

계층적 메트릭 공간(metric space) 구조의 한글 근사 단어 검색 시스템

윤태진*, 조환규*

*부산대학교 정보컴퓨터공학과
e-mail:ytj@pusan.ac.kr

Korean Approximate String Searching System by Hierarchical Metric Space Structure

Taijin Yoon*, Hwan-Gue Cho*

*Dept of Computer Science, Pusan National University

요 약

우리는 지난 연구에서 변형 비속어 필터링 시스템을 위하여 근사 문자열 검색 시스템을 적용하여 서열 정렬 횟수를 비약적으로 줄일 수 있었다. 다차원 데이터 구조를 이용한 한글 근사 검색 시스템은 기준축인 Base-Pivot의 숫자에 따라 검색 결과의 정확도를 높일 수 있으나 BP이 증가한 만큼 질의 단어의 좌표를 계산하기 위한 시간이 오래 걸린다. 소규모 데이터 검색에는 문제가 되지 않으나 60,000 단어 이상의 데이터가 수록되는 국어사전과 같은 대규모 데이터를 검색하게 될 경우 요구되는 BP의 숫자도 증가하여 많은 연산시간을 필요로 한다. 본 논문에서는 기존의 근사 단어 검색 시스템을 계층 구조화 하여 요구되는 BP 숫자를 감소 시켜 성능을 향상 시키는 방법을 제안하고자 한다. 그리고 실험을 통하여 본 아이디어의 실효성을 증명하였다. 본 아이디어는 기존의 6000개의 비속어에 대하여 약 20%정도의 성능향상을 보였다.

1. 서론

우리는 한글 자소 정렬을 이용한 한글 변형 비속어 필터링 시스템을 개발하였고 속도향상을 위하여 근사 단어 검색 알고리즘을 사용하였다[1]. 이 근사 단어 검색 알고리즘을 이용하여 비약적인 속도 향상을 이룰 수 있었으나 여전히 실시간 채팅 시스템에 적용하기에는 많은 데이터 처리량을 요구한다. 비속어 6000개의 데이터베이스에 대하여 100개의 pivot과 180개의 후보군이 출력되어 단어당 약 0.01초의 연산시간을 요구하는데 일반적인 채팅 1문장에 대하여 100번의 검사가 필요하다면 1초의 연산시간을 할애하게 된다. 1초에도 수많은 문장이 오고 갈 수 있는 온라인 게임 채팅 시스템에서 이것은 치명적으로 긴 시간이라 할 수 있다.

본 시스템에서 가장 문제시 되는 부분은 기준축인 Base-Pivot(BP)의 개수이다. BP의 개수가 증가함에 따라 더 정확한 근사 단어 검색이 가능해져서 적은 수의 후보군이 출력된다. 반면에 BP의 수가 증가할수록 BP의 질의 단어의 좌표를 계산하기 위한 각 BP과 질의 단어 간의 편집거리를 측정하는데 소요되는 시간이 증가하여 일정 수 이상이 되면 BP증가에 따른 후보군 감소 비율이 줄어들어 BP의 수 증가가 오히려 성능을 저하시키게 된다. 이것은 검색 시스템의 시간 복잡도는 데이터베이스의 데이터량에 비례한다는 것을 의미한다.

표 1 BP 수에 따른 검색 효율성 실험

BP수	일반단어 1000		비속어 6000	
	시간	평균결과수	시간	평균결과수
10	0.0169	505.2	0.0716	2145.1
20	0.0110	331.3	0.0388	1214.2
30	0.0084	236.1	0.0244	762.5
40	0.0072	179.8	0.0176	541.1
50	0.0064	141.9	0.0152	441.9
60	0.0058	105.4	0.0132	356.5
70	0.0054	77.0	0.0114	287.9
80	0.0056	70.1	0.0109	262.1
90	0.0058	56.6	0.0102	211.0
100	0.0060	48.7	0.0104	182.5

현재의 시스템 구조에서는 데이터베이스의 데이터량에 비례하여 연산시간이 증가하게 된다. 이런 비례적인 증가를 막기 위해서는 계층적인 구조의 검색을 통하여 시간 복잡도를 감소시켜줄 필요가 있다. 그러므로 본 논문에서는 계층적인 구조의 다단계 BP구조를 구성하여 근사 검색시스템의 속도를 향상시킬 수 있는 방법을 제안하고자 한다.

2. 근사 문자열 검색 관련 연구

근사 문자열 매칭문제는 이미 많은 연구가 이루어진 분야로 다양한 알고리즘이 연구되었으며 다양한 해결 방법이 검증되어 있다. 가장 일반적인 방법으로는 편집 거리인 hamming distance를 이용하여 기존의 1차원적인 정렬을 벗어나서 실제 인간의 단어 인식 방법에 가까운 형태로 tree나 다차원 공간을 이용하는 방법이 주로 사용된다[2, 3]. 그 외에도 빠른 속도의 검색이 가능한 compressed suffix array, invert index 등을 이용한 근사 문자열 검색 시스템의 연구도 이루어지고 있다[4, 5]. 본 시스템에서는 편집거리와 다차원 자료구조인 R*tree를 이용한 고속 근사 문자열 검색 시스템을 사용한다. 본 시스템에 적용되는 3가지 알고리즘 3가지 핵심 알고리즘인 단어와 단어, 단어와 데이터베이스 그리고 그 데이터베이스를 이루는 핵심 자료구조에 대해서 설명하도록 하겠다.

2.1. 근사 문자열 매칭을 위한 metric index

실제 검색 시스템에서는 단어와 단어간의 1:1관계의 유사도 계산만으로는 끝나지 않고 데이터베이스내의 수많은 단어와 입력된 검색어간의 1:다 관계에서의 유사도 검색이 필요하다. 서열 정렬 혹은 편집거리를 입력단어와 데이터베이스 내의 모든 단어와 계산할 경우 많은 연산량이 필요하여 시스템의 성능이 저하된다. 그러므로 1차적인 분류를 통해서 검사할 대상을 줄여줄 필요가 있다. 우리는 이 문제를 해결하기 위하여 "A metric index for approximate string matching"의 알고리즘을 사용한다[6].

metric space는 삼각부등식(triangle inequality)를 만족하는 black-box object의 집합과 그들간의 함수이다. 정규적으로 metric space는 (X, d) 의 쌍으로 X 는 object의 전체 집합이고 $d : X \times X \rightarrow R^+$ 은 거리 함수로 양수만 반환한다. 이 거리는 reflexivity($d(x, x) = 0$), strict postiveness($x \neq y \rightarrow d(x, y) > 0$), symmetry ($d(x, y) = d(y, x)$) 그리고 삼각부등식($d(x, y) \leq d(x, z) + d(z, y)$)을 만족한다.

X 의 부분집합 U 는 우리가 찾고자 하는 object의 집합이다. metric space의 많은 질의(query)중에 우리가 주목하는 것은 범위검색(range query): 주어진 query $q \in X$, 허용 반지름 r , U 는 q 에서 r 거리 안에 있는 원소의 집합이다. 정규적으로 질의의 출력은 $(q, r)_d = \{u \in U, d(q, u) \leq r\}$ 이다.

metric space를 색인(index)하는 방법은 크게 2가지로 나뉜다. 첫째 BP기반의 기술로 하나의 일반적인 아이디어로 구성된다. U 에서 k 개의 원소를 선택하여 $\{p_1, \dots, p_k\}$ 각 원소 $u \in U$ 를 k 차원의 점($d(u, p_1), \dots, d(u, p_k)$)으로 인식하는 것이다. 이 정보를 가지고, 삼각부등식을 이용해 어떤 원소 u 를 어떤 BP p_i 에 대해서 $|d(q, p_i) - d(u, p_i)|$

$> r$ 로 필터링할 수 있다. 그 경우 우리는 $d(u, q)$ 를 평가하지 않고도 $d(q, u) > r$ 이라는 것을 알 수 있다. 이 규칙을 이용해 필터링하지 못한 원소들은 q 와 직접적으로 비교하게 된다. 더 많은 BP를 사용할 수록 더 많은 원소가 제거 되지만 색인에 더 많은 메모리를 필요로 하게 되고 좌표 계산에 더 많은 계산량을 요구하게 된다. 효율적인 범위 검색 방법으로는 kd-tree, R-tree, X-tree등에 포함된 방법이 있다.

2.2 R*tree를 이용한 범위검색

효과적인 근사 단어 색인을 위해서는 우수한 다차원 자료구조가 필요하다. 우리는 다수의 BP와 단어와의 편집거리를 이용한 다차원 metric space를 저장할 자료 구조로 R*tree를 선택하였다[7]. R*tree는 다차원 데이터 처리에서 가장 널리 사용되는 데이터 구조이다. R*Tree는 R-tree[8]의 발전된 형태로 R-tree는 우리가 흔히 사용하는 B-tree를 다차원 정보 indexing에 맞게 개량한 것이다. 기본적인 구조는 B-tree와 같이 삽입과 삭제가 일어나더라도 균형을 유지할 수 있는 방법을 취하고 있다.

R-tree는 다차원의 space를 minimum bounding rectangles로 나눈다. 각 노드는 가변적인 수의 원소를 가진다. leaf노드 외에는 자식노드의 주소와 모든 원소를 포함하는 MBR의 정보를 가지고 있다. insertion과 deletion은 가장 가까운 MBR을 이용하여 이루어진다.

R*tree는 이 R-tree의 MBR의 빈 공간을 최소화하기 위하여 원소가 삽입 될때 MBR의 원소가 overflow를 일으킬 경우 MBR의 중심에서 가장 먼 원소를 reinsertion하여 공간효율을 높이는 방법이다. 공간효율이 높아지면 overlap이 줄어들어 자료구조의 성능이 높아지게 된다.

3. 계층적 BP 구조의 개요

우리가 개발한 시스템은 미리 선정된 BP과의 편집거리를 좌표로 가지는 다차원 자료구조에 비속어 데이터베이스의 단어를 저장시켜서 그 단어를 R*tree[2, 3]등의 range query를 이용하여 후보군을 추출, 추출된 후보군과 질의 단어를 전역 정렬을 이용한 유사도 측정 알고리즘을 이용하여 가장 높게 측정된 수치를 이용하여 질의 단어의 비속어 여부를 판정하게 된다.

즉 본 시스템의 시간 복잡도는 BP의 숫자를 k , 후보군의 평균단어 수를 n 이라 하였을때 $O(k + n)$ 이 된다. n 을 줄이기 위해서는 k 를 증가 시키면 되지만 k 의 증가에 따른 n 의 개수는 반비례 관계이므로 일정 시점 이후에는 k 의 증가가 연산 시간의 증가로 이어지게 된다.

<표 2> BP의 편집 거리에 따른 검색 효율성 실험

편집 거리	1	2	3	4	5	6
비율 (%)	0.093	0.088	7.383	36.76	33.64	13.30

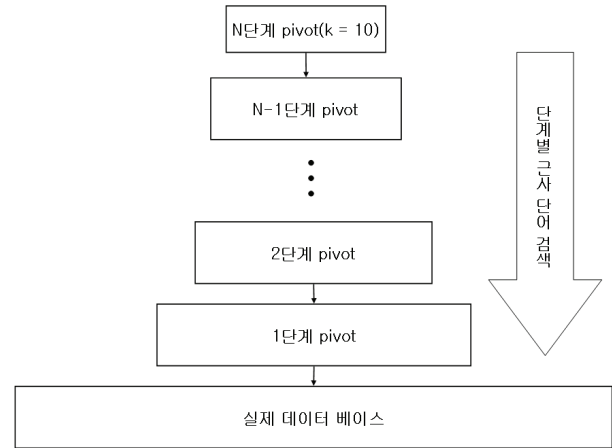
이 문제를 해결하기 위해서는 먼저 BP의 성질에 대하여 파악할 필요가 있다. BP과의 편집거리를 통한 좌표 설정의 경우 거리가 가까울 수록 분포밀도가 낮고 거리가 멀 수록 분포밀도가 높다. 즉 편집거리가 가까운 단어가 먼 단어의 수보다 적다는 것이다. 편집거리가 질의 단어의 전체 길이와 유사할 정도로 멀다면 해당 BP의 좌표는 의미가 거의 없다고 할 수 있다.

그러므로 질의 단어와 거리가 가까운 BP일 수록 근사 단어 검색의 정확도를 높이는데 큰 기여를 하게 된다. 그러나 질의 단어가 어떤 단어가 입력될지 알 수 없으므로 이에 맞춰 BP을 미리 정하는 것은 불가능하다. 이것을 해결하기 위하여 BP에도 근사 단어 검색을 이용한 후보군 추출을 적용하는 것이 본 시스템의 핵심 아이디어이다. 실제 구현에 대해서는 다음 장에서 설명하도록 하겠다.

4. 계층적 BP 구조의 실제 구현

앞서 BP 개선의 필요성과 그 핵심 아이디어에 대하여 설명하였다. 기존의 근사 단어 검색 시스템은 데이터의 양이 증가함에 따라 BP의 수와 후보군의 평균 숫자가 같이 증가하므로 갈 수록 늘어나고 발전해 나가는 비속어에 제대로 대응하기 어렵다. 그러나 계층적인 구조를 통해 BP의 실제 검사 숫자를 줄일 수 있다면 시간 복잡도는 log의 공식을 가지게 되므로 대량의 데이터베이스에 대하여 효율적인 대처가 가능해진다.

핵심 아이디어는 앞에서 설명한 바와 마찬가지로 매우 간단하다. 실제 데이터베이스를 근사 단어 검색을 통하여 후보군을 추출한 것과 마찬가지로 BP에 대하여도 근사 단어 검색을 적용하여 편집거리가 가까운 BP을 추출, 해당 BP의 좌표만을 계산하여 효율적인 range query를 만들어 내는 것이다. 또한 1단계의 추가만으로 부족할 경우 반복적인 단계 추가를 통하여 시간 복잡도를 log의 공식으로 유지할 수 있다.



(그림 1) 계층적 BP구조 - 다단계 구조를 이용하여 시간복잡도를 log의 형태로 낮출 수 있다.

여기서 문제는 BP을 몇단계로 할 것인지 하는 기준과 1단계 BP의 개수, 그리고 단계 간의 BP 개수의 비율을 어떤 기준으로 정할 것인가이다. 기존의 1단계 BP과는 달리 다단계 BP은 더 많은 숫자의 BP에서 성능이 최적화된다. 기존의 BP시스템은 1단계에서 BP의 수를 고려하면 되었기에 비교적 쉽게 BP의 숫자를 최적화 할 수 있었으나 본 시스템은 더 많은 숫자의 BP을 선정하기 때문에 다차원 자료구조를 형성하는데 걸리는 시간 역시 증가하여 BP의 수를 최적화 하기 어려워진다. 이 부분에 있어서는 좀 더 실험과 고찰을 통해 관계식을 정립할 필요가 있을 것이다.

BP 단계 간의 비율 역시 매우 중요한 문제이다. 이것은 기존의 1단계 BP 구조일때의 실험결과를 활용할 수 있을 것이다. 표 2에서 보는 바와 같이 1000개의 일반 한글 단어에 대해서는 약 70개의 BP이 최적화된 BP의 숫자이다. 반대로 1000개의 BP을 1단계 BP으로 사용하게 된다면 평균적으로 77개의 BP을 후보군으로 검출하는 형태의 2단계 BP 구조를 구성할 수 있을 것이다. 최종적으로 단계 수를 설정하는 것은 이 두 단계의 결과를 통하여 구할 수 있을 것이다.

5. 실험

본 아이디어의 실효성을 증명하기 위하여 비속어 데이터베이스를 대상으로 효율성 실험을 수행하였다. 사용된 비속어 데이터베이스는 한국 게임산업진흥원에서 발행한 보고서의 비속어[9]와 인터넷을 통하여 수집된 6000개의 비속어이다. 1단계 BP의 숫자는 100개이고 2단계 BP의 숫자는 10개이다. 대표적인 비속어 4가지에 대하여 실험을 수행 하였다. 검색 반경 3에서 측정되었다.

<표 3> 2단계 BP 시스템의 효율성 실험

질의 단어	추출된 BP수	결과 후보군 수	기존의 후보군 수	2단계 시스템 의 합	기존시 스템의 합
A	27	169	169	206	269
B	11	119	92	140	229
C	26	72	19	108	119
D	29	28	17	67	117

Acknowledge

본 논문은 연구재단(과제번호 : 2009-0070594 “진화유전학 분석기법을 이용한 지능형 인터넷 비속어/악플 필터링 시스템”)의 지원을 받아 수행한 연구를 바탕으로 작성되었습니다.

참고문헌

- [1] 윤태진, 조환규, “반 전역 정렬을 이용한 온라인 게임 변형 욕설 필터링 시스템”, 한국콘텐츠학회논문지, 제 9권, 제12호, pp.113-120, 2009.
- [2] Sreenivas Gollapudi and Rina Panigrahy. A dictionary for approximate string search and longest prefix search. In CIKM '06: Proc. of the 15th ACM international conference on Information and knowledge management, pages 768-775, 2006.
- [3] W. A. Burkhard and R. M. Keller. Some approaches to best-match file searching. Commun. ACM, 16(4):230-236, 1973.
- [4] Trinh N. D. Huynh, Wing-Kai Hon, Tak-Wah Lam, and Wing-Kin Sung. Approximate string matching using compressed suffix arrays. Theor. Comput. Sci., 352(1):240-249, 2006.
- [5] Marios Hadjieleftheriou, Nick Koudas, and Divesh Srivastava. Incremental maintenance of length normalized indexes for approximate string matching. Proc. of the 35th ACM SIGMOD international conference on Management of data, pages 429-440, 2009.
- [6] Gonzalo Navarro and Edgar Ch’avez, “A metric index for approximate string matching,” Theor. Comput.Sci., vol. 352, no. 1, pp. 266 - 279, 2006.
- [7] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger, “The R*-tree: an efficient and robust access method for points and rectangles,” Proc. of the 1990 ACM SIGMOD international conference on Management of data, New York, NY, USA, 1990, pp. 322 - 331.
- [8] Antonin Guttman, “R-trees: a dynamic index structure for spatial searching,” Readings in database systems, pp. 599 - 609, 1988.
- [9] 한국게임산업진흥원, “게임언어 건전화 지침서 연구,” 2008.
- [10] 노강호, 조환규, 박근수, “한글에 대한 Edit Distance 문제(to be published)”, 정보과학회, 2010.

표 3에서 보는 바와 같이 총 서열정렬(alignment)횟수라 할 수 있는 합이 전체적으로 줄어든 것을 볼 수 있다. 이것은 시간 복잡도가 상당부분 개선된 것을 의미한다. 특히 "개새끼"에서는 결과 후보군 수가 일치하여 이상적인 모습을 보여줬다고 할 수 있다. 그러나 나머지 결과에서는 결과 후보군 수가 증가한 것을 볼 수 있는데 이것은 아무래도 모든 BP를 사용한 것에 비하여 근사 단어 검색의 정확도가 떨어졌기 때문이다. 그러나 이것이 시스템의 성능을 저하 시 킬만큼 큰 변동을 일으키지는 않는 것을 볼 수 있다.

이 실험결과는 다단계 BP구조의 실효성을 증명하였다. 따라서 같은 데이터 량에 대하여 더 많은 BP를 선정할 수 있고 시간 복잡도와 정확도에서 성능 개선이 이루어질 수 있다는 것을 의미한다.

6. 결론

본 논문 기존의 근사 단어 검색 시스템의 시간 복잡도를 개선시키기 위한 다단계 BP구조의 아이디어와 그 구현, 성능의 실제 실험에 대하여 설명하였다. 다단계 BP구조는 기존의 $O(k+n)$ 에서 $O(\log(k + n))$ 의 형태로 시간 복잡도를 개선시켰으며 실험을 통하여 실제 성능향상을 증명하였다.

그러나 아직 각 단계별 BP의 비율이나 데이터베이스의 데이터양에 따른 1단계 BP숫자를 정하기 위하여 추가적인 실험을 통해 관계식을 정립할 필요가 있다. 현재의 실험에 의존한 BP 숫자결정 방법은 여러 번의 실험이 필요하여 효율성이 낮기 때문이다. 그리고 최종적으로 선택되는 BP의 숫자가 증가하였기 때문에 데이터베이스를 구성하는데 더 많은 시간이 필요해 진 것도 큰 문제이다. 사전 같은 데이터 변동 빈도가 낮은 데이터베이스에는 적합한 방식이나 상호, 웹검색 등 수시로 데이터가 변하는 시스템에서는 다단계 BP 구조 채택을 통한 데이터베이스 구성 시간의 증가가 큰 영향을 미칠 수 있기 때문이다. 이 문제를 해결하기 위한 추가적인 연구가 필요할 것이다. 그리고 한글 Edit Distance 방식에 대한 다양한 접근을 통해 효율적인 방법론의 정립이 필요할 것이다[10].