서열 정보의 유사성 검사를 위한 가시화 도구

황미녕 강영민 조환규
부산대학교 전자계산학과
e-mail: {*mnhwang,ymkang,hgcho*}*@pearl.cs.pusan.ac.kr*

A Visualization Tool for Similarity Estimation of Sequence Data

Mi-Nyoung Hwang Young-Min Kang Hwan-Gue Cho
Department of Computer Science
Pusan National University

요 약

현재 활발한 연구가 진행 중인 유전자 분석과 같은 분야에서는 유전자 염기 서열과 같은 대규모 서열 정보들에 대한 효과적인 분석 기술을 요구하고 있다. 본 논문은 이러한 서열 정보들 사이의 유사도를 측정하고 분석하는 작업을 효과적으로 지원하기 위한 가시화 도구의 개발을 다룬다. 본 논문에서 사용하는 유사도 가시화 기법은 유전자 정보의 유사도 가시화를 위해 제안되었던 시각적 점-행렬 도면 (Graphical Dot-Matrix Plots) 기법을 이용하는데, 이 시각적 점-행렬 도면 기법은 비교 대상이 되는 서열 정보의 크기가 커지면 효율적으로 가시화하기가 힘들다는 단점을 가진다. 본 논문은 시각적 점-행렬 도면 기법의 이러한 문제를 해결하기 위해 서열 정보 유사도 비교 결과를 화면의 해상도 내에서 표현할 수 있도록 데이터를 영역별로 분할하고 각 영역별 일치도를 이분 그래프 (bipartite graph)의 최대 평면 일치 (maximal planar matching)를 이용하여 결정하고 이를 하나의 화소(pixel)로 출력하는 기법을 제안한다.

1 서론

생명공학 분야에 대한 연구가 활발해지고, 특히 유전자 분석과 같은 분야에서 단백질과 DNA 구조를 분석하고 비교하는 연구가 활발해짐에 따라 유전자 염기 서열과 같은 서열(sequence) 정보들에 대한 효과적인 분석 기술이 더욱 중요하게 되었다. 특히, 인체 게놈 프로젝트 등과 같이 매우 대용량의 서열 정보를 다루는 일이 중요한 연구로 자리를 잡게 되어, 이러한 대용량 서열 정보를 효율적으로 가시화하여 분석하고, 데이터들 사이의 유사도를 검사하는 기술에 대한 요구가 증가하고 있다.

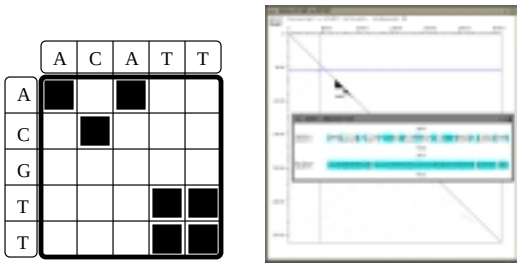
서열 정보들 사이의 유사도를 측정하고 분석하는 작업을 효과적으로 지원하기 위한 가시화 도구로는 모든 쌍을 비교하여 일치 여부를 행렬 형태로 표현하는 시각적 점-행렬 도면 (Graphics Dot-Matrix Plots) 기법이 있는데, 이미 단백질이나 DNA 염기 서열과 같은 서열 정보를 서로 비교하여 유사도를 평가하고 이를 가시화하는 방법으로 이 기법을 사용한 도구들이 제안되었다[2, 3].

시각적 점-행렬 도면 기법의 개념은 두 서열 정보를 각각 좌표축상의 수평과 수직으로 배치하여 전체 점-행렬의 열과 행이 되도록 하고, 행렬의 각 성분은 해당 열과 행의 데이터 원소의 일치 여부에 따라 점이 그려질지 그렇지 않을지가 결정되는 방법이다.

현재의 서열 정보는 하나의 비교 원소쌍을 하나의 화소(pixel)로 표현할 수 없을 정도로 대규모인 것인 일반적이다. 따라서, 서열 정보 유사도 비교 결과를 화면의 해상도 내에서 효율적으로 표현할 수 있는 기법이 요구된다.

2 관련 연구

점-행렬 (dot-matrix)를 이용하여 서열 정보 사이의 유사도를 가시화 하는 기법은 유전자 염기 서열 분석 분야에서 많은 연구가 있었다. Sonnhammer와 Durbin은 DNA와 단백질 서열 분석을 위해 이 기법을 사용하였다[3]. 이 기법의 기본적인 방법은 두 서열 정보 데이타를 행렬의 행과 열에 각각 배치하고 행렬의 원소는 행과 열에 있는 두 데이터 원소 사이가 일치하면 표시를 하는 것이다. 그림 1는 이 방법으로 점-행렬을 만드는 방법을 보이고 있다.



(a) (b)

그림 1: 점-행렬 기법 (a) 점-행렬을 구성하는 방법 (b) 두 개의 유전자 데이터에 대한 점-행렬 기법 적용 결과

이 방법은 하나의 화소를 검정 혹은 흰색으로 나타내게 되지만, 실제 방법에서는 그림 1와 같이 각 행렬 원소를 1 비트(bit)로 표현하는 것 (즉, 그리던가 그리지 않던가의 두 가지 경우로만 표현하는 것)이 아니고, 슬라이딩 윈도 (sliding window)라는 개념을 이용하여 각 원소 주위의 특정한 크기의 영역 전체의 일치 상태를 평가하여 하나의 화소를 흰색에서 검은색까지 255 가지 단계로 나누어 표현한다. 이렇게 슬라이딩 윈도 개념을 적용함으로써 두 서열 정보의 전체적인 유사도를 효율적으로 알 수 있게 한다. 여기서 두 데이터 사이의 유사성이 있는 부분은 출력된 점들이 행렬의 대각선 방향으로 나타나게 되며, 두 서열 정보 사이에서 일치되는 영역뿐만이 아니라, 일직선상의 반복(direct repeats), 역방향의 반복(inverted repeats) 등도 효율적으로 탐색하고 분석할 수 있다는 장점이 있다.

최근에는 이 시각적 점-행렬 도면 기법을 이용하여 웹 브라우저 (web-browser)상에서 단백질의 유사성 결과를 직접 볼 수 있는 도구에 대한 논문이 발표되어 관심을 끌었다[2].

3 기존 점-행렬 가시화 기법의 문제점

점-행렬을 이용한 유사도 가시화 기법은 두 개의 서열 정보를 매우 직관적이며 효율적으로 파악할 수 있는 방법을 제공해 준다. 그러나, 이 방법은 비교 대상이 되는 서열 정보의 크기가 커지면 효율적으로 가시화하기가 힘들다는 단점을 가진다. 이 점-행렬 도면 기법은 비교 대상이 되는 두 서열 정보의 모든 원소쌍에 대한 일치 여부, 혹은 일치 정도를 화면에 출력하기 때문에 서열 정보의 크기가 화면의 해상도 이상으로 증가하면 화면에 표현하기가 힘들다는 문제를 가진다.

전체 데이터의 유사도를 하나의 화면에 가시화하기 위해서는 기본적인 점-행렬 기법을 개선하여 전체 데이터가 하나의 화면에 모두 보여질 수 있도록 해야한다. 즉, 화면의 해상도보다 수평 또는 수직 방향의 원소 개수가 많아지면, 데이터를 영역별로 분할하고 이 영역을 하나의 원소처럼 취급하는 기법이 필요하다. 그림 2는 영역별 일치도를 이용하여 대규모 데이터를 효율적으로 가시화하는 방법의 기본 개념을 보이고 있다. 그림의 왼쪽은 비교 대상이 되는 원래의 데이터가 각각 여섯 개씩의 데이터 원소를 가지고 있는 경우이다. 그러나, 화면에 표시할 수 있는 해상도는 그림의 오른쪽과 같이 2×2 화소 밖에 되지 않는 경우 3×3 행렬 영역을 하나의 화소로 표현하는 것이다.

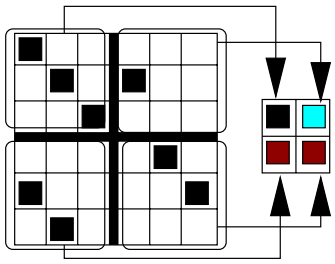


그림 2: 영역별 일치도를 하나의 화소로 표현하는 방법

OpenGL과 같은 그래픽 라이브러리를 이용할 경우 실제 화면 출력 장치의 해상도에 관계없이 원래의 좌표에 그림을 그리고, 카메라를 멀리함으로써 일정한 크기의 영역이 자동으로 하나의 화소에 그려지게 되지만, 이러한 방법은 다수의 화소 값으로 하나의 화소를 계산하는 것인데 이를 사용할 경우는 하나의 화소가 영역 전체의 일치도를 정확히 반영하지 못하고, 이미지 질의 저하(degradation)가 발생한다.

4 이분 그래프의 최대 평면 일치

두 서열 정보 각각의 일부 구간을 비교하기 위해서 우리는 그림 3와 같은 이분 그래프의 최대 평면 일치 (Maximal Planar Matching)를 이용하였다.

그림 3은 비교하는 두 서열 정보 데이타의 하나에서는 "ACTTGTA"라는 구간을 선택하고, 다른 하나의 데이터에서는 "ATTAGCA"라는 구간을 선택한 뒤, 이 두 구간을 서로 비교하여 하나의 화소에 표현하는 과정을 보이고 있다. 우리는 각 구간의 데이터 원소들을 하나의 정점 (vertex)으로 하는 이분 그래프를 구성하고, 이 이분 그래프에서 최대 평면 일치를 구하였다. 이분 그래프의 양쪽 정점들을 동일한 것들끼리 일치시킬 때 간선 (edge)의 교차가 일어나지 않는 일치를 평면 일치라고 하고, 최대 평면 일치는 이러한 평면 일치들 가운데 가장 많은 간선을 가진 집합을 의미한다. 즉, 비교 대상이 되는 두 구간의 데이터 원소들이 각각 V_1, V_2의 각

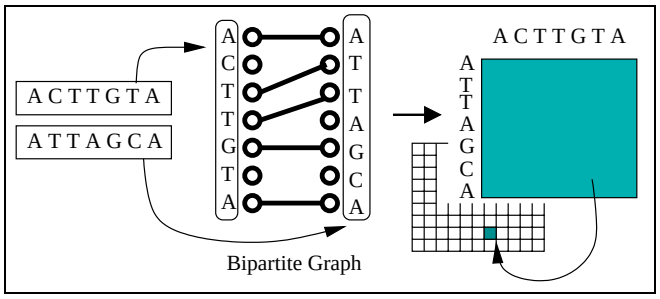


그림 3: 선형 데이터의 구간별 일치도 검사 및 화소 표현

각 n, m개씩의 정점을 가진 집합을 이루고, 이 정점 집합들은 서열 정보의 데이터 순서에 따라 차례로 $v_0, v_1, \cdots, v_n$과 $u_0, u_1, \cdots, u_m$의 정점을 가진다고 할 때, 이 이분 그래프의 평면 일치 $\mathbf{PM}(V_1, V_2)$는 다음과 같다.

$$\begin{aligned}\mathbf{PM}(V_1, V_2) \quad = \quad & \{(v_i, u_j) | v_i \in V_1, u_j \in V_2, \\ & \forall (v_i, u_j), (v_k, u_l) \in \mathbf{PM}(V_1, V_2), \\ & ((i < k) \wedge (j < l)) \vee ((i > k) \wedge (j > l))\}\end{aligned}$$

V_1, V_2에 대한 최대 평면 일치 $\mathbf{MPM}(V_1, V_2)$는 그림 3과 같이 두 정점 집합 사이의 교차하지 않는 최대 평면 간선 집합으로 모든 $\mathbf{PM}(V_1, V_2)$ 가운데 가장 많은 간선 수를 가진 것이다.

두 구간 사이의 일치도는 이 최대 평면 일치로 만들 수 있는 간선의 수와 서열 정보 구간의 길이의 비로 결정된다. 즉, 어떤 구간 (정점 집합) V_1와 다른 어떤 구간 V_2 사이의 일치도 $similarity(V_1, V_2)$는 다음과 같이 결정한다.

$$similarity(V_1, V_2) = \frac{|\mathbf{MPM}(V_1, V_2)|}{max\{|V_1|, |V_2|\}}$$

이때, $|\mathbf{MPM}(V_1, V_2)|$는 최대 평면 일치의 간선 수이다. 그리고, $|V_1|$와 $|V_2|$는 각각 V_1와 V_2의 원소 수 (정점 수)이다.

이 이분 그래프에서의 최대 평면 일치는 두 서열 (sequence) 정보 사이의 최대 공통 서브시퀀스 (Longest Common Subsequence - LCS)를 구하는 것과 동일하다. LCS란 주어진 두 개의 서열 정보 $A = \alpha_1\alpha_2\cdots\alpha_m$와 $B = \beta_1\beta_2\cdots\beta_n$에서 A와 B 각각에서 0 개 이상의 원소를 삭제하여 얻을 수 있는 최대 길이의 서브시퀀스를 의미한다. 이때 A와 B 각각에서 삭제하는 원소는 서로 달라도 무방하며, 인접해 있을 필요도 없다. 예를 들어, "abcca"와 "abacba"의 LCS는 "abca'가 된다. LCS는 스트링 처리 분야에서 많이 연구가 되었던 것으로 이를 구하는 방법 역시 잘 알려져 있다[1]. 또한 이 LCS는 유전자 정보 처리 분야에서도 두 서열 정보의 유사도 분석을 위해 많이 사용되었다.

5 실험 및 결과

본 논문에서 제시한 기법을 이용하여 두 가지 응용 분야에 적용해 보았다. 그 첫 번째는 두 프로그램 코드의 유사성을 검사하는 프로그램이고, 다른 하나는 DNA, RNA 등의 유전자 정보를 담고 있는 서열 정보들 사이의 유사도를 측정하는 프로그램이다. 유사도 측정 실험을 하고 그 결과를 보여보도록 하겠다.

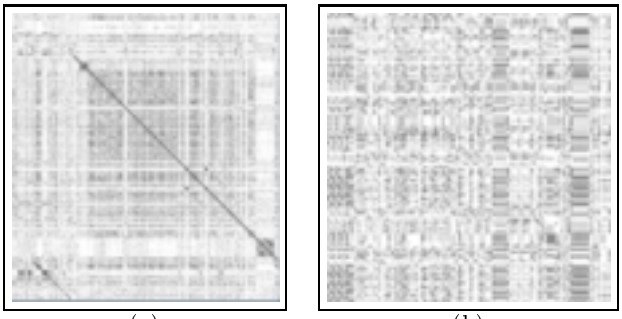
5.1 프로그램 유사성 검사

프로그램 유사성 검사의 목적은 학생들의 프로그램 과제 검사 시 자신이 직접 작성한 코드인지 다른 사람의 코드를 일부만 변경한 것인지를 검사하는데 사용될 수 있다. 프로그램

을 복사하더라 해도 부분적인 구조와 키워드(keyword) 사용의 위치는 바꾸기 힘들기 때문에 각 프로그램의 특성은 그대로 남아 있게 된다. 우리는 우리가 제안한 기법을 이용하여 이러한 특성들을 찾아 낼 수 있는지를 실험하였다.

두 프로그램 사이의 유사도를 검사하기 위해서는 우선, 각각의 프로그램 코드를 토큰 (token) 단위로 읽어들여서 미리 정의된 키워드를 추출한다. 이 키워드 서열 정보를 비교해서 일부 사선으로 나타나는 구간은 구조가 유사한 구간이라고 볼 수 있다. 이를 통해 프로그램 과제를 검사하는 것은 아무리 동일한 작업을 수행하는 프로그램이라 할지라도 서로 다른 사람이 독립적으로 프로그램을 작성한 경우에는 사용되는 키워드나, 논리의 흐름이 다를 수 밖에 없으며, 다른 사람의 프로그램을 수정하여도 원래 작성자의 논리 흐름이 키워드의 배치를 통해 그대로 남아 있다는 가정에 기반을 둔 것이다. 이 프로그램은 C와 Java로 작성된 프로그램을 대상으로 구현되었으며, 검사를 위해 추출되는 키워드는 C와 Java에서 사용되는 예약어들을 사용하였다.

그림 4의 (a) 각각 5,000 여 개의 키워드로 이루어진 두 프로그램 코드를 비교한 결과이다. 이 결과를 통해 우리는 이 두 프로그램이 비슷한 코드를 가지고 있다는 것을 확인할 수 있다. 반대로 그림 4의 (b)는 서로 완전히 다른 프로그램을 비교한 결과이다. 그림에서 확인할 수 있는 바와 같이, 서로 다른 프로그램은 대각선 방향의 선이 짙게 나타나지 않으며 특별한 특성이 발견되지 않는다.

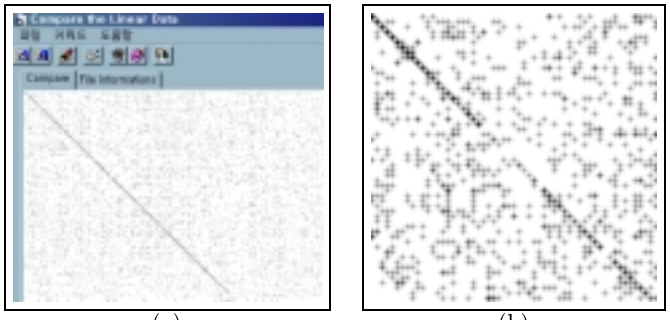


(a) (b)

그림 4: 프로그램 유사도 비교 결과 화면 (a) 유사한 프로그램의 비교 결과 (b) 전혀 다른 내용의 프로그램들의 비교 결과

5.2 DNA, RNA 유사성 검사

DNA나 RNA에 관한 연구가 활발해 지면서, 분자생물학 분야에서 다루는 대규모 서열 정보들 사이의 유사도를 검사하고 분석하는 일이 중요해졌다. 본 논문에서 제안한 기법은 이러한 분야에 적합한 가시화 기법으로 DNA나 RNA 데이터들을 비교하여 어떠한 구간이 서로 일치하며 어느 정도 유사한지를 직관적으로 파악할 수 있는 가시화 방법을 제공할 수 있다. 또한, 이러한 대규모 유전자 정보를 효율적으로 하나의 화면에 표현할 수 있다. 그림 5은 두 개의 단백질 데이타를 서로 비교한 결과이다. 사용된 단백질 데이터는 *Tecoma stans*의 NADH dehydrogenase subunit F와 *Clerodendrum trichotomum*의 NADH dehydrogenase subunit F이다. 그림 5의 (a)는 두 단백질 사이의 전체적인 유사도를 보여주며, 두 데이터 사이의 유사한 영역은 대각선으로 짙은 사선이 나타난다. 그리고, 이 사선 외에도, 서로 반복되는 작은 구간이 존재함을 알 수 있다. 이를 통해 한 서열 정보의 일부와 다른 서열 정보의 일부가 얼마나 자주 나타나는지도 쉽게 분석할 수 있으며, 특정한 구간이 뒤집혀서 나타나는 경우도 쉽게 찾을 수 있다. 그림 5의 (b)는 이 두 단백질이 서로 다른 모습을 나타내는 부분을 확대한 것으로 서로 비슷한 두 단백질이 어느 부분에서 서로 다른 특성을 갖는지를 쉽게 파악할 수 있다.



(a) (b)

그림 5: 두 개의 단백질 데이터 비교 : (a) *Tecoma stans*의 NADH dehydrogenase subunit F와 *Clerodendrum trichotomum*의 NADH dehydrogenase subunit F의 유사도 비교 (b) 두 단백질이 서로 다른 모습을 보이는 부분의 확대

6 결론

본 논문에서 우리는 서열 정보들 사이의 유사도를 평가하여 효율적으로 가시화할 수 있는 기법을 제안하였다. 이 기법은 이전에 제안된 시각적 점-행렬 도면 기법에 기반한 것으로 서열 정보들 사이의 유사도를 직관적인 행렬 표현을 통해 쉽게 분석하고 이해할 수 있게 해준다. 이전의 점-행렬 도면 기법이 제한된 해상도의 화면 때문에 대용량 데이터들 사이의 유사도를 효과적으로 가시화하기 어려웠던 것에 반해, 우리가 제안한 기법은 이분 그래프의 최대 평면 일치를 이용하여 구간별 일치 값을 구하고 이를 토대로 전체적인 유사도를 가시화 할 수 있는 기법으로 대규모 서열 정보에 적합하다. 본 논문에서 제안한 기법은 다음과 같은 특성을 가진다.

- 점-행렬 기법에 기반한 유사도 평가 방법을 사용하여 유사 영역이나 뒤집어진 영역과 같은 특징적인 구간을 직관적으로 파악할 수 있다.
- 구간별 일치도 평가 방법을 제공하여 유전자 데이터와 같은 대규모 서열 정보의 유사를 효율적으로 가시화할 수 있다.

우리는 구간 사이의 일치도를 평가하기 위해 LCS (longest common subsequence)를 찾아 그 길이와 전체 서열 정보 길이의 비를 일치도의 기준으로 삼았다. 그러나, 두 서열 정보의 일치도를 검사하는데 LCS가 가장 좋은 방법은 아니다. 더구나, 이 LCS 방법은 공통 서브시퀀스에서 인접해 있는 심볼들이 원래의 서열 정보에서 얼마나 떨어져 있었는지를 반영하지 못한다는 문제 등을 가지고 있다. 따라서 각 구간별 일치도 평가를 위한 더욱 개선된 방법에 대한 연구가 필요할 것이다.

참고문헌

[1] T. Cormen, C. Leiserson, and R. Rivest. Longest common subsequence. In *Introduction to Algorithms*, pages 314–320. McGraw-Hill, 1990.

[2] T. Junier and M. Pagni. Dotlet: Diagonal plots in a web browser. *Bioinformatics*, pages 178–179, 2000.

[3] E. Sonnhanmmer and R. Durbin. A dot-matrix program with dynamic threshold control suited for genomic dna and protein sequence analysis. *Gene*, pages GC1–10, 1996.