pp. 739-747, 1998.
[11] W. R. Taylor and C. O. Orengo, "Protein structure alignment," Journal of Molecular Biology, Vol.208, pp. 1-22, 1989.
[12] W. R. Taylor, "Protein structure comparison using iterated double dynamic programming," Protein Science, Vol.8, pp. 654-665, 1999.
[13] I. Lotan and F. Schwarzer, "Approximation of Protein Structure for Fast Similarity Measures," Journal of Computational Biology, Vol.11, No.2-3, pp. 299-317, 2004.
[14] J. F. Gibrat, T. Madej and S. H. Bryant, "Surprising similarities in structure comparison," Current Opinion Structural Biology, Vol.6, No.3, pp. 377-385, 1996.
[15] W. Kabsch and C. Sander, "Dictionary of protein secondary structures: pattern recognition of hydrogen-bonded and geometrical features," Biopolymers, Vol.22, pp. 2511-2631, 1983.
[16] D. Frishman and P. Argos, "Knowledge-based protein secondary structure assignment," Proteins, Vol.23, pp. 566-579, 1995.
[17] L. Holm and C. Sander, "3-D lookup: Fast protein structure database searches at 90% reliability," Proceeding of International Conference on Molecular Biology, pp. 179-187, 1995.
[18] R. Nussinov and H. J. Wolfson, "Efficient detection of three-dimensional structural motifs in biological macromolecules by computer vision techniques," Proceeding of National Academy of Sciences of the USA, pp. 10495-10499, 1991.
[19] M. A. Erdmann, "Protein Similarity from Knot Theory: Geometric Convolution and Line Weavings," Journal of Computational Biology, Vol.12, No.6, pp. 609-637, 2005.
[20] F. N. Abu-Khzam, N. E. Baldwin, M. A. Langston and N. F. Samatova, "On the Relative Efficiency of Maximal Clique Enumeration Algorithms, with Applications to High-Throughput Computational Biology," Proceeding of International Conference on Research Trends in Science and Technology, 2005.
[21] C. Bron and J. Kerbosch, "Algorithm 457: finding all cliques of an undirected graph," Communications of the ACM, Vol.16, pp. 575-577, 1973.
[22] V. Stix, "Finding all maximal cliques in dynamic graphs," Computational Optimization Application, Vol.27, No.2, pp. 173-186, 2004.
[23] E. Tomita, A. Tanaka, and H. Takahashi, "The worst-case time complexity for generating all maximal cliques," Proceeding of 10th International Computing and Combinatorics Conference (LNCS 3106), pp. 161-170, 2004.
[24] RCSB Protein Data Bank (http://www.rcsb.org/pdb)
[25] A. G. Murzin, S. E. Brenner, T. Hubbard and C. Chothia, "SCOP: a structural classification of proteins database for the investigation of sequences and structures," Journal of Molecular Biology, Vol.247, pp. 536-540, 1995.

김 우 철
정보과학회논문지 : 데이타베이스
제 34 권 제 3 호 참조

박 상 현
정보과학회논문지 : 데이타베이스
제 34 권 제 1 호 참조

원 정 임
정보과학회논문지 : 데이타베이스
제 34 권 제 2 호 참조