韓國컴퓨터情報學會 論文誌
第14卷 第5號, 2009. 5.

2009-14-5-1-4

근사 알고리즘을 이용한 순차패턴 탐색

산사볼트 가람라흐차 *, 황 영 섭 **

Searching Sequential Patterns by Approximation Algorithm

Sansarbold Garamragchaa *, Hwang Young Sup **

요 약

서열데이터베이스에 있는 자주 발현하는 부분 서열을 패턴으로 찾아내는 순차패턴 탐색은 넓은 응용 분야를 가지는 중요한 데이터 마이닝 문제이다. DNA 서열에서 순차패턴이 모티프가 될 수 있으므로 DNA 서열에서 순차패턴을 찾는 것을 연구하였다. 대부분의 기존 마이닝 방법은 순차패턴의 정의에 따라 정확한 정합에 주력하여 노이즈가 있는 환경이나 실제 문제에서 발생하는 부정확한 데이터에 대하여 제대로 작동하지 않을 수 있다. 이러한 문제가 생물 데이터인 DNA 서열에서 자주 나타난다. 이러한 문제를 다루기 위한 근사 정합 방법을 연구하였다. 본 연구의 아이디어는 자주 발생하는 패턴을 근사 패턴이라 부르는 그룹으로 분류할 수 있다는 관찰에서 기반을 둔다. 기존의 PrefixSpan 알고리즘은 주어진 긴 서열에서 순차패턴을 잘 찾을 수 있다. 본 연구는 PrefixSpan 알고리즘을 개선하여 유사한 순차패턴을 찾을 수 있게 하였다. 실험 결과는 PreFixSpan보다 제안한 방법이 패턴 길이가 4일 때, 근사 순차패턴의 빈도가 5배 높아짐을 보였다.

Abstract

Sequential pattern mining, which discovers frequent subsequences as patterns in a sequence database, is an important data mining problem with broad applications. Since a sequential pattern in DNA sequences can be a motif, we studied to find sequential patterns in DNA sequences. Most previously proposed mining algorithms follow the exact matching with a sequential pattern definition. They are not able to work in noisy environments and inaccurate data in practice. Theses problems occurs frequently in DNA sequences which is a biological data. We investigated approximate matching method to deal with those cases. Our idea is based on the observation that all occurrences of a frequent pattern can be classified into groups, which we call approximated pattern. The existing PrefixSpan algorithm can

• 제1저자 : 산사볼트 가람라흐차 교신저자 : 황영섭
• 투고일 : 2009. 03. 09, 심사일 : 2009. 04. 06, 게재확정일 : 2009. 05. 12.
* 선문대학교 전자계산학과 ** 선문대학교 컴퓨터공학부
※ 이 논문은 2006년 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임 (KRF-2006-311-D00833)

successfully find sequential patterns in a long sequence. We improved the PrefixSpan algorithm to find approximate sequential patterns. The experimental results showed that the number of repeats from the proposed method was 5 times more than that of PrefixSpan when the pattern length is 4.

▸ Keyword : 순차패턴(sequential pattern), 근사정합(approximate matching), 서열(sequences), PrefixSpan

Ⅰ. 서 론

순차패턴 탐색은 주어진 순차 데이터베이스에서 빈번히 발생하는 패턴을 찾는다[1]. 순차패턴은 경제/기업 정보분석, 웹 정보 마이닝 및 생물정보학 등 다양한 분야에서 활용되고 있다. 순차패턴 탐색 기법으로 제안된 선험적(a priori) 기법이 다양하다. 대표적인 방법인 GSP(Generalized Sequence Pattern) 기법은 다중 반복-후보 생성-테스트 방법을 적용하여 순차 패턴을 탐색한다[2-7, 13-16]. 선험적 방법인 GSP 방법보다 훨씬 빠른 것으로 보고되어 순차 패턴 검색에 널리 사용되는 알고리즘으로 FreeSpan과 PrefixSpan 알고리즘이 있다.[3-5].

대부분의 기존 탐색 알고리즘은 순차 패턴의 정의에 따른 정확한 정합을 목표로 한다. 그런데 정확성을 목표로 하면 DNA서열의 모티프처럼 노이즈가 있는 데이터나 부정확한 데이터를 잘 처리하지 못한다. 이런 문제를 해결하기 위하여 근사 정합(approximate matching) 방법을 연구하였다. 근사 순차패턴의 개념은 공통 배열(consensus pattern)을 탐색하기 위해 고안된 ApproxMap에서 제안되었다[8]. 공통 배열을 탐색하는 방법은 일반적인 거리 척도로 좋은 결과를 얻을 수 있다. 그러나 주어진 거리척도에서 순차패턴의 완전한 집합을 효과적으로 찾는 것은 여전히 어려운 문제이다.

본 연구에서 DNA서열의 모티프처럼 가장 빈도가 높은 패턴을 찾는 패턴 성장 방법인 PrefixSpan알고리즘을 적용하고 개선하여 더 일반적인 모델을 세운다. 지지 집합(support set)에 있는 서열(sequence)은 여러 형태로 바뀐 패턴이 있을 수 있다. 이들은 실제 몇 개의 그룹으로 분류할 수 있고, 이를 근사 패턴이라 부른다. 각 근사 패턴은 지지도와 함께 공통되는 패턴을 공유하는 서열의 집합이다.

데이터 마이닝에서 전통적으로 순차 패턴은 서열 데이터베이스에서 빈번히 나타나는 부분서열로 정의되었다. 이 정의에 따라 순차 패턴 탐색 기법에 기반을 둔 지지도 모델이 나왔다[3]. 지지도 모델로 순차 패턴을 탐색하는 방법이 집중적으로 연구되어 많은 연구 결과가 나왔지만 기존 모델은 기본적인 장애가 있다. 즉 노이즈가 있는 긴 서열을 처리하지 못한다는 점이다. 가장 널리 사용하는 순차패턴 탐색 방법은 패턴 성장에 의한 방법이다[3]. 탐색할 순차패턴을 지금까지 얻은 부분서열에 따라 나누고, 패턴의 분할에 따라 서열 데이터베이스를 투영한다.

본 연구의 기본 아이디어는 패턴을 키워나가며 탐색하면 관련된 근사 패턴의 집합을 찾을 수 있고, 이를 분류할 수 있다는 것이다. 처음 근사 패턴은 반복적으로 뭉쳐져 더 긴 근사패턴을 만들어 낸다. 이 과정은 국부적으로 더 이상 긴 근사 패턴이 발견되지 않을 때까지 반복된다. 두 번째 단계는 가장 길고 가장 반복되는 패턴을 일반화하고 그 구성 서열에 기반을 두어 지지도 집합을 생성한다. 지지도 집합을 이용하여 빈번한 근사 패턴을 찾아낸다. 고정된 크기의 창을 이용하여 빈번한 패턴을 찾아내는 대신에 제안하는 알고리즘은 모든 전체적인 빈발 근사 패턴을 탐색할 수 있다. 전체 데이터베이스를 반복하여 탐색하는 대신 후보 서열 집합을 크게 생성하여 테스트한다. 서열 데이터베이스를 지금까지 찾아낸 패턴과 관련된 서열집합으로 재귀적으로 투영할 수 있다. 그리고 각 투영된 데이터베이스에서 국부적으로 빈번한 패턴을 찾을 수 있다.

본 논문의 구성은 2장에 관련 연구를 요약하고, PrefixSpan 알고리즘을 설명한다. 3장에서 근사 PrefixSpan 알고리즘을 제안하고, 4장에서 제안한 알고리즘을 수행한 실험결과를 보이고 5장에서 결론을 맺는다.

Ⅱ. 관련 연구

이 절에서 우선 FreeSpan이라고 불리는 투영기반 순차패턴 탐색 기법을 요약한다. 그리고 이보다 개선된 PrefixSpan 알고리즘을 소개한다.

1. 용어와 문제의 정의

모든 구성요소의 집합을 $I=\{i_1, i_2, \cdots, i_n\}$이라 하자. 구

성요소 집합은 구성요소의 부분집합이다. 서열은 구성요소 집합의 순서 리스트이다. 서열 s 는 $\langle s_1, s_2, \cdots, s_L\rangle$로 표시한다. 여기서 s_i 는 하나의 구성요소 집합이다. 즉, $s_i \subseteq I$, $1 \le i \le L$. s_i를 서열의 원소라고도 부르며 $(x_1x_2\cdots x_m)$으로 표시한다. x_k 는 하나의 구성요소이다. 즉, $x_k \in I$, $1 \le k \le m$. 간략히 표기하기 위하여 하나의 원소가 하나의 구성요소를 가질 때 괄호를 표시하지 않는다. 즉, (x)는 x로 표시한다. 하나의 구성요소는 서열의 한 원소에 많아야 한 번 나타나지만, 서열의 다른 원소들에서 여러 번 나타날 수 있다. 서열에 있는 구성요소의 수를 그 서열의 길이라고 부른다. 길이가 L인 서열을 L-서열이라 부른다. 서열 $\alpha = \langle a_1a_2\cdots a_n\rangle$이 다른 서열 $\beta = \langle b_1b_2\cdots b_m\rangle$의 부분서열이 되려면 다음을 만족하는 정수들이 있어야 한다. 이때 β를 α의 포함서열이라 부른다.

$1 \le j_1 < j_2 < \cdots < j_n \le m$,

$a_1 \subseteq b_{j1}$, $a_2 \subseteq b_{j2}$, $\cdots$, $a_n \subseteq b_{jn}$.

서열 데이터베이스 S는 순서쌍 $\langle sid, s\rangle$의 집합이다. sid는 서열을 구별하기 위한 번호이고 s는 서열이다. 서열 α가 s의 부분서열이라면 순서쌍 $\langle sid, s\rangle$는 서열 α를 포함한다고 말한다. 서열 데이터베이스 S의 서열 α의 지지도는 α를 포함하는 데이터베이스의 순서쌍의 수이며 $support(\alpha)$로 표시한다. 서열 S와 사용자가 정의한 최소값 θ가 주어졌을 때, 그 서열에서 θ번 이상 발생한 부분서열을 빈번하다고 한다. 순차패턴은 빈번하면서 가장 긴 부분서열이다. 즉, $support(\alpha) \ge \theta$. 길이가 l인 순차패턴을 l-패턴이라 부른다.

예를 들어 "CTGATTAAA"는 길이가 9인 DNA 서열의 한 예이다. 여기서 구성요소는 A, G, T, C이고, TTA가 부분서열 중의 하나이다[10-12].

2. 관련된 알고리즘

이절에서 우선 FreeSpan이라고 불리는 투영기반 순차패턴 탐색 기법을 요약한다. 그리고 이보다 개선된 PrefixSpan 알고리즘을 소개한다. 두 방법은 투영된 데이터를 생성하지만 데이터의 투영 기준이 다르다. FreeSpan은 특정한 순서(성장 방향) 없이 현재의 빈번한 패턴에 기반하여 투영 데이터를 생성한다. 반면에 PrefixSpan은 빈번한 접두어를 성장시켜 데이터를 투영한다. PrefixSpan이 대부분의 서열 데이터에서 FreeSpan보다 빠르다.

2.1 FreeSpan

FreeSpan은 다음 성질에 기반하고 있다: 어떤 구성요소 집합 X가 빈번하지 않다면 투영된 구성요소 집합이 X의 포함집합이 되는 어떤 서열도 순차패턴이 될 수 없다. FreeSpan은 탐색공간을 분할하여 투영된 구성요소 집합에 기반하여 재귀적으로 서열의 부분 데이터를 투영하여 순차패턴을 탐색한다.

표 1. 서열 데이터
Table 1. Sequence data

Sequence_id	Sequence
10	<a(abc)(ac)d(cf)>
20	<(ad)c(bc)(ae)>
30	<(ef)(ab)(df)cb>
40	<eg(af)cbc>

표 1의 서열 데이터가 주어지면 FreeSpan은 각 구성요소의 지지도를 수집하여 빈번한 구성요소의 집합을 찾는다. 표 1에서 구성요소의 집합은 {a, b, c, d, e, f, g}이다. 10번 서

표2. 투영 데이터베이스와 순차패턴
Tabel 2. Projected database and sequential pattern

<a>	<(abc)(ac)d(cf)>, <(_d)c(bc)(ae)>, <(_b)(df)cb>, <(_f)cbc>	<a>, <aa>, <ab>, <a(bc)>, <a(bc)a>, <aba>, <abc>, <(ab)>, <(ab)c>, <(ab)d>, <(ab)f>, <(ab)dc>, <ac>, <aca>, <acb>, <acc>, <ad>, <adc>, <af>
<b>	<(_c)(ac)d(cf)>, <(_c)(ae)>, <(df)cb>, <c>	<b>, <ba>, <bc>, <(bc)>, <(bc)a>, <bd>, <bdc>, <bf>
<c>	<(ac)d(cf)>, <(bc)(ae)>, <b>, <bc>	<c>, <ca>, <cb>, <cc>
<d>	<(cf)>, <c(bc)(ae)>, <(_f)cb>	<d>, <db>, <dc>, <dcb>
<e>	<(_f)(ab)(df)cb>, <(af)cbc>	<e>, <ea>, <eab>, <eac>, <eacb>, <eb>, <ebc>, <ec>, <ecb>, <ef>, <efc>, <efcb>
<f>	<(ab)(df)cb>, <cbc>	<f>, <fb>, <fbc>, <fc>, <fcb>

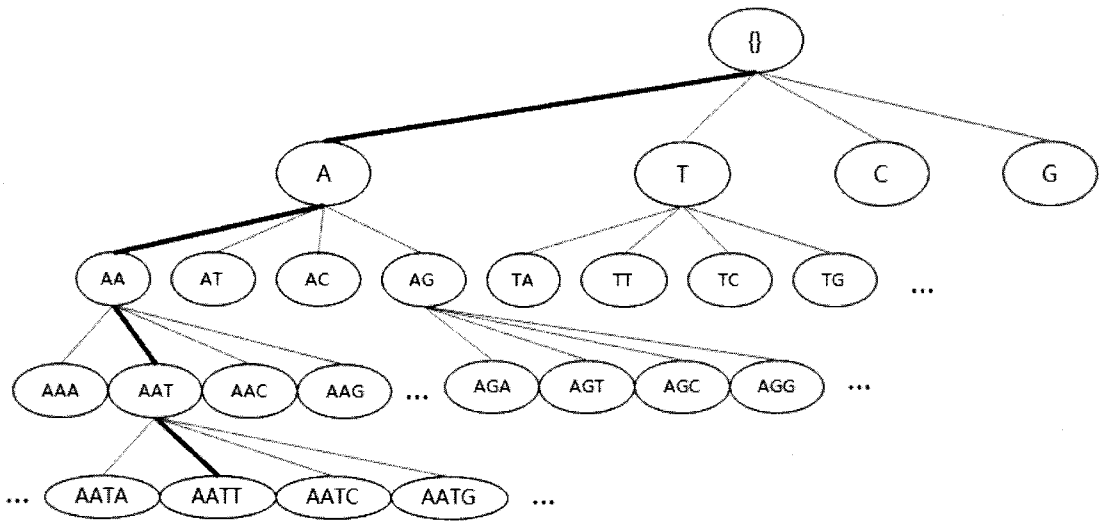

그림 1 PrefixSpan 알고리즘 실행 예. 굵은 실선이 DNA 서열 CTGATAAA에서 AATT 패턴을 찾는 경로이다.
Fig. 1. Illustration of PrefixSpan algorithm execution. Bold line shows the path of finding AATT pattern in a DNA sequence CTGATAAA

열 $\langle a(abc)(ac)d(cf)\rangle$은 다음 다섯 개의 원소를 갖고 있다: (a), (abc), (ac), (d), (cf). 구성요소 a와 c는 서로 다른 원소에서 한 번 이상 나타난다. 10번 서열은 서열에 있는 구성요소의 수가 9이므로 9 서열이다. 구성요소 a가 3번 나타나므로 서열의 길이에 3을 더하지만 $\langle a\rangle$의 지지도에는 1을 더한다. 서열 $\langle a(bc)df\rangle$는 $\langle a(abc)(ac)d(cf)\rangle$의 부분서열이다. 10번과 30번 서열이 모두 부분서열 $s = \langle(ab)c\rangle$를 포함하고 있으므로 s는 길이 3인 순차패턴(즉 3-패턴)이다.

표 1의 데이터에 대해 θ를 2로 하여 빈번한 구성요소를 구하고, 지지도에 따라 내림차순으로 (구성요소: 지지도)를 정렬하면 다음과 같다.

f_list = a: 4, b: 4, c: 4, d: 3, e: 3, f: 3

순차패턴의 부분집합은 투영 데이터베이스를 구성하여 탐색할 수 있다. g처럼 빈번하지 않는 구성요소는 투영 데이터베이스를 구성할 때 제외한다. 탐색 과정은 다음과 같다.

- 구성요소 a만을 포함하는 순차패턴을 찾는다. 서열 데이터베이스를 한 번 탐색하면 구성요소 a를 포함하는 두 개의 순차 패턴 $\langle a\rangle$와 $\langle aa\rangle$를 찾을 수 있다.
- 구성요소 b를 포함하지만 f_list의 b 다음 구성요소가 없는 순차패턴을 찾는다. $\{b\}$ 투영 데이터베이스를 구성한다. f_list의 b 다음에 있는 구성요소를 제거하면 다음의 $\{b\}$ 투영 데이터베이스는 다음 4 서열을 포함한다: $\langle a(ab)a\rangle$, $\langle aba\rangle$, $\langle(ab)b\rangle$, $\langle ab\rangle$. 투영 데이터베이스를 한 번 더 조사하여 b를 포함하지만 b 다음 구성요소가 없는 순차 패턴을 찾으면 $\langle b\rangle$, $\langle ab\rangle$, $\langle ba\rangle$, $\langle(ab)\rangle$이다.
- 순차패턴의 다른 부분집합을 찾는다. 순차패턴의 다른 부분집합도 해당 투영 데이터베이스를 구성하고 그들을 재귀적으로 탐색하여 비슷하게 찾을 수 있다.

2.2 PrefixSpan

PrefixSpan 알고리즘의 기본 아이디어는 "어떤 빈번한 부분서열은 빈번한 접두어를 증가시켜서 항상 찾을 수 있다"에서 나왔다. 표1의 데이터베이스를 이용하여 알고리즘을 간략히 소개하면 다음과 같다.

- 용어정리 (접두어, 투영): 원소의 모든 구성요소가 알파벳 순서로 나열된다고 가정한다. 서열 $\alpha = \langle e_1e_2\cdots e_n\rangle$의 접두어 $\beta = \langle e'_1e'_2\cdots e'_m\rangle$는 다음 조건을 만족해야 한다. (1) $e'_i = e_i$, $i \le m-1$; (2) $e'_m \subseteq e_m$; (3) $(e_m - e'_m)$에 있는 모든 구성요소는 e'_m보다 알파벳 순서로 뒤쪽이다.

  서열 β가 서열 α의 부분서열이고 α의 부분서열 α'가 다음을 만족하면 α'을 접두어 β에 대한 α의 투영이라고 부른다. (1) α'이 접두어 β를 가진다. (2) α의 부분서열이면서 접두어 β를 가지면서, α'이 α''의 부분서열이 되며 둘이 같지 않은 α''이 존재하지 않는다.
- 1단계: 길이 1인 순차패턴을 찾는다. 서열에 있는 모든 빈번한 구성요소를 찾기 위하여 데이터베이스를 한 번 탐색한다. 그 결과는 $\langle a\rangle$:4, $\langle b\rangle$:4, $\langle c\rangle$:4, $\langle d\rangle$:3, $\langle e\rangle$:3,

〈f〉:3이다. 숫자는 지지도를 표시한다.

- 2단계: 탐색공간을 분할한다. 순차패턴의 전체 집합은 1단계에 찾은 여섯 개의 접두어에 따라 6개의 부분집합으로 나눌 수 있다. 즉, 접두어 〈a〉로 시작하는 부분집합, 접두어 〈b〉로 시작하는 부분집합, …, 접두어 〈f〉로 시작하는 부분집합이다.
- 3단계: 순차패턴의 부분집합을 찾는다. 순차패턴의 부분집합은 해당 투영 데이터베이스를 만들고 재귀적으로 탐색하여 구할 수 있다. 표2에 투영 데이터베이스와 찾아낸 순차패턴을 보인다.

III. 근사 PrefixSpan 알고리즘

PrefixSpan 알고리즘은 투영 데이터베이스를 탐색해야 한다. 그런데 우리의 주요 연구대상인 DNA 서열은 매우 긴 연속적인 서열이다.

순차패턴을 찾는 문제에서 중요한 점은 최소 지지도를 가지는 모든 부분서열을 찾아야 한다는 것이다. 어려운 점은 찾아야 할 서열이 무엇인가를 파악하고 빈번한 것을 효율적으로 찾아야 한다는 것이다[2].

패턴을 키워서 순차패턴을 찾는 것은 비교적 최근의 접근방법이다. 핵심 아이디어는 후보를 생성하는 단계를 피하고 주어진 데이터베이스의 제한된 영역을 탐색하는 데 초점을 두는 것이다. PrefixSpan은 패턴을 키우는 방법 중에서 가장 유망하며, 패턴을 재귀적으로 구성하며 동시에 탐색을 투영 데이터베이스로 한정하는데 기반하고 있다[3]. 데이터베이스는 접두어를 가지는 서열의 뒷부분으로 구성된 부분서열의 집합이다. 각 단계에서 특정 접두어에 대한 투영 데이터베이스에서 접두어를 가지는 빈번한 서열을 찾는다. 각 단계에서 탐색 공간이 줄어들어 더 좋은 성능을 낼 수 있다. 그런데 갭(gap) 제한이 있으면 이러한 장점이 유효하지 않게 된다. 그러나 새로 개선한 근사 PrefixSpan 알고리즘은 다른 방법보다 더 빠르며 장점도 그대로 유지한다.

3.1 근사 정합(approximate matching)

전통적인 정합방법의 근본적인 문제점은 데이터에 있는 노이즈(noise)를 고려하지 않고 완전 정합(exact match)을 하는 데 있다. 여기에 두 가지 문제가 있다. DNA 서열과 같은 실세계의 데이터에서 긴 패턴은 노이즈가 있는 경향이 있고, 완전 정합에서 요구하는 지지도 수준을 맞출 수 없을 수 있다. 노이즈가 적더라도 빈번한 긴 패턴이 빈번하지 않다고 판정될 수 있다[8]. 이런 경우에 완전 정합보다 근사 정합이 더 적당하지만 이 분야는 연구가 충분하지 않았다.

우리는 전통적인 서열 패턴 탐색 모델을 확장하여 근사 서열 패턴을 탐색할 수 있게 하였다. 최소 허용 거리가 주어지면 근사 정합에 의해 생성된 패턴은 문자열 정합 전이(transition)에 의해 결정된다. 가능한 서열을 모두 생성하는 대신에 가능한 오류를 고려하여 초기 상태에서 시작하여 오토마톤의 유효한 전이인지 확인한다. 한 원소가 유효한 전이에 해당되지 않을 때마다 알고리즘은 이를 대치(Replacement)하려 한다. 만약 실패하면 무시(Deletion에 해당)하고, 마지막으로 유효한 전이(Insertion에 해당)를 시도한다.

Insertion : Ins(x, i) - 구성요소 x를 i위치에 추가한다.
Deletion: Del(x, i) - 서열의 i위치에 있는 구성요소 x를 제거한다.
Replacement: Repl(x, y, i) - i위치에 있는 구성요소 x를 구성요소 y로 대치한다.

(예제) DNA 서열에서 한 모티프인 ATAT를 찾는다고 하자. 근사 정합에 의하면 다음 3가지 경우가 생긴다.

Insertion: ATAAT
Deletion: ATT
Replacement: ATGT

불행히도 위 모델은 다음 두 가지 문제점이 있다. (1) 수많은 짧고도 대개 의미 없는 패턴을 찾게 된다. 짧은 패턴은 긴 패턴보다 더 유사도 카운트를 가지게 되고 전체 결과의 대부분을 차지하게 된다. (2) 근사 순차패턴이 완전 정합 순차패턴보다 훨씬 커져서 이해하기 어렵게 된다. 사용자는 근사를 통해 노이즈는 무시하면서 전체적인 경향을 파악하기를 원할 것이다. 그런데 위 모델의 근사 패턴은 의미 없는 짧은 패턴을 많이 발생시켜 정보의 탐색을 방해하게 된다.

3.2 근사 PrefixSpan 알고리즘

PrefixSpan 알고리즘은 정확한 패턴을 찾아내고, 근사 정합은 관련된 근사 패턴을 알아낼 수 있다. 둘을 결합하여 근사 PrefixSpan 알고리즘을 개발하여 순차패턴을 찾아낼 수 있게 하였다. 근사 PrefixSpan 알고리즘의 입력은 길고 연속된 DNA 서열이다. 알고리즘은 패턴을 생성하는 부분과 패턴을 탐색하는 두 부분으로 나뉜다. 생성 부분은 다음 탐색 단

계에서 사용할 빈번한 패턴을 정의한다. 생성 부분은 짧은 빈번한 패턴에서 더 긴 순차패턴을 생성하고 길이와 빈도로 구분한다. 탐색 부분은 근사 정합으로 부분 서열을 찾아서 빈발 패턴을 찾아낸다. 규칙은 두 서열을 근사 전이를 이용하여 비교하는 것이다.

I. 순차패턴 생성
- PrefixSpan의 단계 1과 단계 2에 해당된다.
- 패턴 성장 기법에 의해 검색할 순차 패턴을 생성한다.
- DNA 서열의 경우, (A, G, T, C)의 4개 부분집합을 생성한다.

II. 근사 순차패턴 검색
각 순차패턴에 대해
1. 주어진 서열에서 순차패턴의 시작 위치를 탐색한다.
2. 에러 한계(보통 1 또는 2) 이내의 전이만을 검색한다.
   2.1 유효한 전이인지 조사한다. 유효하면 검색을 끝낸다.
   2.2 유효한 전이가 아니면 다음 순서로 검색하고 실패한 경우만 다음 순서로 간다.
      A. Replacement를 수행하고 검색한다.
      B. Deletion을 수행하고 검색한다.
      C. Insertion을 수행하고 검색한다.

그림 2. 근사 PrefixSpan 알고리즘
Fig. 2. Approximate PrefixSpan algorithm

길이가 1에서 3인 짧은 패턴을 탐색할 때 충돌이 생길 수 있다. 또한 패턴 길이가 너무 길면 패턴 탐색 결과가 축소될 수 있다. 미리 정의한 최소 길이와 최대 길이를 이용하여 이 문제를 해결하였다. 전이는 사용자가 정의 한 에러한계(보통 1 또는 2)에 의해 제한된다. 모든 가능한 구성요소가 (A, C, G, T) 이므로 에러한계가 1 증가하면 탐색 시간은 적어도 3배 늘어나게 된다. 근사 정합과정은 유효한 전이가 아니라면, 모든 가능한 요소 (A, G, C, T)에 대하여 조사하는 대신, 대치(Replacement)를 시도하고, 여기서도 실패하면 Deletion을 시도하고, 그래도 실패하면 Insertion을 수행한다.

IV. 실험결과

본 연구의 목표는 근사를 통해서 알려지지 않은 정보를 발견할 수 있다는 것의 유효함을 보이는 것이다. 이를 위해 일반적인 PrefixSpan 알고리즘과 근사 PrefixSpan 알고리즘을 DNA 서열의 순차패턴을 찾는데 적용하여 비교하였다. 실험환경은 일반적인 Windows XP를 사용하는 PC의 Visual C++ 6.0이다. 실험에 사용한 DNA 서열은 길이가 27,126인 길고 연속된 텍스트 파일이다. 서열의 처음 100개 부분은 다음과 같다.

AAGTATTGCGCATACTTCTTATGGGAGATCC
AAATTCCCAGTACATTAGTTGACTCTCCTTC
CTTAATGTTTTCCTCTGTACGATTATTTATG
TTTAAAC…

서열은 A, G, C, T의 4가지 염기 이름으로 구성된다. 알고리즘의 출력은 가장 빈도가 높은 패턴과 빈도이다. 패턴의 길이가 3일 때 제안한 알고리즘의 기존 PrefixSpan 알고리즘보다 2번 더 반복되었고 마지막 염기 이름이 달랐다. 긴 DNA 서열에서 노이즈는 어떤 전이를 포함하는 것을 말하므로, 순차패턴을 탐색할 때 정확한 정합보다 근사 정합을 통해 숨어 있는 정보를 찾는 것이 중요하다. 제안한 방법은 이러한 전이가 포함된 빈번한 패턴을 찾아낸다.

표 3. 제안한 방법과 PrefixSpan과의 실험결과 비교
Table 3. Comparison of experimental results between the proposed method and PrefixSpan

패턴 길이	1	2	3	4	5
PrefixSpan	T	TT	TTT	TTTT	TTTTT
제안한 방법	T	TT	TTA	TTAT	TTATT
PrefixSpan의 결과	9,288	2,646	936	270	90
제안한 방법의 결과	9,288	2,646	2,015	1,350	520

표 3과 그림 3에 제안한 방법과 PrefixSpan 방법으로 얻은 결과를 비교하였다. 찾아낸 패턴의 빈도가 근사 방법을 적용하면서 길이가 3일 경우 936에서 2,105로 크게 증가하였다. 제안한 방법은 다른 방법보다 항상 빈도가 더 많으며 실험 동안 상이한 패턴도 찾아내었다. 따라서 제안한 방법이 DNA 서열에 있는 숨겨진 순차패턴을 효과적으로 찾아낸다고 볼 수 있다.

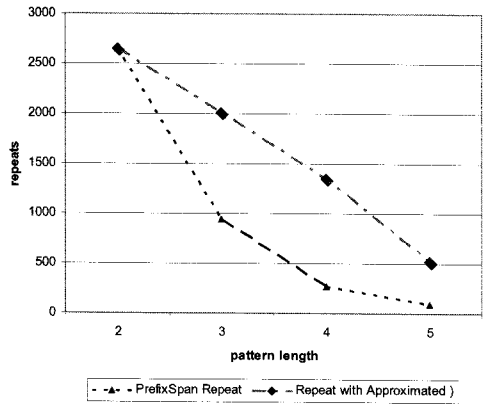

그림 3. 패턴 길이와 순차패턴의 빈도
Fig. 3. The number of repeats with pattern length

V. 결론

빈번한 근사 순차패턴을 탐색하기 위하여 제안한 방법은 기존의 PrefixSpan 알고리즘보다 더 효율적으로 근사 순차패턴을 찾아냄을 실험을 통하여 보였다. 제안한 방법의 기본적인 아이디어는 기존의 PrefixSpan 알고리즘의 패턴 성장 방법에 근사 문자열 정합 방법을 적용하여 순차 패턴을 찾는 것이다.

실험결과 DNA 서열에서 전체적으로 빈도가 높은 근사 순차패턴을 찾을 수 있었다. 찾아낸 패턴의 빈도가 패턴 길이가 4일 때 5배 정도 더 많았다. 노이즈가 있거나 일부 부정확한 순차패턴이 있는 긴 서열에 대해 제안한 알고리즘은 근사 순차패턴을 대부분 찾아낼 수 있다. DNA 서열의 모티프처럼 노이즈가 있거나 일부 부정확한 순차패턴이 있는 경우, 이를 모두 찾아내면 유전자의 발현 분석이나 신약개발 등에 도움을 줄 수 있다. 즉 정확한 순차패턴이 아니라 근사 순차패턴을 모두 찾아내야 하는 응용분야에 제안한 알고리즘은 적합하다.

근사 순차패턴을 찾을 때 주어진 문제에 적합한 제약조건을 두면 탐색되는 패턴의 수를 줄일 수 있다. 제약조건의 예를 들면 삽입, 삭제와 대체에 대하여 가능과 불가능의 조합으로 제약조건을 설정할 수 있다.

찾아낸 근사 순차패턴에서 유효한 정보를 찾아내는 것이 향후 연구과제이다. 예를 들어, 실험에 사용한 DNA 서열에서 근사 순차패턴을 찾아내었는데 이 중에 어느 것이 모티프인지 알아내는 방법을 연구할 수 있다.

참고문헌

[1] R. Agrawal and R. Srikant. "Fast algorithms for mining association rules," In Proc. 1994 Int. Conf. Very Large Data, Bases (VLDB'94), pp.487-499, Santiago, Chile, Sept. 1994.

[2] Antunes C and Oliveira A.L: "Generalization of Pattern-Growth Methods for Sequential Pattern Mining with Gap Constraints," in Int'l Conf Machine Learning and Data Mining, pp.239-251, 2003.

[3] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, Mei-Chun Hsu "Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach," IEEE Transactions on knowledge and engineering , Vol.16, No.10, pp.1424-1440, 2004.

[4] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu, "PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth," Proc. 2001 Int. Conf. on Data Engineering (ICDE'01), Heidelberg, Germany, pp.215-224, April 2001.

[5] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M.-C. Hsu, "FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining," Proc. 2000 Int. Conf. Knowledge Discovery and Data Mining (KDD'00), Boston, MA, pp.355-359, Aug. 2000.

[6] M. J. Zaki, "SPADE: An Efficient Algorithm for Mining Frequent Sequences," in Machine Learning Journal, special issue on Unsupervised Learning (Doug Fisher, ed.), pp.31-60, Vol.42(1/2), Jan/Feb 2001.

[7] J. Han, J. Pei, and Y. Yin. "Mining frequent patterns without candidate generation," In Proc. 2000 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'00), pp.1-12, Dallas, TX, May 2000.

[8] H.C. Kum, J. Pei, W. Wang, and D. Duncan. Approx-MAP : "Approximate Mining of Consensus Sequential Patterns," Technical Report TR02-031, UNC-CH, 2002.
[9] Antunes, C. and Oliveira, A.L., "Sequential Pattern Mining with Approximated Constraints," Int. Conf Applied Computing, IADIS, pp.131-138, 2004.
[10] 이병일, 이종연, 정순기, "클러스터링 분기를 이용한 다중 서열 정렬 알고리즘", 한국 컴퓨터정보학회 논문지, 제10권 제5호, 1-10쪽, 2005년 11월.
[11] 홍창범, 차정호, 이성훈, 신승우, 박근준, 박근용, "클러스터링 환경에서의 MPI 기반 병렬 서열 유사성 검색에 관한 연구", 한국 컴퓨터정보학회 논문지, 제11권 제6호, 69-78쪽, 2006년 12월.
[12] 남성혁, 김태경, 김경란, 조완섭, "서비스 지향 구조 기반의 EST 서열 주해 시스템", 한국 컴퓨터정보학회 논문지, 제13권 제3호, 35-44쪽, 2008년 5월.
[13] 김학자, 황환규, "점진적인 순차 패턴 갱신 알고리즘", 전자공학회 논문지, 제43권 제5호, 17-28쪽, 2006년 9월.
[14] 강태호, 유재수, "생물학적 서열들에서 빈발한 연속 서열 패턴 마이닝", 한국컴퓨터종합학술대회 논문집, Vol.34, No.1(B), 27-31쪽, 2007년 6월.
[15] 금혜정, 장중혁, "대용량 순차 데이터베이스에서 근사 순차패턴 탐색", 정보처리학회논문지D, 제13-D권 제2호, 199-206쪽, 2006년 2월.
[16] 허용도, 조동영, 박두순, "서픽스 검사를 이용한 단계적 순차패턴 분할 탐사 방법", 멀티미디어학회 논문지, 제5권 제5호, 590-598쪽, 2002년 10월.

저 자 소 개

산사볼트 가람라흐차

2002년 2월: 몽골 국립대학교 학사
2009년 2월: 선문대학교 전자계산학과 석사
관심분야: 바이오인포매틱스, 데이터마이닝

황영섭

1989년 2월: 서울대학교 컴퓨터공학과 학사
1991년 2월: POSTECH 컴퓨터공학과 석사
1997년 2월: POSTECH 컴퓨터공학과 박사

1997년-2002년: 한국전자통신연구원 선임연구원
2002년-현재: 선문대학교 컴퓨터공학과 조교수
관심분야: 바이오인포매틱스, 문서인식, 신경회로망, 패턴인식, 영상처리