부분표절구간 검출을 위한 질의문서의 분할 및 탐색 기법

옥창석, 서종규, 조환규
부산대학교 컴퓨터공학과
e-mail:{csock, maniasjk, hgcho}@pusan.ac.kr

A Fragmentation and Search Method of Query Document for Partially Plagiarized Section Detection

Chang-Seok Ock, Jong-Kyu Seo, Hwan-Gue Cho
Dept of Computer Science and Engineering, Pusan National University

요 약

표절과 관련된 이슈가 주목받고 있는 상황에서 표절을 검출하는 방법에 대한 연구가 활발히 진행되고 있다. 일반적으로 표절구간 검출을 위해 복잡한 자연어처리와 같은 의미론적 접근방법이 아닌 비교적 단순한 어휘기반의 문자열 처리 방법을 사용한다. 대표적인 방법으로는 지문법 (Fingerprinting)과 서열정렬 (Sequence alignment) 등이 있다. 하지만 이 방법들을 이용하여 대용량 문서에 대한 표절검사를 수행하기에는 시공간적 복잡도의 문제가 발생한다. 본 논문에서는 이러한 단점을 극복하기 위해 NGS (Next Generation Sequencing)에서 사용하는 BWT (Burrows-Wheeler Transform)[1]를 이용한 탐색방법을 응용한다. 또한 부분표절구간을 검출하고 정확도를 향상시키기 위해 질의문서를 분할하여 작은 조각으로 만든 뒤, 조각들에 대한 질의탐색을 수행한다. 본 논문에서는 질의문서를 분할하는 두 가지 방법을 소개한다. 두 가지 방법은 k-mer analysis를 이용한 방법과 random-split analysis를 이용한 방법으로, 각 방법의 장단점을 실험을 통해 분석하고 실제 부분표절구간의 검출 정확도를 측정하였다.

1. 서론

최근 언론에서 유명 인사들의 논문표절과 관련된 논란이 많이 대두되고 있다. 그 중에는 실제 표절을 한 사람들도 있지만 오해에서 비롯된 의혹을 받는 사람들도 있다. 이렇게 연구윤리와 관련된 사항에 대해 표절여부를 얼마나 정확하게 판정하느냐가 중요한 이슈가 되고 있고 많은 연구가 진행되고 있다. 특히 그 결과가 미치는 영향이 아주 크기 때문에 표절여부 판정의 신뢰도는 아주 중요하다. 그러나 수많은 논문들을 하나씩 살펴보면서 표절여부를 판정하는 것은 상당한 시간과 비용이 필요하다. 따라서 기계적인 도움을 통해 논문들을 한데 모은 논문뭉치에 대해 표절로 의심되는 논문을 검색하여 유사한 구간을 검출하는 일차적인 과정을 거친 후 검출된 구간을 포함하는 논문들에 대해 사람이 직접 표절여부를 판정하는 것이 효율적이다. 하지만 이 과정에서 실제 표절 문서들은 대상 문서를 그대로 가져오는 것이 아니라 부분적으로 가져와서 변형을 시키기 때문에 검출의 정확도 향상을 위해 완전일치 문장 검색뿐만 아니라 부분일치 문장도 검출해야 한다. 또한 논문뭉치를 만드는 과정에서 각 논문에 대한 저작권 침해를 방지하기 위해 우리는 논문뭉치의 초성을 추출하는 방법[2]을 사용하고, 대용량 문서의 처리를 위해 BWT (Burrows-Wheeler Transform)와 FM-Index를 사용한다.

부분일치 문장을 검출하기 위해서는 질의문서를 분할하여 짧은 문자열의 집합으로 만드는 과정이 필요하다. 본 논문에서는 부분표절구간의 검출을 위한 질의문서의 분할 방법 두 가지를 소개하고, 각각의 방법의 장단점과 검출 정확도를 비교 분석한다. 첫 번째 방법은 n-gram으로 잘 알려진 k-mer analysis를 이용한 방법이고, 두 번째 방법은 random-split analysis로 NGS (Next Generation Sequencing)에서 사용하는 DNA의 short read alignment 기법을 응용한 방법이다.

본 논문의 2장에서는 표절검출과 관련된 연구들을 소개하고, 3장에서는 대용량의 논문뭉치 생성 및 색인에 대해 소개한다. 4장에서는 질의문서의 분할방법에 대해 자세하게 소개하고 5장에서는 실험을 통해 도출된 결과를 분석한다. 마지막으로 6장에서는 결론을 내린다.

2. 관련연구

표절을 검출하기 위한 연구들 대부분은 문자열 처리에 기반을 두고 있다. 일반적으로 많이 쓰는 방법은 지문법 (Fingerprinting)과 서열정렬 (Sequence alignment)이 있다. 지문법은 두 문서에서 출현하는 글자들의 빈도를 계산하는 통계적인 방법을 사용하고, 서열정렬은 두 문자열에 Gap을 허용하여 유사한 구간을 나란히 정렬하는 방법으로 Local alignment[3]가 대표적이다. 두 가지 방법은 모두 각각의 특징을 가지고 있지만 대용량의 문서에 대해서

지문법의 경우에는 정확도의 하락과 서열정렬의 경우에는 시공간적 복잡도가 높아진다는 단점이 존재한다. 이러한 단점을 극복하기 위해 BWT (Burrows - Wheeler Transform)라는 블록정렬 알고리즘을 이용하여 원본 문서와 동일용량의 정렬된 문자열을 생성한다[1]. 또한 BWT를 통해 생성된 문자열에서 질의탐색을 위해 색인을 생성하는데 FM-Index라는 BWT문자열 내부 탐색을 위한 자료구조 및 알고리즘을 사용한다[4].

앞서 설명한 문자열처리에 관련된 연구들 뿐 아니라 표절을 검출하는 방법들도 많이 연구되고 있다. C. Lyon이 제안한 방법은 지문법을 통해 일차원적인 유사도판정을 거친 후 집합이론과 통계적 패턴인식 방법을 사용하여 표절을 검출한다[5]. 그리고 M. Joy가 제안한 방법은 Warwick approach로 불리는 방법으로, 표절검출을 위해 점진적 비교방법을 사용한다[6]. 또한 이와 같은 맥락을 확장하여 G. Whale이 제안한 방법은 프로그래밍 소스코드의 표절검출을 위해서 소스코드의 attribute의 개수와 전체적인 구조의 유사도를 이용하여 표절을 판정한다[7].

표절검출을 위한 문자열 처리 및 실제 표절검출과 관련된 연구들을 정리하면 표 1과 같다.

<표 1> 문자열처리 및 표절검출과 관련된 연구들.

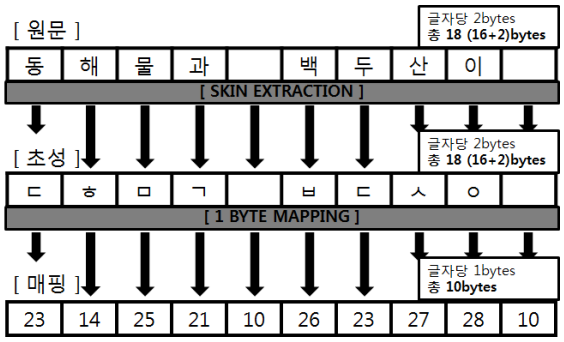
분야	저자	방법
문자열 처리	M. Burrows[1]	Block-sorting
	김성환 외[2]	원문보호 초성추출
	P. Ferragina[4]	FM-Index
	T. Smith[3]	Local Alignment
표절 검출	C. Lyon[5]	Fingerprinting 등
	M. Joy[6]	Warwick approach
	G. Whale[7]	Attribute counts 등

3. 대용량 문서의 전처리

3.1 원문 보호 및 압축을 위한 초성추출

수많은 논문들을 하나씩 비교하며 표절여부를 판정하는 것은 상당한 시간과 인력이 필요한 일이다. 따라서 우리는 수많은 논문들을 하나의 말뭉치로 만들어 표절탐색의 효율을 향상시키는 방법을 사용한다. 이렇게 생성된 대용량 말뭉치 문서에서의 질의탐색을 위해서는 전처리과정이 필요한데, 먼저 원문들의 저작권보호를 위해 한글의 초성을 추출한 후, 이를 특정 숫자에 매핑시켜 관리하고, 이 문서를 초성문서라고 한다. 특정 숫자에 매핑시키는 이유는 초성으로 사용할 수 있는 자음의 개수가 총 19개로 적기 때문이다. 여기에 쌍자음을 단자음으로 변환하게 되면 자음의 수는 14개가 된다. 또한 한글 뿐 아니라 영어, 공백문자 등을 포함하더라도 1byte로 표현가능한 수인 256개 이내이다. 이는 최소 2bytes로 표현되던 한글 한 글자가 1byte로 매핑되기 때문에 용량압축의 효과도 얻을 수 있다.

초성추출을 이용하면 원문의 보호가 가능한 이유는 한번 추출된 초성을 이용하여 다시 원문을 복원하는 것은 그 경우의 수가 많기 때문이다[2]. 원문의 보호기능 뿐 아니라 한글의 특성상 초성은 모든 글자에 존재하고 중성, 종성에 비해 가지고 있는 정보의 양이 많다는 장점을 이용하여 질의탐색의 정확도를 향상시킬 수 있다. "동해물과 백두산이"에 대한 초성추출 및 매핑의 예가 그림 1에 나타나있다.

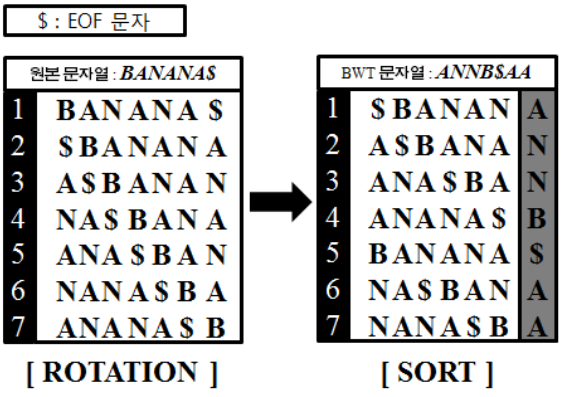


(그림 1) 초성추출 및 1byte 매핑의 예.

그림 1을 보면 원문 "동해물과 백두산이 "에 대해 초성을 추출한 결과 "ㄷㅎㅁㄱ ㅂㄷㅅㅇ "가 되고, 원문과 초성 모두 한글이 2bytes, 공백이 1byte일 때, 18bytes를 차지한다는 것을 알 수 있다. 하지만 1byte 매핑을 수행한 후에 각각의 초성들은 특정 숫자에 매핑되며 용량 또한 10bytes로 줄어든다는 것을 알 수 있다.

3.2 질의 탐색을 위한 색인 생성

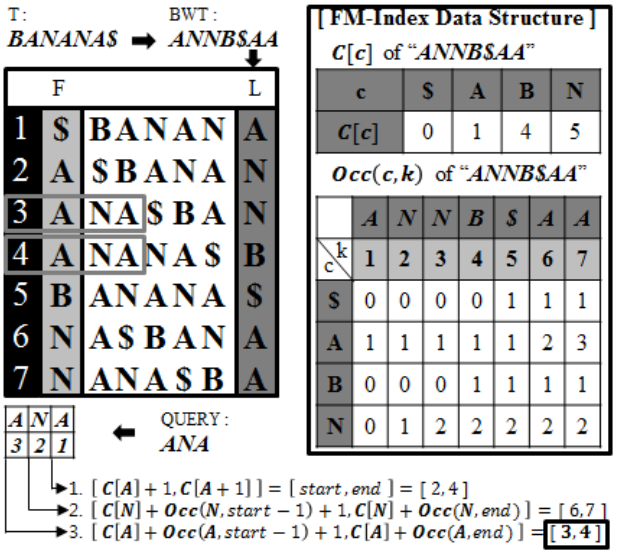
표절구간의 검출을 위해서는 초성추출이 완료된 초성 말뭉치를 이용하여 질의탐색을 수행해야 한다. 질의탐색을 위해서는 초성말뭉치의 색인을 생성해야 한다. 말뭉치의 용량이 크기 때문에 NGS (Next Generation Sequencing)에서 유전자 alignment를 위해 사용하는 기법인 BWT (Burrows-Wheeler Transform)과 FM-Index를 이용하여 색인을 생성한다. 먼저, 초성말뭉치를 이용하여 BWT를 수행하고 그 결과인 BWT문자열을 저장한다. BWT문자열을 생성하는 방법은 원본 문자열의 모든 회전행렬을 만들고, 각 행을 알파벳순으로 정렬한 후 마지막 열을 추출하면 된다. BWT문자열을 생성하는 방법이 그림 2에 나타나있다.



(그림 2) "BANANA$"에 대한 BWT의 예.

다음으로 FM-Index를 이용하여 탐색을 수행하기 위해 FM-Index 자료구조를 생성한다. FM-Index 자료구조는

두 가지로, BWT문자열을 이용하여 각 알파벳(c)별 최초 출현할 때까지의 글자 수를 나타내는 배열 $C[c]$와 BWT 문자열 내 특정 위치 k까지의 누적 출현 횟수를 나타내는 함수 $Occ(c,k)$가 있다. 두 가지 자료구조를 이용하면 BWT문자열만을 이용하여 초고속 질의탐색이 가능하다[4]. FM-Index 자료구조 생성 및 탐색의 예가 그림 3에 나타나있다.



(그림 3) FM-Index 자료구조의 생성 및 탐색의 예.

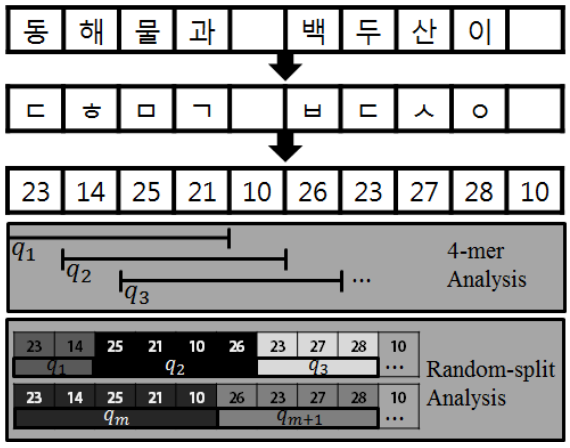
그림 3은 질의 BWT문자열 "ANNB$AA"에 대해 질의 "ANA"를 탐색하는 과정을 나타낸 것이다. 최종적으로 도출된 폐구간 [3, 4]에 대해 구간내의 모든 수 $R=\{3,4\}$를 이용하여 역추적을 수행해야 실제 텍스트 T에서의 위치를 찾을 수 있다. 역추적은 구간내의 숫자부터 시작하여 EOF문자인 '$'를 만날 때 까지 반복하고 이 때, 반복한 횟수가 실제 T에서의 "ANA"가 나타나는 위치가 된다. 구간내의 값 3에 대해 역추적을 수행한 예가 아래에 나타나 있다.

1. $L[3]=N,\ C[N]+Occ(N,3)=5+2=7$
2. $L[7]=A,\ C[A]+Occ(A,7)=1+3=4$
3. $L[4]=B,\ C[B]+Occ(B,4)=4+1=5$
4. $L[5]=\$$

총 4번만에 EOF문자인 '$'에 도달하였으므로 T의 4번째 글자부터 "ANA"가 나타난다. 마찬가지로 구간내의 값 4에 대해서도 위의 3, 4번 과정만 거치기 때문에 총 2번만에 '$'에 도달하고 이는 T의 2번째 글자부터 "ANA"가 나타난다는 것을 의미한다.

4. 질의문서의 분할

k-mer analysis는 정보검색분야에서 널리 쓰이는 n-gram과 같은 개념으로 특정 문자열의 처음위치부터 k개씩 순차적으로 자르는 방법이다. 이 방법의 장점은 한글자 단위로 자르기 때문에 부분표절의 검출에 있어 탐색의 정확도가 높고, 문자의 중복으로 인해 세밀한 탐색이 가능하다는 것이다. 하지만 높은 중복도 때문에 조각의 수가 많이 생성되며 이로 인해 전체 조각에 대한 탐색속도가 느려진다는 단점이 있다. Random-split analysis는 맵핑된 초성문서를 복사하여 각 복사본에 대해 임의의 길이로 자르는 방법이다[8]. 이 방법의 장점은 조각의 수가 k-mer에 비해 적어 탐색시간이 빠르다는 것이고, 단점은 조각의 수가 적은만큼 탐색의 정확도가 낮다는 점이다. 따라서 실제 부분표절구간의 검색을 위해서 상황에 따라 두 방법 중 알맞은 방법을 선택하여 사용하면 된다. k-mer analysis와 random-split analysis를 이용한 질의문서 분할의 예가 그림 4에 나타나있다.



(그림 4) 두 가지 질의문서 분할방법의 예.

그림 4에서와 같이 "동해물과 백두산이 "에 대해 4-mer analysis를 적용하여 질의문서를 분할하면 그 결과로 $q_1=<23,14,25,21>$, $q_2=<14,25,21,10>$등이 생성된다. 마찬가지로 random-split analysis를 적용하여 복사본 2개를 만든 후 임의 분할하게 되면 그림 4의 아래와 같이 다수의 조각이 생성된다.

5. 질의문서 분할 실험 및 분석

이 장에서는 앞서 소개한 두 가지의 질의문서 분할 방법에 대한 탐색의 정확도를 실험한다. k-mer analysis의 경우에는 k값이 작을 경우 매칭되는 경우의 수가 많아져 탐색의 정확도가 낮아진다. 따라서 초성추출을 통한 문자열 탐색의 정확도를 높이기 위해 k의 값은 15로 고정하고, random-split의 경우에는 k-mer analaysis와 유사한 환경을 만들기 위해 복사본의 개수는 3개, 자르는 길이의 범위는 [13, 17]로 고정하여 해당 구간 내의 값 중 임의로 선택하여 자르도록 한다. 실험에 사용한 입력파일 중 말뭉치의 경우에는 세종 말뭉치[9]와 직접 수집한 문서들을 모아 약 1GB정도의 용량을 사용한다. 이 말뭉치 내에는 질의탐색을 위한 문서도 포함되어 있으며, 탐색 실험 시에 말뭉치에 포함시킨 문서를 이용하여 질의탐색을 수행하고 실제 해당 문서의 위치가 검출되는지를 측정하여 정확도를 계산한다.

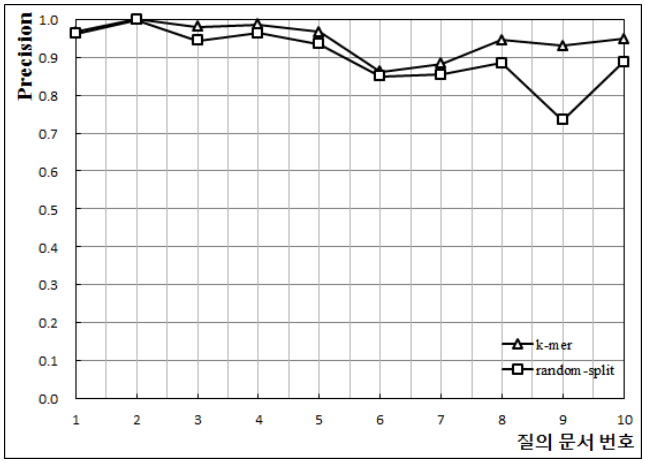
<표 2> 10개의 질의문서에 대한 k-mer analysis와 Random-split analysis의 Precision 실험 결과 표.

구분	k-mer analysis, k=15					Random-split analysis, dup.=3, range=[13,17]				
	조각 수	탐색시간(초)	tp	fp	Precision	조각 수	탐색시간(초)	tp	fp	Precision
1	7763	9.034	7508	255	0.9672	1583	4.125	1524	59	0.9627
2	7356	6.916	7354	2	0.9997	1474	3.652	1472	2	0.9986
3	8034	6.833	7880	154	0.9808	1668	3.68	1574	94	0.9436
4	9393	3.964	9268	125	0.9867	1935	3.104	1864	71	0.9633
5	8138	8.909	7870	268	0.9671	1684	4.104	1574	110	0.9347
6	5587	6.075	4812	775	0.8613	1134	3.472	964	170	0.8501
7	3161	4.731	2792	369	0.8833	658	3.189	562	96	0.8541
8	3151	4.401	2979	172	0.9454	674	3.682	596	78	0.8843
9	4714	3.848	4384	330	0.9300	1202	3.284	882	319	0.7344
10	4292	3.447	4068	224	0.9478	920	2.976	816	104	0.8870

정확도를 계산하기 위해서 Precision과 Recall을 사용한다. Precision과 Recall은 아래 수식 1과 같이 계산된다.

$$Precision = \frac{tp}{tp+fp}\ ,\ Recall = \frac{tp}{tp+fn} \qquad (1)$$

수식 1에서 tp는 true positive로 질의문서의 탐색결과 중 실제 문서가 존재하는 위치에서 발견되는 결과의 수를 의미한다. fp는 false positive로 실제 문서가 존재하는 위치에서 발견되지 않은 결과의 수를 의미한다. 마지막으로 fn은 false negative로 실제 문서에 없는 것을 탐색결과에서 없다고 한 것인데, 본 실험의 특성상 0으로 간주한다. 따라서 본 실험의 특성에 의해 말뭉치에 존재하는 모든 질의 조각은 무조건 검색이 되기 때문에 Recall은 tp/tp가 되어 항상 1이다. 즉, precision을 이용하여 두 가지 분할방법의 정확도를 계산한 결과는 표 2와 그림 5에 나타나 있다.



(그림 5) k-mer와 random-split 방법의 precision 비교 그래프.

6. 결론 및 추후 연구

본 논문에서 최근 이슈가 되고 있는 표절논란과 관련하여 표절검출방법의 일환으로 부분적인 표절구간을 검출하는 방법에 대해 소개하였다. 부분적인 표절구간을 검출하기 위해서는 질의문서에 대한 분할이 필요한데, 본 논문에서 두 가지의 분할방법을 다루었다. 그 결과 k-mer를 이용한 방법의 Precision이 대체로 더 높았지만 조각 수가 많아 탐색시간이 오래 걸린다는 것을 확인할 수 있었다, 이로써 상황에 따라 정확도와 탐색 시간의 중요도에 맞춰 두 방법 중 하나를 선택하거나 두 방법을 융합하는 방법도 고려해 볼 수 있다. 앞으로의 연구를 통해 소개한 방법 외에 다양한 분할방법을 연구할 것이며, 소개한 기법들에 대해서도 좀 더 심층적인 분석을 통해 최적화 할 것이다.

감사의 글

이 논문은 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2012-0005518)

참고문헌

[1] M. Burrows, D. J. Wheeler, “A block-sorting lossless data compression algorithm,” Technical report, 1994.

[2] 김성환, 박선영, 조환규, “한글 초성을 이용한 원문보호 탐색기법,” 제35회 한국정보처리학회 춘계학술발표대회 논문집 vol.18, pp.411-414, 2011.

[3] T. Smith, M. Waterman, “Identification of common molecular subsequences,” Journal of Molecular Biology, vol.147, pp.195-197, 1981.

[4] P. Ferragina, G. Manzini, “Opportunistic data structures with applications,” In Proceedings of the 41st Annual Symposium on Foundations of Computer Science, FOCS’00, pp.390-, 2000.

[5] C. Lyon, J. Malcolm, B. Dickerson, “Detecting short passages of similar text in large document collections,” In Proceedings of the 2001 conference on empirical methods in natural language processing, pp.118-125, 2001.

[6] M. Joy, M. Luck, “Plagiarism in programming assignments,” Technical report, Coventry, 1998.

[7] G. Whale, “Identification of program similarity in large populations,” Comput. J., vol.33(2), pp.140-146, 1990.

[8] 박선영, “버로우즈-휠러 변환과 다단계 정렬을 이용한 초고속 한글 문서 탐색,” 제31회 한국정보과학회 학생논문경진대회 입상논문, pp.545-558, 2012.

[9] 21세기 세종계획, http://www.sejong.or.kr/