GORank: Gene Ontology를 이용한 유전자 산물의 의미적 유사성 검색

(GORank: Semantic Similarity Search for Gene Products using Gene Ontology)

김 기 성 † 유 상 원 † 김 형 주 ††
(Ki-Sung Kim) (Sang-Won Yoo) (Hyoung-Joo Kim)

요 약 유사한 생물학적 특성을 가진 유전자 산물을 검색하는 것은 생물정보학 연구에 필수적인 기술이다. 현재 대부분의 생물학 데이타베이스에서 Gene Ontology의 용어를 사용하여 유전자 산물의 생물학적 특성을 기술하고 있다. 본 논문에서는 이런 유전자 산물의 주석 정보를 사용해 의미적으로 유사한 유전자 산물을 검색하는 방법을 제안한다. 이를 위해 우선 정보 이론에 기반한 유전자 산물간의 의미적 유사도를 정의하였다. 그리고 이 유사도를 이용한 의미적 유사성 검색 알고리즘을 제안하였다. 의미적 유사성 검색을 처리하기 위해 Fagin의 문턱값 알고리즘(threshold algorithm)을 다음과 같이 변형한 기법을 사용하였다. 우선 사용하는 유사도 함수가 단조 증가 성질을 갖지 않기 때문에 유사도 함수에 맞는 문턱값을 재정의 하였다. 또 역색인 리스트의 구조를 사용하여 중간 검색을 생략할 수 있는 클러스터 스키핑 기법과 역색인 리스트 액세스 순서를 제안하였다. 실제 GO와 주석 정보를 이용하여 성능 평가를 했으며 제안한 알고리즘은 효율적인 알고리즘임을 보였다.

키워드 : Gene Ontology, 의미적 유사성 검색

Abstract Searching for gene products which have similar biological functions are crucial for bioinformatics. Modern day biological databases provide the functional description of gene products using Gene Ontology(GO). In this paper, we propose a technique for semantic similarity search for gene products using the GO annotation information. For this purpose, an information-theoretic measure for semantic similarity between gene products is defined. And an algorithm for semantic similarity search using this measure is proposed. We adapt Fagin's Threshold Algorithm to process the semantic similarity query as follows. First, we redefine the threshold for our measure. This is because our similarity function is not monotonic. Then cluster-skipping and the access ordering of the inverted index lists are proposed to reduce the number of disk accesses. Experiments with real GO and annotation data show that GORank is efficient and scalable.

Key words : Gene Ontology, Semantic similarity search

1. 서 론

최근 많은 생물학 데이타베이스에서 주석(annotation) 데이타를 제공하고 있다. 주석 데이타란 유전자 산물의 기능, 작용 위치와 같은 정보를 기술한 데이타를 말한다. Gene Ontology(GO)는 이와 같은 주석 데이타를 기술하기 위한 공통의 용어를 제공함으로써 여러 데이타베이스간에 주석 데이타를 공유할 수 있고 액세스 하기 쉽도록 해준다[1]. 최근 GO의 의미적 정보를 활용하기 위한 방안으로 의미적 유사성(semantic similarity)에 대한 연구가 활발히 이루어 지고 있다. 의미적 유사성이란 두 객체가 의미적으로 얼마만큼의 관련이 있는가를 측정하는 것이다. 이와 같이 의미적 유사성이 주목을 받으며, GO의 의미적 정보를 사용한 질의 기법의 하나로 의미적 유사성 검색이 주목을 받고 있다. 의미적 유사성 검색이란 주석 데이타와 온톨로지의 의미 정보를 활용하여 유사한 기능을 하는 유전자 산물을 검색하는 것을

· 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 육성 지원 사업(IITA-2005-C1090-0502-0016)의 연구결과로 수행되었으며 BK21사업의 지원을 받아 수행되었음.

† 학생회원 : 서울대학교 전기컴퓨터공학부
kskim@oopsla.snu.ac.kr
swyoo@oopsla.snu.ac.kr
†† 종신회원 : 서울대학교 전기컴퓨터공학부 교수
hjk@snu.ac.kr

논문접수 : 2006년 1월 19일
심사완료 : 2006년 9월 9일

말한다.

이와 같은 의미적 유사성 검색을 위해서는 의미적 유사도의 측정 방법이 필요하다. 그러나 아직까지 유전자 산물에 대한 유사도 측정 방법은 명확히 제안된 바는 없다. 이에 본 논문에서는 Lin[2]의 유사도 정의를 사용한 유사도 함수를 정의하였다. Lin의 정의를 사용한 이유는 이 정의가 매우 일반적이며 따라서 여러 도메인에서 널리 사용되고 있기 때문이다.

그러나 Lin의 정의에 따른 의미적 유사도 함수는 메트릭(metric) 공간상의 거리 함수와는 달리 삼각부등식 성질을 만족시키지 못한다[2]. 기존의 유사성 검색 기법은 주로 메트릭 공간을 다루고 있기 때문에 기존의 유사성 검색 기법을 의미적 유사성 검색에 그대로 적용하는 것은 어렵다. 따라서 본 논문에서는 의미적 유사성 질의를 처리하기 위한 방법으로 Fagin의 문턱값 알고리즘(Threshold Algorithm)[3]을 변형한 알고리즘을 제안하였다. 문턱값 알고리즘을 적용하기 위해 변경한 사항은 다음과 같다.

- 문턱값(threshold)의 재정의: 우리가 사용하는 유사도 함수는 단조 증가 성질을 갖고 있지 않기 때문에, 단조 증가 함수를 필요로 하는 문턱값 알고리즘을 그대로 사용할 수 없다. 이를 해결하기 위해 우리의 유사도 함수에 맞는 문턱값을 재정의하여 사용하였다.
- 중간 검색의 생략: 검색 성능을 향상시키기 위해 역색인 리스트의 구조를 활용한 중간 검색 생략 기법을 제안하였다.

우리는 제안한 기법이 항상 올바른 상위 k 결과를 찾는 것을 보장함을 증명하였으며, 실제 GO 데이타를 사용한 성능 평가를 통해 제안한 기법이 효율적이며 확장성을 가진 알고리즘임을 보였다.

본 논문에서는 Lin의 유사성 정의에 따른 유사도 함수를 사용하였을 때의 검색 기법에 대한 효율성만을 다뤘으며 유사도 함수의 효과성(effectiveness)에 대해서는 다루지 않았다.

2. 관련연구

2.1 의미적 유사성

온톨로지를 사용한 의미적 유사성을 측정하는 방법에 대해 많은 연구가 이루어졌다. 그 중 Lin[2]은 정보 이론의 관점에서 유사도에 대한 일반적인 정의를 제안하였다. 이 유사도 정의는 문헌 검색[4], 웹페이지의 유사성 검색[5] 등 많은 도메인에서 사용되었다. 이와 같이 Lin의 정의는 매우 일반적이며 많은 도메인에서 사용되기 때문에, 본 논문에서도 유전자 산물의 유사도 측정을 위해 Lin의 유사도 정의를 사용하였다.

온톨로지 용어의 의미적 유사성에 대해서는 많은 연구가 이루어졌다. [6,7]에서는 온톨로지 그래프에서의 용어 사이의 경로를 이용한 유사도 측정 방법을 제안하였다. 그래프 경로를 이용하는 방법은 온톨로지의 모든 간선(edge)의 가중치가 균일하다고 전제한다. 그러나 실생활의 온톨로지는 이런 가정이 성립하기 힘들며, GO 역시 이런 가정이 성립하지 않는다[8]. 이런 한계를 극복하기 위한 방안으로 최근에는 정보 이론적 측정방법이 제안되었다[2,9,10]. 그 중 Lin은 정보이론[11]에 바탕을 둔 유사성에 대한 일반적인 정의를 제안하였다[2]. Lin의 정의에 따르면 두 개체 A, B의 유사성 $\mathrm{Sim}(A,B)$는 다음과 같이 정의된다.

$$\mathrm{Sim}(A,B)=\frac{\mathrm{IC}(\mathrm{common}(A,B))}{\mathrm{IC}(\mathrm{description}(A,B))}$$

여기서 IC는 정보량(information content)을 나타낸다. 즉, 이 수식은 두 개체 A, B의 유사성을 A, B를 정의하기 위해 필요한 정보량과 A, B의 공통점을 기술하기 위해 필요한 정보량의 비율로 정의한다.

위의 개념을 바탕으로 서로 독립적인 특성 t_i의 집합으로 표현 될 수 있는 개체에 대해 Lin의 유사성 정의는 다음과 같이 표현된다[2,4].

$$\mathrm{Sim}(A,B)=\frac{2\cdot\sum_{t\in A\cap B}\mathrm{IC}(t)}{\sum_{t\in A}\mathrm{IC}(t)+\sum_{t\in B}\mathrm{IC}(t)} \quad (1)$$

즉 이 수식은 A, B 공통으로 갖는 특성에 대한 정보량과 A, B 각각이 갖는 특성들의 정보량 합의 비율로 유사도를 정의한다.

특성 t를 객체가 갖는 용어라 본다면 $\mathrm{IC}(t)$는 다음과 같이 정의된다.

$$\mathrm{IC}(t)=-\log\Pr[t]$$

$\Pr(t)$는 용어 t가 나타나는 확률로서 이 정의에 따르면 빈번히 나타나는 용어일수록 정보량이 작으며 희박하게 나타나는 용어의 정보량은 증가하게 된다. 용어가 나타나는 확률을 계산할 때 온톨로지 상의 계층 구조를 고려하면 자손 노드의 용어가 나타나면 선조 노드의 용어 역시 나타난 것이 된다. 따라서 선조 노드로 올라갈수록 (일반적인 용어가 될 수록) 용어의 정보량이 감소하는 효과가 있다.

2.2 유사성 검색

유사성 검색 역시 활발한 연구가 진행중인 분야이다. 효율적인 k-최근접 질의 처리를 위한 여러 방법[12-14]들이 제안되었으나 대부분 유클리디안 거리를 사용하는 경우 또는 메트릭 공간에서의 거리 함수만을 고려하고 있다. 이런 알고리즘은 주로 거리 함수의 삼각 부등식 성질을 이용한 색인 기법을 사용한다. 우리가 사용한 유

사도 함수는 메트릭 성질을 갖지 않기 때문에 이런 기법을 사용하기에는 어려움이 있다. 문턱값 알고리즘[3]은 역색인 리스트를 사용하여 상위 k 개의 질의 결과를 검색하는 기법으로 보다 일반적인 상황에 적용할 수 있다. 따라서 본 논문에서는 문턱값 알고리즘을 사용해서 의미적 유사성 검색을 처리하는 방법을 제안하였다.

[15]에서는 같은 시기에 문턱값 알고리즘과 동일한 방법을 제안하였다. [15]에서는 기본적인 검색 방법 외에도 역색인 리스트의 분포를 사용해 디스크 액세스를 줄이는 방법을 제안하였다. 이는 우리가 5.2절에서 제안한 역색인 리스트 액세스 순서와 유사한 면이 있지만 우리는 클러스터를 이룬 역색인 리스트를 고려하고 있다는 점에서 차이가 있다.

본 논문에서 제안한 기법은 문턱값 알고리즘과 관련이 많기 때문에 문턱값 알고리즘의 기본 구조에 대해 간략히 설명하겠다.

2.3 문턱값 알고리즘 개괄[3]

문턱값 알고리즘은 객체가 m개의 특성(feature)을 갖고 있으며 각 특성에 대한 점수를 갖는 상황을 고려한다. 각 특성에 대해 특성값의 내림차순으로 정렬된 역색인 리스트가 있으며 이 리스트를 순차적으로 액세스 할 수도 있고(정렬 액세스), 특정 객체에 대해 어떤 특성값이 얼마인지도 알 수 있다(랜덤 액세스). m개의 점수를 합하는 집계 함수(aggregation function) $f(x_1,\cdots,x_m)$가 있으며, 문턱값 알고리즘의 목적은 이 집계 함수의 값이 높은 상위 k개의 객체를 검색하는 것이다. 문턱값 알고리즘의 질의 처리 방식은 다음과 같다.

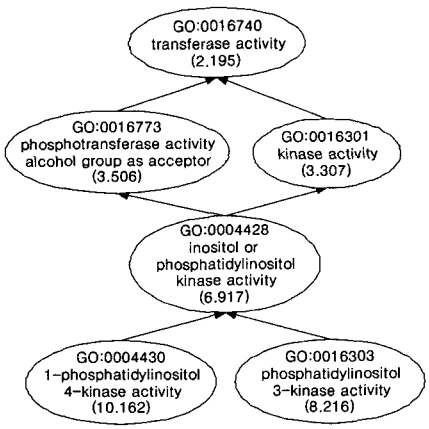
라운드-로빈 방식으로 m개의 리스트 L_i에 대해 정렬 액세스를 한다. 이때 각 리스트에서 탐색된 객체 R에 대해 다른 리스트에 대한 랜덤 액세스를 통해 전체 점수 $f(x_1,\cdots,x_m)$를 계산한다. 만일 이 점수가 지금까지 발견한 객체들의 점수 중 상위 k개에 들면 객체 R과 그 점수를 기억해둔다.

문턱값 알고리즘에서는 모든 리스트를 전부 검색하지 않고 중간에 종료하기 위해 문턱값(threshold value)을 사용한다. 각 리스트 L_i에 대한 정렬 액세스에서 마지막으로 액세스한 객체의 점수를 $\underline{x_i}$라 하자. 문턱값은 $\tau = f(\underline{x_1},\cdots,\underline{x_m})$으로 계산한다. 만일 현재 τ보다 전체 점수가 큰 k개의 결과를 발견했다면 검색을 중단하고 현재까지 발견한 상위 k개의 객체를 결과로 반환한다.

문턱값 알고리즘은 이와 같이 중간에 검색을 중단하여도 $f(\underline{x_1},\cdots,\underline{x_m})$가 단조 증가 함수이면 항상 올바른 결과를 찾을 수 있음을 보장한다. 문턱값의 의미는 앞으로 발견될 수 있는 최대 점수를 의미한다. $\underline{x_i}$는 앞으로 발견할 수 있는 x_i의 최대값이다. 따라서 단조 증가 함수의 정의에 따라 앞으로 발견하게 되는 객체에 대해서는 $f(x_1,\cdots,x_m) \le f(\underline{x_1},\cdots,\underline{x_m})$이 성립한다. 문턱값 알고리즘은 이와 같은 구조의 문제를 풀기 위한 알고리즘 중 가장 최적화된 성능을 보장한다.

3. 유사성 검색 예제 및 시스템 구조

그림 1은 GO 그래프의 일부와 두 유전자 산물 PI4KB_HUMAN, Q15134의 주석 정보를 나타낸다. GO 그래프는 DAG(Directed Acyclic Graph) 형태의 모델



(a) Subgraph of GO

PI4KB_HUMAN

GO ID	Evidence Code
GO:0004430	TAS
GO:0016301	IEA
GO:0016740	IEA
GO:0016773	IEA

Q15134

GO ID	Evidence Code
GO:0004428	IEA
GO:0016303	TAS

(b) Annotation information

그림 1 GO 데이타와 주석 정보 예제

을 갖는다. GO 그래프에서 각 노드는 용어를 내며 괄호 안의 숫자는 해당 용어의 정보량을 말한다. 정보량의 의미는 4절에서 설명한다. 간선은 용어 사이의 관계를 나타내며 is_a, part_of 관계가 있다.

주석 정보는 각 유전자 산물의 생물학적인 특성을 기술한다. 각 주석은 증거 코드가 있는데, 이는 주석이 어떤 출처를 통해 제공되었는지를 알려준다. 주석 코드는 유사도 검색시 주석의 가중치를 계산하는 데에 사용할 수 있다. 가중치를 결정하는 데에는 GO 홈페이지에의 주석 계층 구조[1]를 참고할 수 있다.

의미적 유사성 검색을 예제를 통해 살펴보자. 생물학자가 "kinase activity"와 관련된 유전자 산물을 검색하기를 원한다고 하자. 이때 "kinase activity" 용어로 주석된 유전자 산물만을 검색 결과로 원하지는 않을 것이다. 대신 "kinase activity"와 유사한 의미를 가진 용어에 대해 주석 된 유전자 산물을 질의 결과로 제공한다면 온톨로지의 의미적 정보를 활용한 결과를 얻을 수 있다.

또한 새로 밝혀진 유전자 산물이 Q15134와 유사한 기능을 한다고 밝혀졌을 때 이 유전자 산물과 유사한 기능을 하는 유전자 산물이 무엇인지 검색하는 것이 필요하다. 이때에는 Q15134의 주석 용어들과의 온톨로지 상에서 의미적으로 유사한 다른 용어들로 주석 된 유전자 산물들에 대해 검색을 할 수 있어야 한다.

GORank 시스템은 이와 같이 온톨로지의 의미적 정보를 이용해 의미적 유사성 검색을 할수 있는 시스템이다. GORank의 시스템 구조는 그림 2과 같다. GO 용어간의 유사도는 미리 계산하여 저장해 둔다. 역색인 리스트는 각 용어 별로 만들며 유전자 산물 ID를 갖고 있다. 역색인 리스트의 자세한 구조는 5.1절에서 설명할 것이다. 질의는 유전자 산물 ID 또는 GO 용어의 집합과 함께 결과 크기로 이루어진다. 질의가 주어지면 질의 처리기는 역색인 리스트를 검색하며 실시간으로 유전자 산물간의 유사도를 계산하고 결과로는 k개의 유사한 상위 유전자 산물이 반환된다.

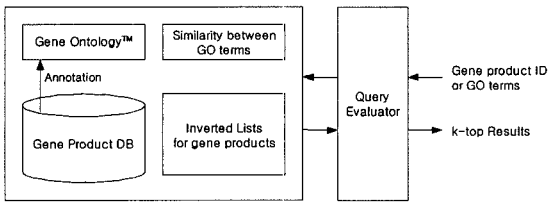


그림 2 GORank 시스템 구조

4. 의미적 유사성

4.1 유전자 산물의 의미적 유사성

이번 절에서는 식 (1)을 사용하여 유전자 산물간의 유사도를 측정하는 함수를 정의한다. 식(1)을 사용하기 위해서는 각각의 유전자 산물이 갖는 정보량과 두 유전자 산물간에 공유하는 정보량이 필요하다. 한 유전자 산물이 갖는 정보량은 주석 용어의 정보량의 합으로 나타낸다. 주석 용어의 정보량은 앞에서 설명한대로 용어가 나타나는 확률을 이용해 구할 수 있다. 이때 각 주석 용어가 나타나는 확률은 독립적이라는 가정이 필요하다. 또한 각 주석 용어의 주석 타입이 다를 수 있기 때문에 다음과 같이 주석 용어의 주석 타입에 따른 가중합으로 계산한다. 다음 식에서 $AW(A,t)$는 유전자 산물 A에서의 용어 t의 주석 가중치이다. 이 가중치는 앞서 말했듯이 주석의 증거 코드에 따라 결정된다. $Ann(A)$는 유전자 산물 A의 주석 용어의 집합을 나타낸다.

$$IC(A)=\sum_{t\in Ann(A)}(AW(A,t)\cdot IC(t))$$

이제 두 유전자 산물이 공유하는 정보량을 정의하자. 두 유전자 산물간 공유 정보량은 주석 용어간 유사성을 사용하여 계산할 수 있다. 한 주석 용어 t의 상대 유전 산물 A의 주석용어와의 유사도중 최대값을 그 용어에 대한 공유하는 정도를 나타낸다고 하자. 이를 $MaxSim(A,t)$이라고 하면 다음과 같이 나타낼 수 있다. 식 (2)에서 $Sim_T(k,t)$는 용어 k, t 사이의 유사도를 나타낸다.

$$MaxSim(A,t)=\max_{k\in Ann(A)} Sim_T(k,t) \quad (2)$$

$MaxSim(A,t)$을 이용해 유전자 산물 A, B의 공유 정보량을 나타내면 다음과 같다.

$$SharedIC(A,B)=\sum_{t\in Ann(A)}\{MaxSim(t,B)\cdot AW(A,t)\cdot IC(t)\}$$

이때 $SharedIC(A,B)\neq SharedIC(B,A)$임을 주목하자.

위의 정의를 사용하여 두 유전자 산물간 유사도 $Sim(A,B)$는 다음과 같이 나타낼 수 있다.

$$Sim(A,B)=\frac{SharedIC(A,B)+SharedIC(B,A)}{IC(A)+IC(B)} \quad (3)$$

4.2 유사도 계산 예제

위에서 정의한 식을 사용해 그림 1의 예제의 PI4KB_HUMAN, Q15134의 유사성을 계산해 보자. 우선 각 유전자 산물의 정보량을 계산하자. 주석 코드 TAS, IEA 각각에 대한 주석 가중치는 1.0, 0.1로 가정하자. 유전자 산물의 정보량은 다음과 같다.

1) http://www.geneontology.org/GO.evidence.shtml

$$IC(PI4KB\_HUMAN) = 11.063$$
$$IC(Q15134) = 8.908$$

다음으로 공통 정보량을 계산한다. $MaxSim(p,t)$을 계산하기 위해 그림 3과 같은 용어간 유사도 배열을 만든다. 이 배열은 용어 유사성을 측정하는 방법으로 계산할 수 있다. 이 배열의 행은 PI4KB_HUMAN의 주석 용어가 위치하고 열에는 Q15134의 주석 용어가 위치한다. 배열의 각 원소는 행과 열에 대응하는 용어간의 유사도를 나타낸다. 이때 $MaxSim(Q15134, GO0004430)$을 계산하기 위해서는 첫 행의 최대값을 구하면 된다. $MaxSim(Q15134, GO0004430)$의 값은 0.773이 된다.

	GO:0004428	GO:0016303
GO:0004430	0.598	0.773
GO:0016301	0.581	0.515
GO:0016740	0.482	0.422
GO:0016773	0.589	0.770

그림 3 용어간 유사도 배열

공통 정보량은 다음과 같다.

$$SharedIC(PI4KB\_HUMAN, Q15134) = (1.0 \times 773 \times 0.10.162) + (0.1 \times 0.581 \times 3.307) + (0.1 \times 0.482 \times 2.195) + (0.1 \times 0.770 \times 3.506) = 8.423$$

$$SharedIC(Q15134, PI4KB\_HUMAN) = (0.1 \times 0.598 \times 6.917) + (1.0 \times 0.773 \times 8.216) = 6.765$$

따라서 두 유전자 산물의 유사도는 다음과 같다.

$$Sim(PI4KB\_HUMAN, Q15134) = \frac{8.423 + 6.765}{11.063 + 8.908} = 0.761$$

4.3 의미적 유사도 함수의 성질

이번 절에서는 4.2절에서 정의한 유전자 산물 유사성 측정 함수가 갖는 성질에 대해 알아보자. 우선 다른 유사도 함수와 비교를 통해 어떤 성질을 갖고 있는지 살펴본다. 현재 다른 문헌에 소개된 유전자 산물간 유사도 측정함수는 다음 두 가지가 있다[16].

$$Sim(A,B) = \frac{1}{m \times n} \times \sum_{t_i \in Ann(A), t_j \in Ann(B)} Sim_T(t_i, t_j) \quad (4)$$

$$Sim(A,B) = \frac{1}{m \times n} \times \left(\sum_{t_i \in Ann(A)} \max_{t_j \in Ann(B)} \left(Sim_T(t_i, t_j) \right) + \sum_{t_i \in Ann(B)} \max_{t_j \in Ann(A)} \left(Sim_T(t_i, t_j) \right) \right) \quad (5)$$

두 함수 모두 주석 용어의 유사성을 조합하는 방식으로 유사도를 계산한다. 식 (4)는 모든 용어쌍에 대한 유사도를 평균한 것이고 식 (5)는 가장 유사한 용어 쌍의 유사도 만을 평균한 것이다.

식 (4)의 경우에는 유전자 산물 주석 용어가 여러 개인 경우 $Sim(A,A) \neq 1$되는 문제점이 있다. 즉 환원성(Reflexivity)을 갖지 못한다. 식 (5)의 경우 이 문제를 해결하기 위해 고안되었다. 이렇게 가장 유사한 용어 쌍을 사용하는 다-대-일 매칭은 환원성을 얻기 위해 많이 사용된다. 우리가 제안한 유사도 함수 역시 다-대-일 매칭을 사용하였다.

위의 두 함수가 용어 유사도만 사용한데 반해 식 (3)은 유전자 산물의 정보량을 고려하고 있다. 이는 각 용어의 정보량을 가중치로 고려하여 계산한 것으로 볼 수 있다. 이는 정보량이 큰 용어에 대한 차이가 클수록 유사성의 차이가 크도록 해준다. 또한 식 (3)은 주석의 종류에 따른 가중치를 반영하였다. 반면 식 (5)는 단순히 용어간 유사도를 평균하기 때문에 이러한 차이를 반영할 수 없다.

마지막으로 의미적 유사성 함수는 메트릭 공간에서의 거리 함수와는 달리 삼각 부등식 조건을 만족시키지 못한다. 이는 의미적 유사성 함수의 일반적인 성질로 알려져 있다. [2]에서는 의미적 유사성 함수가 본래 삼각 부등식 조건을 갖지 못함을 실제적인 반례를 들어 설명하고 있다. 다음 장에서 설명하듯이 이는 유사성 검색을 처리하기 위한 알고리즘의 선택에 영향을 준다.

5. 의미적 유사성 검색

이번 장에서는 4절에서 정의한 유사도 함수를 사용한 유사성 검색 방법을 제안하겠다. 우리가 의미적 유사성 검색을 위해 제안하는 방법은 문턱값 알고리즘을 기본 구조로 한다. 이를 위해 우선 집계 함수 및 각 특성의 점수에 대한 정의가 필요하다. 다음 절에서는 문턱값 알고리즘을 사용한 의미적 유사성 검색 방법을 설명한다.

5.1 질의 처리 과정 및 문턱값 정의

질의 유전자 산물을 p_q, 주석 용어 집합을 $Ann(p_q) = \{t_1, \cdots, t_n\}$라 하자.

$d_i = MaxSim(p, t_i)$로 나타내면, $Sim(p_q, p)$는 다음과 같이 나타낼 수 있다.

$$Sim(p_q, p) = \frac{1}{IC(p_q) + IC(p)} \times \{d_1 \cdot AW_1 \cdot IC_1 + \cdots + d_n \cdot AW_n \cdot IC_n + SWS(p, p_q)\} = f_{p_q,p}(d_1, \cdots, d_n) \quad (6)$$

즉 n개 주석 용어의 $MaxSim(p, t_i)$에 대한 집계 함수 $f_{p_q,p}(d_1, \cdots, d_n)$로 유사도 함수를 나타낸다. 이 집계함수에 문턱값 알고리즘을 적용하기 위해 다음과 같이 역색

인 리스트를 구성한다.

역색인 리스트는 온톨로지의 각 용어에 대해 만들어진다. 주석 용어 t_i에 대한 역색인 리스트는 $MaxSim(p,t_i)$이 0 이상인 유전자 산물 p의 ID를 $MaxSim(p,t_i)$의 내림차순으로 색인한다. 이렇게 역색인 리스트를 구성하면 d_i 값의 내림차순으로 순차 액세스를 할 수 있게 된다. 그림 4는 역색인 리스트의 예제로 GO:0004430과 GO:0004428 용어에 대한 역색인 리스트를 나타낸다.

GO:0004430

MaxSim	Gene Product ID
1.0	PI4KB_HUMAN
...	...
0.773	Q15134
...	...

GO:0004428

MaxSim	Gene Product ID
1.0	Q15134
...	...
0.598	PI4KB_HUMAN
...	...

Sorted by MaxSim

그림 4 역색인 리스트 구조

질의 처리는 문턱값 알고리즘 방식에 따라 다음과 같이 이뤄진다. 질의 p_q가 주어지면 주석 용어 $t_i(1 \le i \le n)$의 역색인 리스트들을 순차적으로 검색한다. n개의 리스트에 대해 정렬 액세스하며 리스트의 유전자 산물과 p_q와의 유사도를 계산한다. 계산한 리스트 중 유사도가 높은 상위 k개의 유전자 산물을 기억해두고 검색이 끝난 후 결과로 반환 한다. 이때 문턱값 알고리즘의 중단 조건을 적용하게 되면 모든 역색인 리스트를 풀 스캔하지 않고 검색을 종료 할 수 있다. 그러나 문턱값 알고리즘의 중단 조건에서 사용한 문턱값은 집계함수가 단조 증가해야 한다는 조건이 있다. 하지만 식 (6)의 유사도 함수는 d_i에 대해 단조 증가하지 않는다. d_i 값이 증가해도 IC(p)와 $SharedIC(p,p_q)$에 따라 전체 값이 감소할 수 있기 때문이다. 따라서 식 (6)에 맞는 문턱값을 정의해야 한다.

이제 부터는 식 (6)의 유사도 함수에 대한 문턱값을 정의하겠다. 우선 $SharedIC(p,p_q)$의 최대값에 대해 다음의 정리가 성립한다.

보조 정리 5.1. ($SharedIC(p,p_q)$의 최대값)

모든 p에 대해 다음이 성립한다.

$$SharedIC(p,p_q) \le \max\{d_1,\cdots d_n\} \cdot IC(p)$$

증명: $SharedIC(p,p_q)$는 다음과 같다.

$$SharedIC(p,p_q) = \sum_{t \in Ann(p)} (AW(p,t) \cdot MaxSim(p_q,t) \cdot IC(t))$$

$SharedIC(p,p_q)$이 최대값을 가질 때에는 $AW(p,t)$와 $MaxSim(pq,t)$가 최대값을 가질 때이다. 모든 p, t에 대해, 다음이 성립한다.

$$0 \le AW(p,t) \le 1$$

$$0 \le MaxSim(p_q,t) \le \max\{d_1,\cdots,d_n\}$$

따라서 $SharedIC(p,p_q) \le \max\{d_1,\cdots,d_n\} \cdot IC(p)$. □

보조 정리 5.1을 이용해 $Sim(p,p_q)$의 최대값을 다음과 같이 알 수 있다.

보조 정리 5.2. ($Sim(p,p_q)$의 최대값)

각 역색인 리스트에서 정렬 액세스를 통해 마지막으로 검색한 유전자 산물의 d_i 값을 $\underline{d_i}$라 하고, 데이타베이스 내 유전자 산물 정보량의 최대값을 IC_{max}라 하자. 앞으로 검색할 유전자 산물 p에 대해 다음이 성립한다.

$$Sim(p_q,p) \le \frac{1}{IC(p_q)+IC_{max}} \times \{\underline{d_1} \cdot AW_1 \cdot IC_1 + \cdots + \underline{d_n} \cdot AW_n \cdot IC_n + \max(\underline{d_1},\cdots,\underline{d_n}) \cdot IC_{max}\} \quad (7)$$

증명: 식 (6)의 유사도 함수를 다음과 같이 IC(p)에 대한 함수로 보자.

$$f(IC(p)) = \frac{\alpha \cdot IC(p_q) + \beta \cdot IC(p)}{IC(p_q)+IC(p)} \qquad (0 \le \alpha,\beta \le 1)$$

특정 IC(p)에 대해 α, β의 값이 최대일 때 $f(IC(p))$의 값이 최대가 된다.

α의 최대값은 다음과 같다.

$$\alpha \le \frac{\underline{d_1} \cdot AW_1 \cdot IC_1 + \cdots \underline{d_n} \cdot AW_n \cdot IC_n}{AW_1 \cdot IC_1 + \cdots AW_n \cdot IC_n}$$

보조 정리 5.1에 의해 β의 최대값은 $\max(\underline{d_1},\cdots,\underline{d_n})$이다.

$f(IC(p))$는 IC(p)>0에 대해, $\alpha < \beta$인 경우는 단조 증가하고 $\alpha > \beta$인 경우에는 단조 감소한다. α와 β가 모두 최대값을 가진다면, $\alpha < \beta$이다.

따라서 IC(p)가 IC_{max}일 때, $Sim(p_q,p)$의 최대값이 된다. □

보조 정리 5.2를 이용해 문턱값을 정의할 수 있다. 정리 5.1은 이를 보여준다.

정리 5.1 (식 (6)의 문턱값)

식 (7)을 문턱값으로 사용하면 문턱값 알고리즘은 올바른 상위 k 결과를 찾는다.

증명: 식 (7)의 최대값은 정렬 액세스를 진행하는 도중, $\underline{d_1},\cdots,\underline{d_n}$ 값만 변화한다. 또 $\underline{d_1},\cdots,\underline{d_n}$ 의 값에 단조증가 한다. 따라서 이를 문턱값으로 사용하면 문턱값 알고리즘의 정의에 따라 올바른 상위 k 결과를 검색할 수 있음이 보장된다. □

5.2 중간 검색의 생략

앞에서 정의한 문턱값을 사용하면 올바른 질의 결과를 찾는 것을 보장하지만 성능상 문제점을 보인다. 극단적인 경우를 고려한 값이기 때문에 문턱값이 충분히 감소하는데 까지 많은 유전자 산물에 대해 유사도를 계산해야 한다. 유사도 계산과정에는 IC(p), α 와 β 를 계산해야 하는데 이는 디스크 액세스를 필요로 한다. 따라서 되도록 적은 수의 유전자 산물을 검색하는 것이 성능향상에 도움이 된다. 이를 위해 이번 절에서는 역색인 리스트의 구조를 사용해 중간 유전자 산물 검색을 생략하는 방법에 대해 알아보겠다.

5.2.1 클러스터 스키핑

앞에서 설명했듯이 역색인 리스트는 MaxSim에 대해 정렬되어 있다. 따라서 용어 t의 역색인 리스트의 유전자 산물은 MaxSim에 대해 클러스터를 이룬다. 즉 같은 MaxSim을 갖는 유전자 산물들은 역색인 리스트에서 인접해있다. 이것은 클러스터 안의 유전자 산물들이 공통적으로 같은 주석 용어를 갖는 것을 뜻한다. 실제 GO에서는 같은 주석 용어를 갖는 유전자 산물이 많기 때문에 이와 같은 클러스터가 역색인 리스트에 많이 나타난다. 이러한 역색인 리스트의 구조를 사용하면 클러스터 내에서의 검색 횟수를 감소시킬 수 있다. 이렇게 클러스터 내의 검색을 생략하는 것을 클러스터 스키핑이라고 한다.

이를 위해 클러스터 안의 유전자 산물을 $IC(p)$에 대해 내림차순으로 정렬하여 역색인 리스트를 구성한다. 즉 전체 역색인 리스트는 MaxSim에 의해 정렬되며 MaxSim이 같은 유전자 산물들에 대해서는 $IC(p)$로 정렬한다. 그림 5는 클러스터 된 역색인 리스트를 나타낸다.

현재 결과 힙의 최소값을 최소 유사도라 하면, $IC(p)$에 대해 정렬된 클러스터 검색시 다음 조건이 성립할 때 클러스터의 검색을 중단할 수 있다.

$$\frac{1}{IC(p_q)+IC(p)}\times\{\underline{d_1}\cdot AW_1\cdot IC_1+\cdots+\underline{d_n}\cdot AW_n\cdot IC_n$$

$$+\max(\underline{d_1},\cdots,\underline{d_n})\cdot IC(p)\}\leq \text{최소유사도}$$

즉, 클러스터 내에서 위의 새로운 중단 조건을 적용하면 클러스터 내의 모든 유전자 산물을검색하지 않아도 된다. 이 중단조건은 다음과 같은 이유에서 성립한다. $Sim(p,p_q)$의 최대값을 계산할 때 발견할 수 있는 $IC(p)$의 최대값인 IC_{max} 을 사용하였다. 한 클러스터 내에서는 현재의 $IC(p)$값이 IC_{max} 이 된다(클러스터 내에서는 $IC(p)$에 대해 내림차순으로 정렬되어 있기 때문에). 따라서 한 클러스터 내에서는 위와 같은 새로운 중단 조건을 적용하여 클러스터내의 모든 유전자 산물을 검색하지 않을 수 있다.

5.2.2 역색인 리스트 액세스 순서

클러스터 스키핑을 사용할 경우 역색인 리스트의 액세스 순서가 중요하다. 이유는 역색인 리스트 액세스 순서에 따라 클러스터 스키핑의 조건을 빨리 만족시킬 수 있기 때문이다. 그림 5의 클러스터 역색인 리스트를 고려해보자. 박스는 클러스터를 나타내며 박스 안의 숫자는 클러스터 안의 유전자 산물의 개수이다. 라운드-로빈 방식으로 각 리스트를 액세스 할 경우 $\underline{d_2}$ 가 감소할 때까지 액세스 해야 하는 유전자 산물의 개수는 149(=50+50+49)개이다. 만일 2번 리스트만을 액세스 하게 된다면 50개의 유전자 산물 액세스 후 $\underline{d_2}$ 가 감소한다. 반대로 1번 리스트만을 액세스 한다면 100개 유전자 산물을 액세스 한 후에야 $\underline{d_1}$ 이 감소한다. $\underline{d_n}$ 값의 감소가 중요한 이유는 클러스터 스키핑 조건에 영향을 주기 때문이다. $\underline{d_n}$ 의 값이 작아야 클러스터 스키핑의 조건을 좀더 빨리 만족시킬 수 있게 된다.

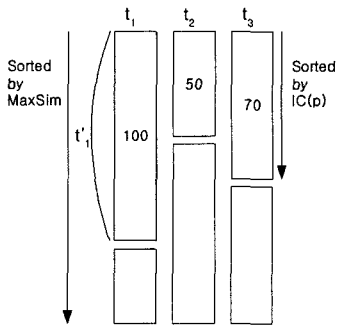


그림 5 클러스터 역색인 리스트

이와 같이 클러스터 스키핑의 효과를 최대화 하기 위해서 다음 순서로 역색인 리스트를 액세스 한다. 우선 라운드-로빈 방식이 아닌 클러스터 단위로 역색인 리스트를 액세스한다. 클러스터 단위로 액세스 한다는 것은 한 개의 클러스터 내의 유전자 산물에 대한 검색을 한 후 다른 클러스터에 대한 검색을 한다는 것을 말한다.

클러스터 스키핑의 조건을 만족하거나 클러스터의 모

든 유전자 산물에 대해 검색을 하면 한 클러스터에 대한 검색을 마치고 다음 클러스터를 선택해야 한다. 클러스터를 선택할 때에는 n개의 리스트 중에서 $\underline{d}_n$ 값이 가장 크고 클러스터의 크기가 가장 작은 리스트를 선택한다. 이는 $\underline{d}_n$ 값 중 최대값을 식 (12)에서 사용하고 있으며 이 최대 $\underline{d}_n$ 값을 줄이기 위해서는 해당 클러스터를 먼저 검색하는 것이 유리하기 때문이다. 그리고 최대 $\underline{d}_n$ 값을 가진 클러스터가 여러 개일 경우에는 가장 작은 클러스터를 검색하는 것이 유리하다.

5.3 알고리즘

그림 6은 전체적인 질의 처리 알고리즘을 나타낸다. EvaluateQuery() 함수는 질의 유전자 산물의 ID와 결과의 크기 k를 인수로 받으며 결과 유전자 산물들과 그들의 유사도를 반환한다.

질의 처리 알고리즘의 주 루프(06~16)에서는 질의 유전자 산물의 주석 용어에 대한 역색인 리스트를 순차 검색하여 각 반복(iteration) 마다 유전자 산물의 ID를 한 개씩 얻어온다. 어느 역색인 리스트에서 ID를 검색할지는 5.2절에서 설명한 방식으로 결정한다. 클러스터 단위로 검색을 하며 ChooseIList() 함수는 역색인 리스트에서 5.2.2 절에서 설명한 방식으로 다음 클러스터를 찾는다. 검색한 유전자 산물과 질의 유전자 산물의 유사도 계산을 한 뒤에는 이를 결과 힙에 넣는다. 결과 힙은 자동으로 유사도의 내림차순으로 k개의 유전자 산물을 저장한다. 그 후 문턱값을 계산하여 결과 힙의 가장 낮은 유사도보다 낮거나 같다면 검색을 종료한다.

Next() 함수에서는 역색인 리스트의 커서가 가리키는 엔트리를 반환하고 커서를 다음 엔트리로 옮기는 역할을 하는데, Next() 함수에서 B_{term}과 B_p는 중복된 검사를 줄이기 위해 이미 검사한 용어와 유전자 산물을 기억해두는 역할을 한다. 검색도중 한 리스트에서 검색한 유전자 산물을 다른 리스트에서 검색할 수 있기 때문에

알고리즘 1 질의 처리

```
//pid_q is the id of query gene product
// k is the size of results
//iList[i](i=1,…,n) is the inverted index list for an annotation term t_i
//lowest[i](i=1,… n) is the last value of MaxSim which has been seen in iList[i]
// B_term is a set of examined terms
// B_p is a set of examined gene products

01: procedure EvaluateQuery(pid_q,k) {
02:     lowest={1,…,1};
03:     resultHeap={};
04:     isEndOfCluster=true;
05:     isEnd=false;
06:     while(!isEnd and eof has not been reached on all inverted index lists) {
07:         if(isEndOfCluster==true) {
08:             i=ChooseIList();
09:             isEndOfCluster=false;
10:         }
11:         entry=Next(i);
12:         ss=CalculateSS(pid_q, entry.pid);
13:         resultHeap.offer(entry.pid, ss);
14:         threshold=CalculateThreshold(lowest[i]);
15:         if(threshold≤lowest_similarity of resultHeap) isEnd=true;
16:     }
17:     return resultHeap;
18: }
19:
20: procedure Next(i) {
21:     entry=iList[i].PresentEntry();
22:     do {
23:         nextEntry=iList[i].NextEntry();
24:         if(nextEntry.isStartOfCluster||CheckSkipCondition(nextEntry.ic)) {
25:             while(nextEntry.term∈B_term) {
26:                 nextEntry=iList[i].nextCluster();
27:             }
28:             B_term=B_term∪nextEntry.term;
29:             isEndOfCluster=true;
30:         }
31:     } while(nextEntry.p∈B_p)
32:     B_p=B_p∪nextEntry.p;
33:     lowest[i]=nextEntry.MaxSim;
34:     return entry;
35: }
```

그림 6 질의 처리 알고리즘

이와 같이 이미 검색한 유전자 산물을 기록하는 과정이 필요하다.

6. 실험 결과

실험은 GO 2005년 11월 버전을 사용하였다. 분자적 작용(Molecular Function)의 용어에 대해서만 실험을 하였다. 사용한 유전자 산물은 GO내에 포함된 유전자 산물을 사용하였다. 주석 정보의 개수는 3,711,735개이며 유전자 산물의 개수는 1,397,464개이다. 시스템 구현은 Java SDK 1.5를 사용하였다. GO 데이타와 용어간 유사도 테이블은 MySQL 4.1을 사용하여 저장하였다. 실험을 한 환경은 PentiumM 1.6GHz CPU, 1G RAM 이며 WindowsXP 운영체제를 사용하였다.

6.1 의미적 유사성 질의 성능

우선 제안한 클러스터 스키핑과 역색인 리스트 액세스 순서의 효과를 알아보기 위해 질의 성능을 측정하였다. 이를 위해 표 1의 5개 유전자 산물에 대해 유사한 유전자 산물 20개를 검색하는 질의를 수행하였다.

표 1 실험에 사용한 질의

No	Gene product	# of Annotation
1	Q4EPJ2_LISMO	7
2	BACH_HUMAN	4
3	TRI34_HUMAN	4
4	Q46YR3_RALEJ	4
5	Q4EPE1_LISMO	5

알고리즘의 성능 개선의 초점이 유전자 산물 검색 횟수를 줄이는 것이기 때문에 결과는 유전자 산물 검색횟수에 대해 측정하였다.

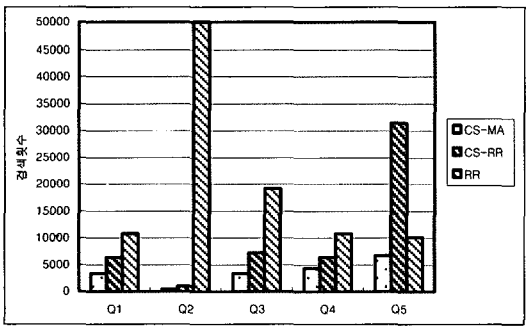


그림 7 질의 수행 결과

질의 수행 결과는 그림 7과 같다. 그래프에서 CS-MA는 클러스터 스키핑과 역색인 리스트 액세스 순서를 모두 적용하였을 때의 결과이고 CS-RR은 클러스터 스키핑을 사용하고 라운드-로빈 방식으로 액세스 하였을 때이다. RR은 클러스터 스키핑을 사용하지 않고 라운드-로빈 방식으로 액세스 하였을 때의 결과이다. 그래프에서 알 수 있듯이 클러스터 스키핑과 역색인 리스트 액세스 순서 적용 모두 성능상 이득을 주었음을 알 수 있다. 성능상 이득의 정도는 질의에 따라 다르게 나타났다. 이득의 정도는 역색인 리스트의 클러스터의 성질에 영향을 받았다. 역색인 리스트에서 초기에 사이즈가 큰 클러스터가 나타나는 경우 라운드-로빈 방식은 매우 비효율적이 된다. 문턱값의 변화가 거의 없게 되어 많은 유전자 산물을 검색하기 때문이다. 이 현상은 2번 질의에서 분명하게 드러난다. 2번 질의의 경우 역색인 리스트 중 한 개의 초기 클러스터의 크기가 115,445개 이다. 이는 주석 용어가 매우 일반적이어서 이 용어로 주석된 유전자 산물의 개수가 매우 많기 때문이다. 이렇게 초기 클러스터의 사이즈가 큰 경우 클러스터 스키핑을 사용하면 클러스터의 모든 유전자 산물에 대해 검색을 하지 않을 수 있음을 알 수 있다.

6.2 문턱값의 변화

그림 8은 질의 2번에 대한 질의 처리 도중의 문턱값의 변화 그래프이다. Lowest는 현재 결과 힙에 저장된 가장 낮은 유사도의 변화를 나타낸다. 문턱값이 Lowest보다 낮아져야 검색이 종료된다. CS-MA의 경우 문턱값이 빨리 감소하는 것을 알 수 있다. RR의 경우에는 문턱 값의 변화가 거의 없으며 이로 인해 검색이 매우 오래 걸린다.

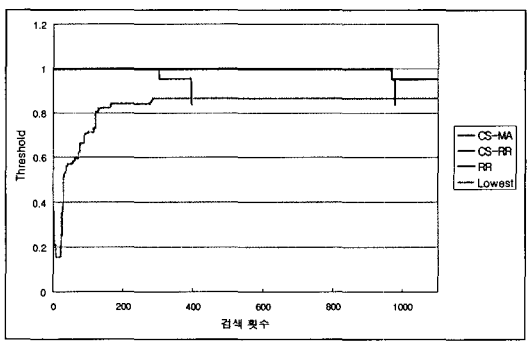


그림 8 문턱값의 변화

6.3 결과 개수에 따른 확장성

다음으로 결과 개수에 따른 성능 변화를 살펴보았다. 그림 9는 1번 질의에 대해 결과의 크기를 변화시키며 질의 수행 시간을 측정한 결과 이다. CS-MA의 경우 결과 크기에 대해 수행시간의 변화가 거의 없음을 알 수 있다. 반면 CS-RR과 RR은 결과 크기가 커지면 수행시간도 증가함을 알 수 있다. 따라서 우리가 제안한 방법은 질의 결과 크기에 대해 확장성(scalability)이 있

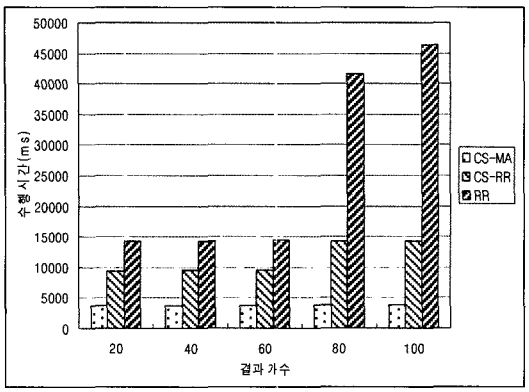


그림 9 결과 개수에 대한 수행시간의 변화

음을 알 수 있다.

7. 결론 및 향후 연구

본 논문에서는 유사한 유전자 산물 검색을 위한 기법을 제안하였다. 우선 유사성 검색을 위해 유전자 산물간의 유사도 측정 방법을 제안하였다. 이 유사도 함수는 Lin의 정보 이론적 유사도 정의를 사용하였다. 그리고 상위 k 질의 처리를 위해 문턱값 알고리즘을 사용하여 의미적 유사성 검색을 할 수 있음을 보였다. 이를 위해 우리가 제안한 유사도 함수에 맞는 문턱값에 대한 정의를 하였다. 또한 역색인 인덱스가 클러스터를 이루고 있음을 이용해 탐색 횟수를 줄이는 기법을 제안하였다. 실제 GO를 이용한 실험에서 우리가 제안한 방법이 효율적이며 결과의 개수가 증가해도 질의 성능에 변화가 적음을 보였다.

GO를 이용한 의미적 유사성 검색은 앞으로 많은 활용 분야가 있을 것으로 기대된다. 본 논문의 의의는 이런 의미적 유사성 활용에서 기본이 될 상위 k 질의를 처리하는 기법을 제안한 것으로 볼 수 있다. 또한 본 논문에서 제안한 기법은 GO에만 국한되지 않고 일반적인 온톨로지-주석 데이타 구조에도 적용이 가능하다.

향후 연구로는 유전자 산물에 대한 다른 여러 정보를 이용해 의미적 유사성 검색에 포함하는 방법에 대한 연구가 필요할 것으로 보인다. 생물정보학에서는 여러 데이타에 대한 통합이 무엇보다도 필수적이다. 또한 GO의 세가지 온톨로지에 따른 의미적 유사성 검색 결과를 통합하는 것도 흥미로운 연구 주제가 될 것이다.

참 고 문 헌

[1] The Gene Ontology Consortium, *Creating the Gene Ontology Resource: Design and Implementation*. Genome Res, 2001. **11**(8): p. 1425-33.

[2] Lin, D. *An Information-theoretic Definition of Similarity*. in *15th International Conf. on Machine Learning*. 1998. San Francisco, CA.

[3] Fagin, R., A. Lotem, and M. Naor, *Optimal Aggregation Algorithms for Middleware*, Journal of Computer and System Sciences, 2003. **66**(4): p. 614-656.

[4] Aslam, J.A. and M. Frost. *An Information-theoretic Measure for Document Similarity*. in *SIGIR*. 2003. Toronto, Canada.

[5] Maguitman, A.G. and F. Menczer. *Algorithmic Detection of Semantic Similarity*. in *WWW*. 2005. Chiba, Japan.

[6] Lee, J.H., M.H. Kim, and Y.J. Lee, *Information Retrieval based on Conceptual Distance in is-a Hierarchies*. Journal of Documentation, 1989. **49**(2): p. 188-207.

[7] Rada, R., et al., *Development and Application of a Metric on Semantic Nets*. IEEE Transactions on Systems, Man and Cybernetics, 1989. **19**(1): p. 17-30.

[8] Lord, P.W., et al. *Semantic Similarity Measures As Tools For Exploring the Gene Ontology*. in *Pacific Symposium on Biocomputing* 2003.

[9] Resnik, P., *Semantic Similarity in a Taxonomy: An Information-based Measure and its Application to Problems of Ambiguity in Natural Language*. Journal of Artificial Intelligence Research, 1999. **11**: p. 95-130.

[10] Jiang, J.J. and D.W. Conrath, *Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy*, in *International Conference Research on Computational Linguistics*. 1997: Taiwan.

[11] Cover, T. and J. Thomas, *Elements of Information Theory*. 1991: Wiley-Interscience.

[12] Hjaltason, G.R. and H. Samet, *Indexing-Driven Similarity Search in Metric Space*. ACM Transactions on Database Systems, 2003. **28**(4): p. 517-580.

[13] Bohm, C., S. Berchtold, and D.A. Keim, *Searching in High-Dimensional Spaces: Index structures for Improving the Performance of Multimedia Databases*. ACM Computing Surveys, 2001. **33**(3): p. 322-373.

[14] Chavez, E., et al., *Searching in Metric Spaces*. ACM Computing Surveys, 2001. **33**(3): pp. 273-321.

[15] Guntzer, U., W.-T. Balke, and W. Kießling, *Optimizing Multi-Feature Queries for Image Databases*, in *VLDB*. 2000: Egypt.

[16] Azuaje, F., H. Wang, and O. Bodenreider, *Ontology-driven Similarity Approached to Supporting Gene Functional Assessment*, in *ISMB Sig meeting on Bio-ontology*. 2005.

김 기 성
정보과학회논문지 : 데이타베이스
제 33 권 제 2 호 참조

유 상 원
정보과학회논문지 : 데이타베이스
제 33 권 제 2 호 참조

김 형 주
정보과학회논문지 : 데이타베이스
제 33 권 제 2 호 참조