접두사 그래프 모델을 이용한 최장공통비상위문자열 찾기

최시원[O1] 이도경[2] 김동규[3] 심정섭[1]
[1]인하대학교 컴퓨터정보공학부, [2]서울대학교 컴퓨터공학부, [3]한양대학교 전자통신컴퓨터공학부
siwonred@gmail.com, domeng@naver.com, dqkim@hanyang.ac.kr, jssim@inha.ac.kr

Finding the longest common non-superstring based on prefix graph model

Siwon Choi[O1] Dokyoung Lee[2] Dong Kyue Kim[3] Jeong Seop Sim[1]
[1]Inha University, [2]Seoul National University, [3]Hanyang University

1. 서론

문자열 불포함 문제에 대한 연구는 최근 들어 여러 분야에서 활발히 진행되어 왔다. 서로 포함관계가 없는 금지문자열 집합 F가 주어질 때, F 내의 어떤 문자열도 포함하지 않는 문자열을 F에 대한 공통비상위문자열 즉, $CNSS_F$(Commom Non-Superstring)라 한다. 유한길이의 $CNSS_F$중에서 가장 긴 문자열을 최장공통비상위문자열 즉, $LCNSS_F$(Longest Common Non-Superstring)라 한다.

예. 알파벳 $\Sigma=\{a,b\}$에서 $F=\{aaa,aba,bba,bbb\}$일 때, $CNSS_F$ 집합은 $\{\lambda,a,aa,aab,aabb,ab,abb,b,ba,baa,baab,baabb,bab,babb,bb\}$ 이며, 이들 중 $baabb$가 $LCNSS_F$이다.

본 논문에서는 $CNSS_F$와 관련된 2가지 주제에 대한 연구 결과를 제시한다. 먼저 [1,2]의 $CNSS_F$에 대한 접미사(suffix) 그래프 모델과 달리 접두사(prefix)를 이용하여 직관적인 그래프 모델링이 가능함을 증명한다. 다음으로, 상수 크기의 알파벳에 대해 정의된 문자열 집합 F의 모든 문자열들의 길이의 합을 $\|F\|$라 할 때 $O(\|F\|)$시간에 접두사 그래프를 생성하고 이를 이용하여 $LCNSS_F$를 찾는 알고리즘을 제시한다.

문자열 포함 및 불포함 문제들에 대한 연구는 DNA 염기서열과 관련된 문제에 많은 응용이 기대 된다[3,4,5]. 특히 DNA에서 특정한 DNA 세그먼트(segment)가 존재하면 생명체에 유전 질환이 발생함이 알려졌으며 현재 이런 유전 질환을 발현시키는 DNA 세그먼트들이 많이 발견되었다. 본 논문을 이 세그먼트들을 금지문자열로 하는 DNA 분석 문제에 응용할 수 있을 것으로 기대한다.

2. $CNSS_F$ 모델링

본 논문의 모델링 아이디어는 현재 상태에서 만들어진 $CNSS_F$문자열 α에 문자 x를 연결해 보다 긴 $CNSS_F$를 생성하는 것이다. 이때 α의 접미사를 이용해 αx의 $CNSS_F$ 여부를 확인할 수 있다. 문자열 α의 접미사중 F의 어떤 진접두사(proper prefix)와 일치하는 문자열을 s라고 하자. 모든 s에 대해 sx가 어떤 금지문자열과 일치한다면 αx는 $CNSS_F$가 아니며, 그렇지 않으면 αx는 $CNSS_F$이다.

F의 각 모든 금지문자열의 진접두사 집합을 P라 하자. P는 공백문자열 $p_0=\lambda$를 포함한다. 본 논문에서는 $F\cup P$에 대응되는 정점 집합 V를 갖는 유향그래프 $G_F(V,E)$를 이용하여 $CNSS_F$를 모델링한다. F에 대응되는 정점은 모두 특별한 정점 \$로 하고, P의 각 문자열 p_j는 각 정점 v_j와 유일하게 대응된다. $F\cup P$의 문자열 s에 대응되는 정점을 $ver(s)$라 표기하고 p_j에 대해 $p_j=str(v_j)$로 표기한다. V에서 정점 \$를 삭제한 정점 집합을 $V'(=V-\{\$\})$이라 하자. G_F에서 \$를 삭제한 그래프를 $G_F'(V',E')$라 한다. $LS(A)$는 A의 접미사 중 $F\cup P$에 속하는 가장 긴 문자열을 나타낸다. $ver(LS(A))$를 편의상 $LSV(A)$로 축약하여 사용하기로 한다. 하나의 간선 $<v_i,x,v_j>\in E$는 두 정점 v_i,v_j와 문자 x의 쌍으

로 구성되며, 다음과 같이 v_i와 x에 따라 v_j가 정의된다.

1. $v_i = \$$인 경우 : $v_j = \$$ 이다.
2. $v_i \neq \$$이고 $p_i x \in F \cup P$인 경우 : $v_j = ver(p_i x)$ 이다.
3. $v_i \neq \$$이고 $p_i x \notin F \cup P$인 경우 : $v_j = LSV(p_i x)$ 이다.

별도로 간선은 함수 $\delta(v_i, x) = v_j$로 나타낸다. $\delta^*(v,\alpha)$는 정점 v에서 α의 각 문자에 δ함수를 순서대로 적용한 함수이다. LS함수는 문자열 α에 대해 $LS(\alpha x) = LS(LS(\alpha)x)$의 관계가 성립한다. 이를 이용해 G_F'의 정점 $v_0(= ver(p_0))$에서 출발하는 경로가 나타내는 문자열과 $CNSS_F$가 일대일로 대응됨을 증명할 수 있다. 즉, $\alpha \in CNSS_F \Leftrightarrow \delta^*(v_0, \alpha) \in V'$이다.

3. 알고리즘

본 알고리즘은 3단계로 구성된다. 먼저 F의 접두사를 이용하여 G_F를 생성하고, 다음으로 G_F에서 \$를 삭제한 그래프 G_F'을 생성한다. 마지막으로 G_F'에서 최장경로를 찾아 $LCNSS_F$문제를 해결한다.

G_F의 정점은 $F \cup P$에 해당하므로 $O(\|F\|)$의 시간에 쉽게 생성할 수 있다. 간선은 경우 1과 2에서 F의 접두사를 생성하며 쉽게 생성할 수 있다. 경우 3의 간선은 다음의 SV, PV, LC 함수들을 통해 생성한다. 함수 $SV(v)$와 $PV(v)$는 각각 $str(v)$의 진접미사(진접두사) 중 P에 속하는 가장 긴 문자열에 대응되는 정점을 구한다. 함수 $LC(v)$는 $str(v)$의 마지막 문자를 나타낸다. SV는 $SV(v) = \delta(SV(PV(v)), LC(v))$로 구해지며, 경우 3의 간선은 $\delta(v, x) = \delta(SV(v), x)$로 얻어진다.

G_F를 생성하는 알고리즘은 다음과 같다. 각 $f_i \in F (1 \le i \le n)$ 의 길이 t인 접두사에 대응되는 정점들을 원소로 하는 집합 $S[t]$라 하자. 즉, $S[t] = \{v \mid v = ver(f_i[1..t]), \text{단 } t \le |f_i| \text{ 이고 } 1 \le i \le n\}$이다. 더불어 $V[t] = S[0] \cup S[1] \cdots \cup S[t]$라 한다. 본 알고리즘은 $t = 0, 1, \cdots, \lceil F \rceil$ 순서로 $V[t]$를 생성해 나가며 최종적으로 생성된 $V[\lceil F \rceil]$로 G_F의 정점집합 V를 얻는다. 앞서 언급한 대로 간선 집합 E는 δ함수로 구할 수 있다. 각 정점에 대해서 $|\Sigma|$ 만큼 연산을 수행하므로 총 $O(\|F\| \times |\Sigma|)$의 수행시간을 갖는다.

마지막으로 $LCNSS_F$를 찾는 알고리즘은 G_F'을 생성하고 사이클이 없다면 최장경로를 찾아 $LCNSS_F$를 구한다. 사이클 검사는 잘 알려진 방법으로 DFS를 통해 검사할 수 있으며, DFS 또는 BFS를 이용해 $O(|V|+|E|)$시간에 최장경로를 찾고 최장경로에 대응되는 문자열을 출력하여 $LCNSS_F$를 구한다. 따라서 이들은 모두 총 $O(\|F\| \times |\Sigma|)$의 수행시간이 소요되며, 일반적으로 상수 크기의 알파벳을 가정하면 $O(\|F\|)$의 수행시간을 갖는다.

참고문헌

[1] A. R. Rubinov and V. G. Timkovsky, String noninclusion optimization problems, *SIAM Journal on Discrete Mathematics* 11, 3, 456-467, 1998.

[2] T. Jiang and V. G. Timkovsky, Shortest consistent superstrings computable in polynomial time, *Theoretical Computer Science* 143, 1, 113-122, 1995.

[3] R. Drmanac and C. Crkvenjakov, Sequencing by hybridization (SBH) with oligonucleotide probes as an integral approach for the analysis of complex genomes, *International Journal of Genome Research* 1, 59-79, 1992.

[4] T. Jiang and M. Li, DNA Sequencing and String Learning, *Mathematical Systems Theory* 29, 4, 387-405, 1996.

[5] M. Li. Towards a DNA sequencing theory. *31st IEEE Symposium on* Foundations of Computer Science, 125-134, 1990.