다단계정렬을 활용한 효율적인 문서 유사도 비교법

서종규, 황혜련, 조환규
부산대학교 컴퓨터공학과

e-mail : maniasjk, haelyen, hgcho@pusan.ac.kr

An effective method for comparing similarity of document with Multi-Level alignment

Jong-Kyu Seo, Hae-Lyen Hwang, Hwan-Gue Cho
Dept. of Computer Engineering, Pusan National University

요 약

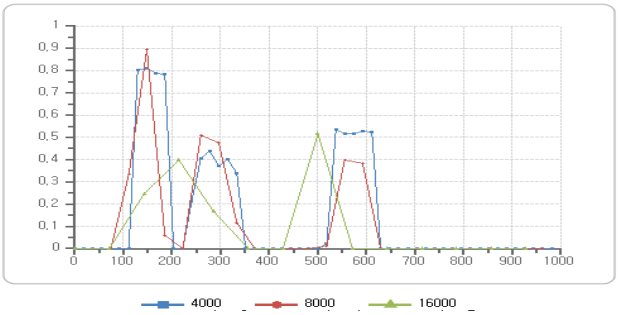
문서와 문서간의 유사도를 측정하는 방법은 크게 지문법(fingerprint)을 이용한 방법과 서열 정렬(sequence alignment)알고리즘을 이용한 방법이 있다. 두 방법은 각각 속도와 정확도라는 장점을 가지고 있다. 다단계정렬(MLA, Multi-Level alignment)는 이러한 두 방법을 조합하여 탐색 속도와 정확도 사이의 비중을 사용자가 결정할 수 있도록 하기 위한 방법이다.[1] 다단계 정렬은 두 문서를 단위 블록(basis block)로 나누고 블록 간의 벡터를 비교하여 유사도를 측정하게 되는데, 본 연구에서는 초성 추출 및 어간 추출을 통해 단위 블록의 벡터를 빠른 시간에 생성하고 비교하는 방법과 다단계 탐색을 통해 정확도를 유지하면서 빠르게 유사도를 측정하는 방식에 대해 설명한다. 실험결과 제안 방법을 통해 다단계 정렬 방법을 이용한 대용량 문서 비교의 속도가 2 배이상 빨라짐을 보인다.

1. 서론

문서와 문서간의 유사도를 측정하는 방법은 크게 지문법(fingerprint)을 이용한 방법과 서열정렬(sequence alignment)알고리즘을 이용한 방법이 있다. 지문법을 이용한 방법은 문서의 크기에 큰 영향을 받지 않고 빠르게 유사도를 측정할 수 있는 반면, 정확도가 떨어지고 부분적인 유사도가 나타나지 않는다. 서열 정렬 알고리즘을 이용한 방법은 시간 복잡도가 높은 대신 짧은 길이의 문자열에 대해 매우 정확한 유사도를 찾을 수 있다. 다단계정렬(MLA, Multi-Level alignment)는 이러한 두 방법을 조합하여 탐색 속도와 정확도 사이의 비중을 사용자가 결정할 수 있도록 하기 위한 방법이다. 다단계정렬은 두 문서를 단위 블록(basis block)으로 나누고, 각 블록 사이의 유사도를 지문법으로 측정한 다음, 측정된 블록 유사도 정보를 기반으로 서열정렬을 수행하여 연속된 유사 블록을 찾아낸다. 이때 단위 블록(basis block)의 크기가 탐색속도와 정확도를 결정하게 된다. 특히 단위 블록의 생성 및 유사도 비교가 다단계 정렬에서 가장 많은 시간을 소비하게 되는데, 본 연구에서 한글 초성 및 어간 추출을 통해 단위 블록의 벡터를 빠르게 생성하고 비교하는 방법을 제안한다. 또한 다단계 탐색 방식의 이용한 효율적인 문서 비교 방법을 제안한다

2. 관련연구

다단계 정렬(Multi-Level Alignment, MLA)이란, 두 문서 간의 유사도를 탐색하는 방법의 하나이다. [1] 입력은 두 개의 문서 D_x와 D_y이며, 출력은 두 문석간의 유사한 영역 $Area_xy$와 그 유사한 정도 $Score_xy$이다. 이 방법은 기존의 문서 비교 방법인 시퀀스 정렬과 지문법을 조합하여 탐색속도와 정확도 사이에서 중요한 정도를 사용자가 단계적으로 선택할 수 있도록 하기 위해 고안된 방법이다. 다단계 정렬에서는 우선 문서를 단위 블록(basis block)으로 나누고, 각 블록의 벡터 값을 추출 한 다음 지문법을 이용하여 블록간의 유사도를 탐색한다. 이 블록 하나를 문자 하나로 취급하여 시퀀스 정렬을 수행하면 지문법과 시퀀스 정렬의 장점이 조합된 다단계 정렬 방법이 된다. 이러한 방법을 다단계 정렬이라 부르는 이유는 시퀀스 정렬이 수행되는 단위블록의 크기를 단계별로 나누어 사용자가 선택할 수 있기 때문인데, 가령 단위 블록의 크기가 1 글자이면 다단계 정렬은 일반적인 시퀀스 정렬과 동일하게 되며, 블록의 크기를 문서의 크기와 같게 두면 지문법과 동일하게 된다.



(그림 1) 다단계 정렬을 통한 문서 유사도 비교

그림 1 은 다단계 정렬의 예로, 블록크기가 4000, 8000, 16000 일때의 비교결과이다. 비교에 걸린 시간은 각각 4.24 초, 2.19 초, 1.05 초로 블록 크기가 커질 수록 탐색 시간은 줄어 들지만, 점점 부분 유사도의 정확도가 떨어져 유사구간이 모호해 진다.

3. 다단계 탐색방법

MLA 방법에서 가장 중요한 것은 비교의 단위가 되는 블록을 효과적으로 나누는 것이다. 이 블록들은 서로 지문법으로 비교 되는데, 지문법에서 사용할 성분에 따라 블록의 벡터가 결정 된다. 가장 간단한 방법으로 글자 하나하나를 성분으로 볼 경우, 단위 블록의 벡터는 표 1 과 같은 경우의 수를 가진다.

<표 1> 유니코드에 등록된 한글의 문자 개수

문자 범위	개수	문자범위	개수
가 ~ 낗 다 ~ 띻 바 ~ 삫 사 ~ 쌓 자 ~ 찧	1176	나 ~ 닣 라 ~ 맇 마 ~ 밓 아 ~ 잏 차 ~ 칳 카 ~ 킿 타 ~ 팋 파 ~ 핗 하 ~ 힣	588

각 블록당 11172 차원의 속성벡터가 존재하여, 이들을 지문법으로 비교하면 유사도를 계산 할 수 있다. 하지만 한글 언어의 특성상 한 자로 이루어진 글자는 큰 의미를 갖지 못하며, 적어도 형태소 또는 단어 이상의 의미를 가진 속성을 취하여야 블록의 특징이 반영 된다고 할 수 있다. 여러 글자로 이루어진 벡터를 취할 경우 벡터의 크기는 표 2 와 같이 증가한다.

<표 2> 글자 길이에 따른 벡터 의 크기의 변화

길이	1	2	3	4	5
벡터 크기	11172	1.25 $\times 10^8$	1.39 $\times 10^{12}$	1.55 $\times 10^{16}$	1.74 $\times 10^{20}$

벡터의 크기는 글자의 길이 k 에 따라 지수 승으로 늘어나므로, $k > 3$일 경우 비교에 걸리는 시간과 메모리 용량이 너무 커 현실적으로는 사용할 수 없게 된다. 따라서 벡터의 크기를 줄여야 할 필요가 있는데, 가장 간단하면서 효과적인 방법은 단어의 초성을 추출 하는 것이다. 참고문헌[2]에 따르면 한글문서에서 초성을 추출하면, 원문의 내용을 보호할 수 있으며, 초성이 가지는 정보에 의해 일정길이 이상의 질의문장에서는 유일한 검색결과를 보장 한다는 것을 알 수 있다.[3] 이 방법을 적용하면 원문을 보호하고 벡터의 크기를 줄이면서도, 블록의 특징을 효과적으로 표현할 수 있다. 참고문헌[1]에 제안된 한글 초성을 추출하는 방법은 다음과 같다.

$$skin(D) = < ic(x_1), ic(x_2), \dots ic(x_n) > \quad (1)$$

$$ic(x_i) = \begin{cases} f_i & \text{if } f_i\text{가 단자음} \\ f_i\text{의 단자음} & \text{if } f_i \text{ 가 쌍자음} \end{cases} \quad (2)$$

이때 쌍자음 ㄲ,ㄸ,ㅃ,ㅆ,ㅉ의 경우 출현 빈도가 낮아 불필요하게 메모리를 낭비하게 되므로 식(2)와 같이 각 쌍자음을 단자음으로 변환해 준다.또한 한글은 어미의 변화가 자유로운 편으로, 변화가 자유롭지 않은 어간에 의해 문장의 의미가 결정된다. 그러므로 문자열의 길이 k 를 3 정도로 제한하면 어미의 변화에 강건하면서도 원문의 의미를 유지하도록 할 수 있다. 이러한 점들을 고려하면, 최종적으로 블록에 적용할 벡터 v 의 크기는 $16 \times 17 \times 17 = 4624$ 가 된다. 각 단어를 매칭되는 벡터의 인덱스로 변환하는 공식은 아래와 같다.

$$\text{array W} = tokken(skin(D)) \quad (3)$$

$$\text{index(W[i])} = f(W[i]_0) * 17 * 17 + (f(W[i]_1) + 1) * 17 + (f(W[i]_2) + 1) \quad (4)$$

$$\text{f(x)} = \begin{cases} 0, \ if\ x\text{가 영어} \\ 1, \ if\ x\text{가 숫자} \\ 2, \ if\ x\text{가 }'ㄱ' \\ 3, \ if\ x\text{가 }'ㄴ' \\ \quad \dots \\ 15, \ if\ x\text{가 }'ㅎ' \end{cases} \quad (5)$$

위의 식 (3)는 블록전체의 문장 D의 초성을 추출한 뒤, 단어단위로 쪼개는 것을 나타내며. 식 (4)는 식(5)를 이용하여 각 단어에 연관되는 벡터의 인덱스를 계산 하는 과정을 나타낸다. 위의 식을 이용해 [EEE ~ 힣힣힣]까지 3 글자로 이루어진 모든 단어를 0~4623 의 인덱스로 빠르게 변환할 수 있으므로, 별도의 정렬과정 필요 없이 단순 비교를 통한 유사도 계산이 가능해 진다.

4. 다단계 트리 구조를 이용한 블록 유사도 측정

블록 벡터들의 유사도를 비교하기 위하여 다음과 같이 비교함수를 정의 한다.

$$sim(B_1, B_2) = \frac{1}{n}\sum_i \ diff(V_{B_1}[i], V_{B_2}[i]) \quad (6)$$

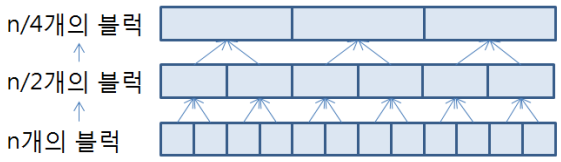
,

$$n = min\ (|V_{B_1}|, |V_{B_2}|) \quad (7)$$

$$diff(x, y) = \begin{cases} \text{x} & -(y - x)\ ,\ x < y \\ \text{y} & -(x - y)\ ,\ \text{x} \geq \text{y} \end{cases} \quad (8)$$

벡터의 유사도를 비교에는 벡터에 속한 각 요소의 *일치 횟수 - 불일치 횟수* 를 합하여 전체 횟수로 나누는 방법을 택하였다. 이 방법은 직관적이고 간단한 비교를 통해 두 벡터의 유사도에 따라 최소 -1 에서

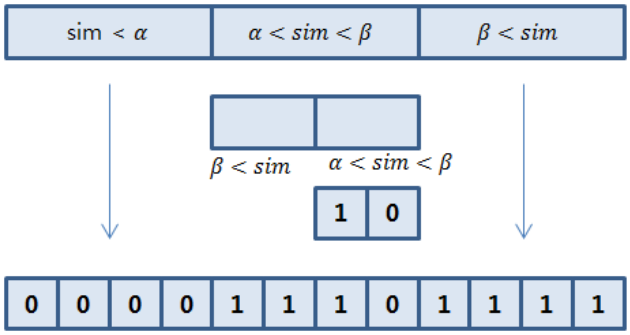
최대 1 까지의 결과를 나타낸다.다단계 정렬에서 단위 블록이 만들어지면 단위블록의 벡터를 이용하여, 아래의 그림 2 과 같이 선형 시간 $O(n)$에 상위 단계 블록 n/2 개의 벡터를 만들 수 있다.



<그림 2> 단위 블록의 벡터로부터 상위 블록의 벡터를 생성하는 과정

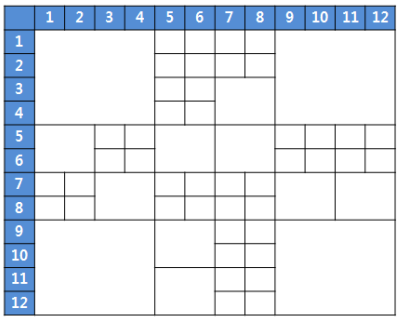
이 과정을 반복하여 최상 블록의 벡터까지 계산하게 되면, 최종적으로 $2n$개의 벡터를 만들 수 있다

이렇게 만들어진 여러 단계의 블록 벡터를 이용하면 효율적인 문서 비교가 가능하다. 아래의 그림 3 은 이러한 탐색과정을 나타낸 것이다



<그림 3> 상위레벨의 비교결과를 이용한 다단계 트리 형식의 유사도 탐색 방법

그림 3 에서 sim은 블록벡터의 계산된 유사도이며, α는 최소 임계값, β는 최대 입계값이다. 다단계 탐색 방법은 단위 블럭이 아닌 몇 단계 위의 블록에서 시작한다. 블록 벡터의 유사도가 최대 임계 값을 초과하면(그림 3 에서 $\beta < sim$) 그 블록이 포함하는 모든 단위 블록들은 완전히 일치한다고 판단하며, 반대로 블록 벡터의 유사도가 최소 임계 값 미만의 값을 가진다면(그림 3 에서 $sim < \alpha$) 그 블록이 포함 하는 모든 단위 블록들은 유사도가 0 이라고 판단한다. 두 임계 값 사이의 유사도를 가지는 블록의 경우는 (그림 3 에서 $\alpha < sim < \beta$) 하위 블록으로 분할 하여 각 블록에 대해 다시 유사도를 검사한다. 이 과정을 그림으로 나타낸 것이 그림 3 이며, 이것을 두 벡터간의 비교에 적용시키면 아래의 그림 4 와 같은 모습이 된다. 이러한 방식의 탐색은 상위 레벨에서의 블록 비교 정보를 이용하여 확실히 구분 가능한 영역을 제외시키고 하위 레벨에서의 블록 비교가 수행 되기 때문에, 상위레벨의 빠른 비교속도와 하위레벨의 정확한 부분 유사도 측정이라는 두 장점을 모두 얻을 수 있다.



<그림 4> 두 문서의 유사도 비교에 다단계 탐색을 적용한 유사도 테이블

5. 실험 및 결과

초성 추출을 통한 블록 벡터 생성 실험과 다단계 탐색을 통한 유사도 비교의 성능 실험을 위해 21 세기 세종 말뭉치[4]를 가공하여 각 용량 별로 문자 데이터를 만들어 이용하였다.

<표 3> 실험에 사용한 데이터의 크기, 단어 수 및 문자 수. 세종계획[4]으로부터 추출 하였다

#	파일크기 (KB)	단어 수	문자 수
1	1,400	223,420	1,334,869
2	1,500	231,201	1,439,751
3	7,000	1,020,117	7,175,185
4	63,000	9,223,169	65,297,029

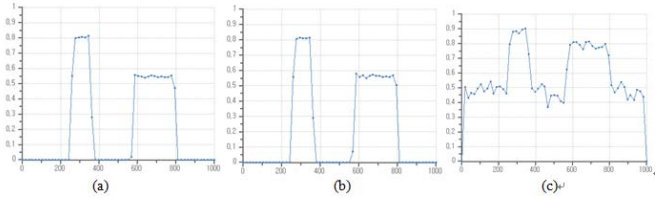
5.1 초성 추출을 통한 블록벡터 생성 성능실험

성능은 크게 속도와 정확도 부분으로 나누어 실험하였다. 먼저 표 3 의 데이터 3 번과 4 번을 사용하여 초성을 추출 했을 때와 추출 하지 않았을 때의 블록 벡터 크기 및 문서 전체의 블록 벡터 비교 시간을 측정해 보았다.

<표 5> 초성 추출 유무에 따른 벡터 크기 및 유사도 계산 속도 비교

	초성 추출 안 함		초성 추출 함	
블록 크기	벡터 크기	비교 시간 (초)	벡터 크기	비교 시간(초)
1k	891	757.3	4,624	44.55
2k	1,646	352.2		14.36
4k	3,008	165.0		3.83
8k	5,364	85.2		1.00
16k	9,450	37.9		0.25
32k	16,205	16.1		0.07

초성 추출 없이 가변크기 벡터를 사용하면, 값 비교를 통한 인덱스 검색에 $O(\log n)$의 시간이 필요하지만 초성 추출을 수행하여 고정 크기의 벡터를 사용하면 hash 방식과 비슷하게 상수 시간에 인덱스를 찾을 수 있어 유사도 비교 속도가 월등히 향상 되었다.

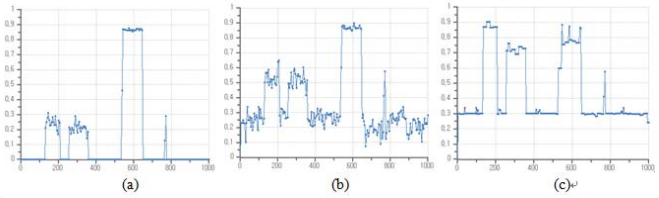


<그림 5> 그림 5 표 3 의 2 번 3 번 데이터를 4k 의 블록 크기로 나누어 유사도를 측정한 결과. (a)는 한 단어, (b)는 단어의 어간 3 글자, (c)는 초성 추출 한 단어의 어간 3 글자를 벡터로 사용했을 때의 유사도 비교 결과이다.

위의 그림 5 은 2 번 3 번 데이터를 비교한 것으로, 실제 동일한 영역은 250~380 부분과 580 ~ 800 의 부분이다. (a)와 (b)를 보면 어간 3 글자를 추출 했을 때 정확도가 약간 향상 되었지만 결과에 큰 영향을 주지는 않았다. (c)는 어간 및 초성 추출을 적용한 것으로 매우 빠른 속도로 벡터 생성 및 비교가 가능한 반면, 전체 영역에서 일정 크기의 거짓 유사도가 측정되었다. 따라서 적절한 threshold 를 지정하여 유사영역을 명확히 구분해 줄 필요가 있다.

5. 2 다단계 탐색 방법의 성능실험

다단계 탐색의 실험에는 표 3 의 1 번 3 번 데이터를 이용하였다. 1 번 데이터는 3 번 데이터와 전혀 다른 내용에 부분적으로 3 번 데이터의 문자열이 포함된 데이터이다.



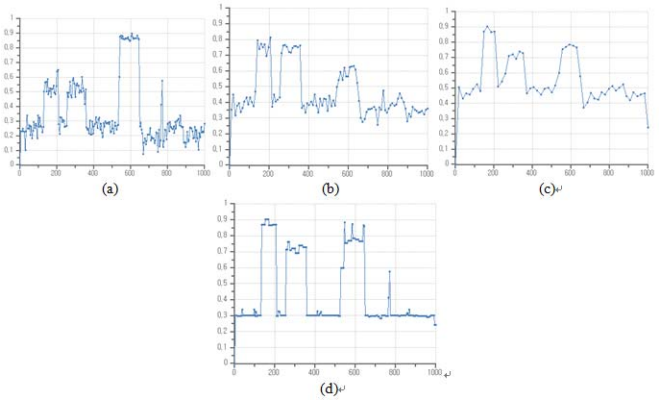
<그림 6> 표 3 의 1 번 3 번데이터를 1k 단위의 블록으로 나누어 유사도를 비교한 그림. (a)는 한 단어의 벡터를 사용한 결과이며, (b)는 초성 추출된 단어의 벡터를 사용한 그림이다. (c)는 초성 추출된 단어의 벡터를 이용하여 3 단계로 다단계 탐색(최대 임계 값 0.6, 최소 임계 값 0.3)을 수행한 결과이다.

그림 6 의 결과를 보면, 세가지 탐색 방법 모두 실제 유사영역에서 높은 유사도를 보여준다. 특히나 (c)의 다단계탐색 방법의 경우, 오히려 (a), (b)에 비하여 유사영역과 그렇지 않은 영역이 확실하게 구분이 되도록 결과가 나타난 것을 알 수 있다. 다음의 표 6 에서 수행 시간을 비교해 보면 확실히 다단계 트리 탐색의 유용성을 알 수 있다. 같은 크기의 블록을 이용하더라도 상위 블록의 정보를 참조하여 유사도를 결정하기 때문에 굉장히 빠른 시간에 유사도를 계산할 수 있다.

<표 6> 유사도 탐색 방법에 따른 수행 시간 비교

유사도 탐색 방법	수행 시간(초)
(a) 초성 추출 사용 안 함	17.722
(b) 초성 추출 사용 함	1.743
(c) 위의 (b)에 다단계 트리 탐색을 적용	0.465

아래의 그림 7 은 일반적인 탐색과 다단계 탐색의 정확도를 비교한 것이다. 그림의 a,b,c 는 각각 블록 크기 1k, 2k, 4k 일 때의 유사도 그래프이며, d 는 블록 크기 1k 에 다단계 탐색을 이용한 유사도 그래프 이다.



<그림 7> 다단계탐색방법의 정확도 비교

위의 실험 에서 블록크기가 2k, 또는 4k 일 때 가로축 800 근처의 유사지점을 특정할 수 가 없다. 하지만 다단계 탐색에서는 정확하게 해당 지점이 높은 유사도를 나타내고 있다.

6. 결론 및 향후 연구과제

본 연구에서는 다단계정렬을 위한 고정 크기의 블록벡터 생성 방법과 다단계 탐색을 통한 빠르고 정확한 유사도 비교 방법에 대하여 설명하였다. 제안한 블록 벡터 생성 방법은 초성 및 어간 추출을 통해 고정된 크기의 블록벡터를 만드는 것이며, 다단계 탐색 방법은 상위 레벨에서의 빠른 비교결과를 통해 하위 단계에서 정확하게 비교해야 할 부분을 줄여 주는 것이다. 실험을 통하여 초성 추출 및 어간 추출이 최종 비교 결과에 큰 영향을 주지 않으면서도 유사도 비교 속도가 10 배 이상 향상 되었으며, 다단계 탐색을 통해 여러 단계의 정보를 모아 정확도는 유지하면서 더욱 빠른 비교가 가능함을 보였다.

감사의 글

이 논문은 2011 년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2011-0015359)

참고문헌

[1] 박선영, 조환규, “버로우즈-휠러 변환과 다단계 정렬을 이용한 초고속 한글문서 탐색”, 부산대학교 공학석사 학위논문, 2012.
[2] 김성환, 박선영, 조환규, “한글 초성을 이용한 원문보호 탐색기법”, 한국정보처리학회 춘계학술대회 논문집, 18(1):386-389, 2011.
[3] 이재홍, 오상현, “한글 음절의 초성, 중성, 종성 단위의 발생확률, 엔트로피 및 평균상호정보량”, 전자공학회논문지, 27(9):1299-1307,1989.
[4] 21 세기 세종 계획, http://www.sejong.or.kr/, last visited on 29 Fab 2012.