근사 단어 검색 효율성 개선을 위한 기준 Pivot 선택방법 실험적 연구

윤태진○ 정우근 조환규
부산대학교 정보컴퓨터공학부
{ytj, wkchung, hgcho}@pusan.ac.kr

An Empirical Study of Base Pivot Choosing Method for Approximate Word Searching

Taijin Yoon, WooKeun Chung, Hwan-Gue Cho
Pusan National University Dept of Computer Science & Engineering

요 약

한글 근사 단어 검색 시스템은 사용자의 오류를 포함한 검색 질의에 효과적으로 대응할 수 있는 방법이나 검색 속도가 매우 느려서 실제 사용에 큰 어려움이 있다. 일반적으로 DNA 검색에 사용하는 서열 정렬 기법을 사용할 경우 데이터 베이스의 모든 문자열과 비교가 이루어져야 하기 때문에 많은 검색 시간이 걸리게 된다. 이것을 해결하기 위해 우리는 편집거리가 metric space를 만족하는 성질을 이용한 한글 근사 단어 검색 시스템을 사용하여 실제 서열정렬을 사용하여 비교가 필요한 후보 단어를 거르게 된다. 이 한글 근사 단어 검색 시스템에서 가장 중요한 것은 기준축의 역할을 하는 Base-Pivot의 선택 방법이다. 본 논문에서는 이 Base-Pivot의 효율적인 선택방법을 실험을 통해서 분석하도록 한다.

1. 서 론

서열 정렬을 이용한 근사 문자열 검색 방법은DNA서열의 분석, 단어의 변형을 통한 금지 단어 입력 등을 막기 위하여 유용하게 사용된다 우리는 사용자의 변형 비속어 입력을 필터링 하기 위해 이 서열 정렬 알고리즘을 이용하여 데이터베이스의 단어와 입력 단어 간의 유사도를 측정하여 기준치 이상의 단어를 필터링 하는 방식의 비속어 필터링 시스템을 개발하였다[1]. 그러나 데이터베이스의 금칙어는 약 6000여개의 비속어를 사용하는데 모든 단어와 입력 단어의 유사도를 측정할 경우 방식은 너무 비효율적이므로 근사 단어 검색 시스템을 이용하여 입력단어와 유사한 단어를 추출하여 추출된 단어만 검사하는 방식을 사용한다.

이 근사 단어 검색 시스템은 편집 거리가 Metric Space를 만족하는 성질을 이용하여 구성된 다차원 자료구조의 범위 검색 방법을 사용하게 된다. 이 자료구조는 데이터베이스에서 기준 축의 역할을 위하여 추출된 단어인 Base-Pivot(BP)과 데이터베이스의 단어 사이의 편집거리가 각 차원의 좌표가 되므로 이 BP을 어떻게 구성하는가에 따라 검색의 정확성이 크게 변화된다. 예를 들어 "가"와 같은 너무 짧은 단어를 BP으로 선택할 경우 "ㄱ"과 "ㅏ"가 포함된 단어를 제외한 모든 단어가 한 구역으로 뭉치게 되므로 효율성이 떨어지고 반대로 너무 긴 단어를 BP으로 선택할 경우 모든 단어에 대해 유사한 부분이 존재하게 되어 오히려 특정 위치에 단어를 집중시켜 분포 효율을 떨어뜨리게 된다. 이러한 길이 외에도 추출된 단어의 다양성 등 여러 가지 요소가 필터링 효율을 결정하는 기준이 된다.

기존의 근사 단어 검색에서 사용하는 방법은 영문 알파벳이나 DNA 서열과 같은 한정된 알파벳에서 유용한 BP 선택 방법을 사용하고 있다. 본 논문에서는 기존의 BP 선택 방법을 분석하고 이를 한글 근사 단어 검색 시스템에 맞는 추가적인 아이디어에 대하여 설명하도록 하겠다.

2. 근사 단어 검색 시스템의 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 사

용된다[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.

실제 검색 시스템에서는 단어와 단어간의 1:1관계의 유사도 계산만으로는 끝나지 않고 데이터베이스내의 수많은 단어와 입력된 검색어간의 1:n 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 obeject의 집합과 그들간의 함수이다. 정규적으로 metric space는 (X, d)의 쌍으로 X는 object의 전체 집합이고 d : X × X → R+은 거리 함수로 양수만 반환한다. 이 거리는 reflexivity(d(x, x) = 0, strict postiveness(x ≠ y → d(x, y) > 0), symmetry (d(x, y) = d(y, x)) 그리고 삼각부등식(d(x, y) ≤ d(x, z) + d(z, y))을 만족한다.

X의 부분집합 U는 우리가 찾고자 하는 object의 집합이다. metric space의 많은 질의(query)중에 우리가 주목하는 것은 범위검색(range query): 주어진 query q ∈ X, 허용 반지름 r, U는 q에서 r거리 안에 있는 원소의 집합이다. 정규적으로 질의의 출력은 (q, r)_d = {u ∈ U, d(q, u) ≤ r} 이다.

metric space를 색인(index)하는 방법은 크게 2가지로 나뉜다. 첫째 BP기반의 기술로 하나의 일반적인 아이디어로 구성된다. U에서 k개의 원소를 선택하여 {p_1,...,p_k} 각 원소 u ∈ U를 k차원의 점(d(u, p_1),...,d(u, p_k))으로 인식하는 것이다. 이 정보를 가지고, 삼각부등식을 이용해 어떤 원소 u를 어떤 BP pi에 대해서 |d(q, pi) − d(u, pi)| > r로 필터링할 수 있다. 그 경우 우리는 d(u, q)를 평가하지 않고도 d(q, u) > r이라는 것을 알 수 있다. 이 규칙을 이용해 필터링하지 못한 원소들은 q와 직접적으로 비교하게 된다. 더 많은 BP을 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, R-tree등에 포함된 방법이 있다[7, 8].

3. 효과적인 BP 선택 방법

근사 문자열 검색에서 BP은 각 단어의 좌표를 측정하는 척도로서 기준 축의 역할을 하게 된다. 그러므로 이 BP이 다양한 형태를 가질 수 있도록 선택하는 것이 가장 좋다고 할 수 있다. 예를 들어 BP을 모두 같은 단어나 유사한 단어만으로 선택한다면 모든 차원의 좌표가 유사한 수치를 보이기 때문에 단어의 분류가 이루어지지 않아 너무 많은 검색 결과를 보이게 된다.

효과적인 단어 분류를 위해서는 다양한 패턴의 단어를 BP으로 선택하는 것이 중요하다. 이때 이 다양한 패턴의 단어를 선택하기 위한 방법을 정하는 것이 근사 문자열 검색 시스템의 효율성을 성능을 결정하게 된다. 우리는 무작위 선택방법, 최소 편집거리를 이용한 방법 그리고 최대 유사도를 이용한 방법의 세가지 방법을 제안하고 이를 분석하도록 한다.

3.1 무작위 선택 방법

무작위 선택 방법은 데이터베이스의 단어 중 무작위로 k개의 단어를 선택하여 BP으로 사용하는 방법이다. 무작위 선택 방법은 알고리즘이 단순하여 구현이 쉬운 장점이 있으며 데이터베이스 구성 시간이 짧은 장점이 있다. 무작위 함수를 이용하여 데이터베이스의BP을 선택하면 되므로 BP 선택에 필요한 시간 복잡도는 O(1)로 자주 데이터베이스를 갱신해줘야 하는 시스템에 유용하다고 할 수 있다. 그러나 무작위 선택 방법의 경우 성능을 보장할 수 없다는 것이 문제이다. 특히 적은 수의 BP을 선택할 경우 BP이 다양한 패턴을 가지지 못하고 편향적인 단어들을 선택할 가능성이 높아져 시스템의 성능을 저하시키게 된다.

3.2 최소 편집거리를 이용한 방법

무작위 BP 선택 방법의 문제점은 BP선택의 규칙성이 없으므로 일정한 성능을 보장하기 어렵다는 점이다. 그러므로 BP을 추가할 때 이미 선택된 BP들과 편집 거리를 측정하여 그 중에서 가장 낮게 측정된 편집거리가 가장 큰 단어를 추가하는 방법을 이용하게 된다면 특정 패턴의 단어를 집중해서 추출할 가능성을 줄일 수 있다. 이 방법을 이용하면 기존의 단어와 같은 단어가 선택되는 것을 배제할 수 있으며 적은 수의 BP을 선택하더라도 일정한 성능을 보장할 수 있다. 그러나 이 방법을 사용할 경우 선택된 BP과 데이터베이스의 단어 간의 편집거리를 측정하여야 하므로 BP을 선택하는데 O(k * n)의 시간 복잡도를 가지게 된다(k는 BP의 수, n은 데이터베이스의 단어 수). 그리고 BP의 편집거리가 긴 단어가 우선적으로 뽑히는 만큼 길이가 긴 단어를 위주로 BP이 선택되어 단어의 검색에 필요한 편집거리 측정 시간이 증가하게 된다.

3.3 최대 유사도를 이용한 방법

최소 편집거리를 이용한 방법은 다양한 패턴의 단어를

BP으로 선택할 수 있는 방법이나 길이가 긴 단어를 위주로 BP으로 선택되게 되며 기존의 단어를 부속 문자열로 가지고 있는 단어라 할지라도 추가적인 길이가 길 경우 선택되게 되어 효율성에 문제가 생길 수 있다. 그래서 그 문제를 해결하기 위하여 기존의 BP과 유사도를 측정하는 방법을 사용하고자 한다. 기존의 BP과 편집거리를 측정하는 것이 아니라 일치하는 문자가 나오는 숫자를 측정하여 이 숫자들 중 가장 높은 수치가 가장 낮은 단어를 BP으로 추가하는 방식이다. 이 방식은 기존의 편집거리를 측정하는 방식과 거의 유사하나 Match값을 제외한 나머지를 모두 0으로 주어 Match가 일어나는 문자의 수만을 평가한다. 이 방식을 이용할 경우 단어의 길이가 긴 단어 위주로 편향적으로 선택될 가능성을 줄여주면서 기존의 pivot이 가지지 못한 패턴의 단어를 우선적으로 추가하므로 근사 단어 검색 효율을 향상시키는데 도움을 준다.

4. 비교 분석 실험

우리는 한글 근사 단어 검색 시스템을 위한 세 가지 BP 선택 방법에 대하여 제안하였다. 이 장에서 우리는 세 가지 방법의 성능에 대해 실험적인 분석을 수행한다. 실험에 사용된 단어는 국어사전에서 추출된 1000개의 단어를 사용하였고 그에 대해 BP의 숫자를 증가시키면서 각각 1000번의 실험을 수행하였다. 시간은 하나의 단어를 검색하기까지 걸리는 시간으로 BP와 입력단어 간의 편집거리 측정 시간과 검색 결과로 나온 단어들과의 유사도 분석을 위한 서열정렬 값 측정 시간을 더한 시간의 평균이고 평균 결과 수는 각각의 검색 결과 단어 수의 평균이다.

표 1 무작위 선택방법에 대한 실험

BP수	시간(s)	평균결과수	BP수	시간(s)	평균결과수
10	0.0169	505.2	60	0.0058	105.4
20	0.0110	331.3	70	0.0054	77.0
30	0.0084	236.1	80	0.0056	70.1
40	0.0072	179.8	90	0.0058	56.6
50	0.0064	141.9	100	0.0060	48.7

무작위 선택 방법은 실험의 기준이 결과값을 보여준다. BP 숫자의 증가에 반비례하여 평균결과 수가 감소하는 것을 알 수 있다. 그러나 일정 이상 BP이 증가하면 입력단어의 좌표를 계산하기 위한 편집거리 계산량이 증가하므로 오히려 검색에 필요한 시간이 증가하는 것을 알 수 있다.

표 2 최소 편집거리를 이용한 실험

BP수	시간(s)	평균결과수	BP수	시간(s)	평균결과수
10	0.0193	547.3	60	0.0076	124.6
20	0.0140	386.4	70	0.0074	102.1
30	0.0105	273.6	80	0.0075	84.7
40	0.0091	204.9	90	0.0076	70.6
50	0.0080	151.9	100	0.0079	60.1

최소 편집거리를 이용한 실험은 오히려 전체적인 성능의 악화를 가져왔다. 이것은 편집거리의 수치가 크게 나오는 BP을 선택하는 과정에서 길이가 긴 단어가 우선하여 선택되었기 때문이다. 길이가 긴 단어는 한 단어에 너무 많은 패턴을 가지고 있어 오히려 분류 효율을 떨어뜨리게 된다. 그리고 BP의 길이가 길어진 만큼 편집거리를 측정하는 연산량 또한 증가하여 성능을 한층 악화시켰다.

표 3 최대 유사도를 이용한 실험

BP수	시간(s)	평균결과수	BP수	시간(s)	평균결과수
10	0.0169	509.3	60	0.0059	113.5
20	0.0110	323.8	70	0.0057	96.4.0
30	0.0083	229	80	0.0056	81.5
40	0.0071	173.4	90	0.0056	70.6
50	0.0064	140.4	100	0.0057	59.1

최대 유사도를 이용한 BP 선택 방법은 가장 높은 성능을 보여준다. 평균 결과 수는 무작위 선택방법에 비해서 낮은 성능을 보여 준다. 그러나 검색시간의 경우 오히려 향상된 것을 볼 수 있는데 무작위 선택 방법에 비해서 BP의 평균길이가 오히려 짧아 졌기 때문이다 이는 길이가 긴 단어 일수록 유사도가 높게 날 가능성이 높기 때문이다. 즉 길이가 짧으면서도 다양한 패턴을 가진 BP의 선택이 가능하였다.

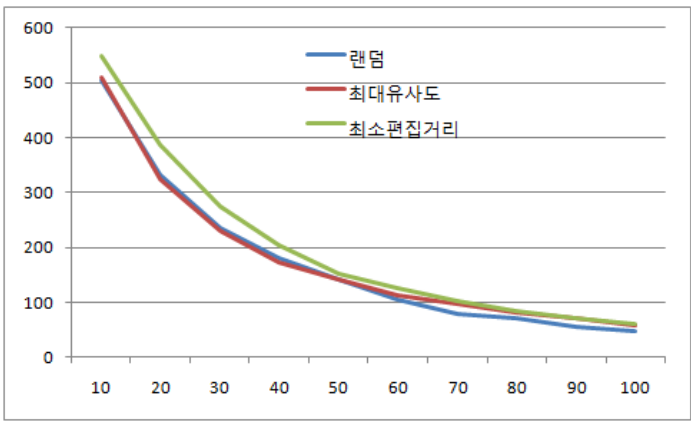

그림 1 BP의 개수에 따른 평균 결과 수

위의 그림 1은 BP의 개수에 따른 세 가지 선택 방법의 평균 결과수를 나타내는 차트이다 BP의 숫자가 적은 구간에서는 최대 유사도 방법이 가장 높은 성능을 보이고 60개 이상부터 무작위 방법이 가장 높은 성능을 보인다. 무작위 선택 방법의 경우 BP의 숫자가 적을수록 무작위로 인한 패턴의 편중이 일어날 가능성이 높아져 성능의 저하가 일어나게 된다 반면에 충분한 숫자의 BP을 선택할 경우 무작위 방법으로 BP을 선택한다 하더라도 충분히 다양한 패턴을 가질 확률이 높아지기 때문에 성능의 저하가 줄어들게 된다 최대 유사도를 이용한 방법의 경우 평균적인 BP의 길이가 짧게 선택되기 때문에 일부구간에서 평균결과 수에 있어서는 무작위 방법보다 낮은 성능을 보인다

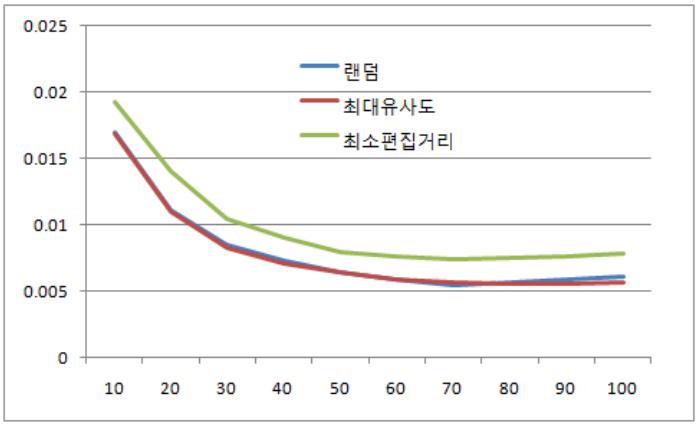

그림 2 BP의 숫자에 따른 검색시간

검색 시간의 경우 전반적으로 최대 유사도를 이용한 방법이 높은 성능을 보이는 것을 알 수 있다 평균적으로 길이가 짧은 BP을 선택하면서도 다양한 패턴을 포함하는 BP이 선택되는 것을 알 수 있다

5. 결론

우리는 변형 비속어 필터링에 사용되는 한글 근사 검색 시스템의 효율성을 높이기 위한BP 선택 방법을 제안하고 이에 대한 실험적 분석에 대하여 설명하였다BP을 선택하는 방법 중 가장 높은 성능을 보인 방법은 최대 유사도를 이용한 방법이었다 반면에 최소 편집거리를 이용한 방법은 무작위 선택 방법보다 낮은 성능을 보여 실제 필터링에 활용하기에는 무리가 있다 무작위 선택방법은 BP의 숫자가 많은 경우 안정적으로 높은 성능을 보였다. 이 방법은 BP을 구성하는데도 가장 적은 시간을 소요하게 되므로 데이터베이스를 자주 갱신하거나 대규모 데이터베이스를 구성하는데 적합한 것을 알 수 있다.

추후 연구과제로는 BP의 선택 방법에 대해 실험적 분석방법에서 더 나아가 수학적 모델을 구성하는 일이다 BP과 검색 효율에 대한 관계식을 정립하여 효율적인 데이터베이스를 구성할 수 있게 하는 것이다

6. Acknowledge

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

참고문헌

[1] 윤태진, 조환규, “반 전역 정렬을 이용한 온라인 게임 변형 욕설 필터링 시스템”, 한국콘텐츠학회논문지, 제9권, 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.