클러스터와 온톨로지 정보를 이용한 웹 서비스 매칭 알고리즘☆

Web Service Matching Algorithm using Cluster and Ontology Information

이 용 주*
Yong-Ju Lee

요 약

웹 서비스들의 수가 급격하게 증가함에 따라 사용자가 적합한 웹 서비스를 찾는 것은 매우 중요한 문제로 대두되고 있다. 그러나 전통적인 키워드 탐색 방법은 다음의 두 가지 이유 때문에 문제가 있다: (1) 웹 서비스에 대한 의미적인 정보들을 활용하지 못한다. (2) 사용자의 요구사항을 정확하게 표현하지 못한다. 이러한 키워드 기반 탐색 방법의 한계를 극복하기 위해 본 논문에서는 하나의 새로운 구문 분석 및 온톨로지 학습 방법을 제안한다. 구문 분석 방법은 키워드를 일반화하여 검색 범위를 넓혀주고, 온톨로지 학습 방법은 상관관계를 표현하여 깊이 있는 탐색을 유도한다. 이러한 두 방법을 결합함으로써 재현율과 정확률 둘 다 향상 시킬 수 있는 기법이 될 수 있다. 제안된 방법은 508개의 웹 서비스 집합에 대한 실험을 수행하여 그 성능의 우수함을 보인다.

ABSTRACT

With the growing number of web services, there arise issues of finding suitable services. But, the traditional keyword search method is insufficient for two reasons: (1) this does not capture the underlying semantics of web services. (2) this does not suffice for accurately specifying users' information needs. In order to overcome limitations of this keyword search method, we propose a novel syntactic analysis and ontology learning method. The syntactic analysis method gives us a breadth of coverage for common terms, while the ontology learning method gives a depth of coverage by providing relationships. By combining these two methods, we hope to improve both the recall and the precision. We describe an experimental study on a collection of 508 web services that shows the high recall and precision of our method.

☞ KeyWords : 웹 서비스(web service), 매칭 알고리즘(matching algorithm), 키워드 탐색(keyword search), 구문분석(syntactic analysis), 클러스터(cluster), 온톨로지(ontology)

1. 서 론

웹 서비스는 동적으로 느슨하게 결합된(loosely coupled) 서비스 지향 구조(SOA: Service Oriented Architecture)로 되어있다. 이에 반해 기존의 비즈니스 시스템들은 하부 시스템과 단단히 결합된(tightly coupled) 시스템 의존적인 구조로 되어있어 이식성 및 유지관리 비용이 많이 든다. 향후 소프트웨어 개발은 단순히 한 컴퓨터에서 머무는 것이 아니라 웹 서비스 기술을 이용하여 필요한 모듈을 인터넷에서 구할 수 있고 개발된 소프트웨어를 인터넷에 공개할 수 있다. 이러한 서비스 지향 구조는 '소프트웨어를 서비스로 보는(software as a service)' 중요한 개념이 된다.

현재 웹 서비스의 사용은 사용자들이 이미 알고 있는 몇 개의 서비스들을 이용하거나, 웹 서비스 저장소(예, UDDI[1], Xmethods[2], Web Service List[3])에서 키워드 탐색에 의해 서비스들을 발견하고 있다. 하지만 이러한 키워드 탐색 방법은 다음의 두 가지 이유 때문에 문제가 있다. (1) 나쁜 재현율(recall): 기존의 키워드 탐색 방법은 웹 서

* 정 회 원 : 경북대학교 이공대학 컴퓨터정보학부 교수
yongju@knu.ac.kr
[2009/04/09 투고 - 2009/05/07 심사 - 2009/09/14 심사완료]
☆ 이 논문은 2009년도 경북대학교 학술연구비에 의하여 연구되었음

비스에 대한 의미적인 정보들을 활용하지 못한다. 기존의 방법에서는 키워드가 정확히 일치하는 웹 서비스일 경우에만 발견이 되므로 사용자가 원하는 웹 서비스일지라도 키워드가 일치되지 않는다는 이유로 검색되지 않은 웹 서비스들이 존재한다. (2) 나쁜 정확률(precision): 키워드는 사용자의 요구사항을 정확하게 표현하지 못한다. 검색 결과 중에는 사용자가 원하지 않지만 키워드가 포함된 수많은 웹 서비스들이 포함될 수 있다. 따라서 사용자는 이러한 결과 중에서 다시 원하는 웹 서비스를 찾아야 하는 불편함이 있다.

이러한 키워드 기반 탐색 방법의 한계를 극복하기 위한 기법으로서 시멘틱(semantic) 정보를 이용하는 온톨로지(ontology) 활용 방법이 있을 수 있다[4, 5]. 웹 서비스 저장소에 추가적인 시멘틱 정보(예, WSDL-S[6], OWL-S[7])를 주석처리(annotation)하여 키워드와 일치하지 않은 웹 서비스일지라도 의미적으로 연관성 있는 서비스들에 대해 확장 검색한다. 그렇지만 온톨로지는 대부분 전문가의 수작업으로 구축되고 있으며, 시간 및 인적 제약 때문에 실용적인 온톨로지를 구축하기 어렵다[8]. 또한 현시점에서 웹 서비스 전체에 대해 주석을 다시 단다는 것은 거의 불가능하게 보이며, 이러한 문제는 오늘날 웹 서비스의 확산과 발전을 가로막는 큰 저해 요인이 되고 있다.

다른 기법으로서 클러스터링(clustering) 방법을 이용한 웹 서비스 유사성(similarity) 탐색 방법[9, 10]이 있다. 이 방법에서는 상호 연관성이 높은 단어들을 함께 묶어 클러스터를 형성하게 하고, 웹 서비스를 탐색할 때 사용자가 입력한 키워드뿐만 아니라 그것이 포함되어 있는 클러스터 내의 모든 단어들에 대해 탐색을 수행함으로써 보다 의미 있는 검색이 되도록 한다. 그렇지만 이 방법은 연관성이 높은 단어들을 단지 한 클러스터에 묶어서 동의어(synonym)처럼 취급할 뿐 객체지향 모델과 같은 계층관계(hierarchy)에 따라 유사도를 결정하는 시멘틱 기능은 제공하지 못하고 있다.

본 논문에서는 클러스터링 탐색 방법에 추가적으로 온톨로지를 자동 구축하여 보다 더 효율적인 검색을 지원할 수 있는 클러스터-온톨로지 웹 서비스 매칭 알고리즘을 제안한다. 이를 통해 키워드가 정확하게 일치하지 않더라도 사용자가 원하는 웹 서비스를 검색할 수 있고, 반대로 키워드가 일치하지만 사용자가 의도하지 않은 웹 서비스는 검색 결과에서 제거할 수 있다. 일반적으로 하나의 웹 서비스는 오퍼레이션들의 집합으로 구성되어 있고, 각 오퍼레이션은 다수의 입출력 매개변수(parameter)들로 이루어져 있다. 따라서 직관적으로 이들 입출력 매개변수들이 일치한다면 두 개의 오퍼레이션은 매치된다고 할 수 있다. 본 논문의 주된 아이디어는 매개변수들 사이의 숨은 시멘틱 개념을 찾아내어 온톨로지를 학습(learning)하고, 확장된 키워드 탐색 방법과 온톨로지 학습 방법을 혼합 사용하여 보다 지능적인 웹 서비스 매칭을 수행하는 것이다.

지금까지 본 연구와 유사한 많은 연구가 수행되어졌다. [11]은 웹서비스들을 분류하기 위해 머신 러닝(machine learning) 방법을 제안하였다. 그렇지만 이 논문에서는 WSDL로부터 추출된 모든 텀(term)들을 단지 단어의 백(bag)으로만 취급할 뿐 계층관계와 같은 시멘틱 개념은 없다. [5]는 구문분석 방법과 온톨로지 활용 방법을 제안하였다. 시멘틱 개념이 없는 구문분석 방법에 온톨로지 활용방법을 추가로 제안했지만 전문가에 의해 수작업으로 구축되는 온톨로지는 시간이 많이 소비되고 날로 증가하는 새로운 웹 서비스들에 대한 확정성에 문제가 많다. [9]는 연관성이 높은 웹 서비스 매개변수들을 같은 개념으로 묶는 클러스터링 메카니즘을 제안하였다. 이 방법에서는 재현율은 향상시킬 수 있으나 이와 비례하여 원하지 않은 결과도 증가하므로 정확률의 향상은 기대하기 어렵다.

본 논문의 구성은 다음과 같다. 2장에서 웹 서비스 구조를 간단히 살펴보고 3장에서 구문 분석 방법을 설명한다. 4장에서 온톨로지 학습 방법을

기술하고 5장에서 클러스터-온톨로지 웹 서비스 매칭 알고리즘을 제안한다. 6장에서 구현 및 실험 분석을 수행하고 7장에서 결론을 내린다.

2. 웹 서비스의 구조

본 논문의 웹 서비스 매칭 알고리즘 이점을 알아보기 위해 먼저 웹 서비스의 구조에 대해 간단히 살펴본다. 각각의 웹 서비스는 그 기능과 인터페이스를 기술하고 있는 하나의 WSDL 파일과 연관되어 있다. 일반적으로 하나의 웹 서비스는 UDDI 비즈니스 레지스트리(Registry) 내에 자신의 WSDL 파일과 서비스에 대한 간단한 설명을 등록함으로 공개된다. 각 웹 서비스는 그림 1과 같이 오퍼레이션들의 집합으로 구성되어 있고, 각 오퍼레이션은 다수의 입출력 매개변수들로 이루어져 있다. 따라서 UDDI로부터 다음과 같은 정보를 얻을 수 있다.

- 웹 서비스 정보: 하나의 웹 서비스는 WSDL 파일 내에 서비스의 이름과 이에 대한 텍스트 설명이 기술된다. 그리고 UDDI 레지스트리 내에 비즈니스에 관한 정보가 입력된다.

- 오퍼레이션 정보: 각 오퍼레이션은 WSDL 파일 내에서 오퍼레이션에 대한 이름과 이에 대한 텍스트 설명이 기술된다.

- 입출력 정보: 어떤 오퍼레이션의 입출력은 각각 매개변수들의 집합으로 구성되어 있다. WSDL 파일에서는 각 매개변수에 대한 이름과 데이터 타입이 기술되어 있고, 데이터 타입은 복합(complex) 타입이 사용될 수도 있다.

Web Service(1)
W1: QueryCustomerInfo
Operation1: CustomerInfo
Input: CustomerName, State
Output: CustomerID
Operation2: QueryCompany
Input: CustomerAddress, ZipCode
Output: CompanyCode

Web Service(2)
W2: CheckClientInfo
Operation1: ClientInformation
Input: ClientName, Province
Output: ClientIdentification
Operation2: CheckCountry
Input: PersonAddress, PostalCode
Output: CountryCode

(그림 1) 웹 서비스의 예

그림 1에서 W1과 W2의 Operation1은 입출력 매개변수들에 대해 토큰화(tokenization), 단어 확장, 그리고 시소러스(thesaurus)에 의한 동의어를 적용하면 두 오퍼레이션은 동일한 것임을 알 수 있다. 예를 들면, CustomerID는 Customer와 ID로 분리되고, ID는 Identification으로 확장되며, Customer와 Client, State와 Province는 동의어로 처리된다. 그러나 Operation2는 동의어 사용만으로 이들 간의 유사성을 발견할 수 없다.

W1, W2의 Operation2는 실제적으로 입력은 같은 개념으로 매치되어야 하고, 출력은 다르게 판단되어야 한다. 하지만 입력도 Customer와 Person이 동의어가 아니므로 다르게 취급된다. 이런 매치 형태는 단지 동의어만 사용하여 결정할 수 없고, 이들 간의 상관관계(relationship)를 해석하기 위한 온톨로지 정보가 필요하다. 즉 Person과 Customer는 동일한 개념(예, equivalentClass(Customer, Person))으로 취급되어야 한다. 한편, 출력 부분에서는 CompanyCode는 Company에 관한 내용이고 CountryCode는 Country에 관한 내용으로 판단되어(예, isProperty(CompanyCode, Company), isProperty

(CountryCode, Country)) 이들은 매치되지 말아야 한다.

위의 보기에서 입출력 매개변수를 대표하는 단어를 토큰화하고 동의어를 적용하는 구문 분석 방법(syntactic analysis method)은 키워드를 일반화하여 검색 범위를 넓혀주고, 온톨로지 정보의 사용은 상관관계를 표현하여 깊이 있는 탐색을 유도한다. 이러한 두 방법을 결합함으로써 재현율과 정확률 둘 다 향상 시킬 수 있는 기법이 될 수 있다.

3. 구문 분석 방법

오퍼레이션 입출력 매개변수들 간에 유사성을 발견하는 것은 쉬운 일이 아니다. 왜냐하면, 입출력 매개변수 이름은 복합단어, 약어, 개발자의 명명(naming) 습관 등으로 인해 매우 다양해 질 수 있다. 따라서 WordNet[12]과 같은 전자 사전을 바로 적용하기 어렵다. 또한 웹 서비스 오퍼레이션 내에는 일반적으로 매개변수들이 몇 개 존재하지 않으며, 이에 대한 충분한 설명도 거의 제공하고 있지 않다. 따라서 단어 빈도수를 기반으로 하는 TF/IDF[13]와 같은 전통적인 IR 기법들은 잘 적용될 수 없다.

따라서 보다 효율적인 웹 서비스 탐색을 위해서는 (1) 오퍼레이션 입출력 매개변수들은 일반적으로 다수의 단어가 연결된 복합단어로 이루어져 있으므로(예, ClientName), 이들에 대한 토큰화가 요구된다. (2) 올바른 매치을 발견하기 위해서는 POS(part-of-speech) 뿐만 아니라 시소러스를 통한 단어의 뜻도 고려되어야 한다. (3) 다수의 매개변수들에 대한 몇 개의 부분 일치도 고려해야만 한다. (4) 서비스 명세의 구조나 매개변수 데이터 타입 정보가 고려되어야 한다.

구문 분석 방법에서는 먼저 복합단어로 구성된 매개변수들을 파싱하여 텀으로 분리한다. 그리고 POS와 불용어(stop-word) 필터링이 수행되고, 필요 시 단어 내 약어들이 확장된다. 그 후 동의어 리스트를 발견하기 위해 시소러스가 사용된다. 각 단계에 대한 자세한 내용은 아래와 같다.

복합단어 토큰화: 웹 서비스를 파싱하여 모든 단어들을 뽑아낸 후에 복합단어는 여러 개의 텀으로 분리한다. 예를 들면 ClientName은 Client와 Name으로 나눈다. 단어를 토큰화하기 위해 프로그래머들에 의해 사용되는 일반적인 명명 규칙을 조사할 필요가 있다. 본 연구에서는 빈칸, 하이픈(-), 언더스코어 문자(_), 문자 내 숫자 등과 같은 구분 문자를 사용하여 복합단어를 분리한다.

POS와 불용어 필터링: POS는 접두사 또는 어미 등 어근에 붙어있는 부분을 제거하는 알고리즘으로서 단어를 어근으로 분리하므로 같은 단어이지만 접미사나 어미의 변화에 의해 다른 단어로 인식되는 것을 막을 수 있다. 또한, 미리 만들어진 불용어 리스트에 의해 불용어들이 필터링된다. 본 연구에서 사용되는 불용어는 상용 검색 엔진에서 사용되는 것과 비슷한 and, or, the, is 등이 된다.

약어 확장: 약어(abbreviation)는 완전한 단어로 확장된다. 예를 들면 CustomerInfo는 CustomerInformation으로 확장이 수행된다. 여러 개의 확장 단어 후보가 존재할 경우 복수개의 단어 확장도 가능하다. 따라서 CustPurch는 CustomerPurchase와 CustomaryPurchase 등으로 확장될 것이다.

동의어 탐색: 텀들에 대한 동의어 리스트를 발견하기 위해 WordNet 시소러스를 사용한다. 시소러스란 같은 의미를 갖고 있는 단어이지만 단어의 철자가 다른 경우 이를 해결하기 위해서 제안된 동의어 사전이다. 예를 들어 “bike”와 ″bicycle″은 같은 의미를 갖고 있으나 서로 철자가 틀리므로 다른 단어로 인식될 수 있다.

유사도 계산: 하나의 쿼리와 웹 서비스 저장소로부터 매치되는 임의의 후보 매개변수 쌍을 (Q, S_k)라 하고, 매개변수 Q와 S_k에는 각각 m과 n개의 텀들이 있다고 가정하자.

$$Q = q_1, q_2, \cdots q_i, \cdots q_m$$

$$S_k = s_1, s_2, \cdots s_j, \cdots s_n$$

Q의 q_i와 S_k의 s_j 간의 매치를 고려할 때, 변개변수 Q와 S_k 사이의 유사성은 다음과 같이 계산된다.

$$Similarity(Q, S_k) = \frac{2 * \sum match(q_i, s_j)}{m+n}$$

여기서, $match(q_i, s_j) = \begin{cases} 1 & \text{if } sucess \\ 0 & \text{if } fail \end{cases}$,
$i = 1, 2, \cdots m$, $j = 1, 2, \cdots n$

예를 들면, CustomerCare와 ClientSearch 사이의 유사성은 0.5이다. 왜냐하면 Customer와 Client는 동의어지만, Care와 Search는 매치가 실패하기 때문이다. 매치되는 서비스 개수를 *p*라 할 때, 주어진 쿼리에 대한 최상의 매칭 서비스는

$$BestService = max\{Similarity(Q, S_k)\} \text{ for all } 1 \le k \le p$$

로 주어지고 Similarity(Q, S_k) 값은 우선순위 리스트를 위해 정렬될 수 있다.

이러한 유사성은 State와 Province, CustomerInfo와 ClientInformation과 같은 매개변수 매치는 허용하지만 Customer와 Person과 같은 동일한 개념의 매치는 허용하지 못한다. 이는 구문 분석 방법에서는 텀들 사이의 상관관계는 알 수 없기 때문에 발생되는 현상이다. 이러한 문제들은 다음의 온톨로지 학습 방법(ontology learning method)을 이용하여 해결할 수 있다.

4. 온톨로지 학습 방법

본 연구의 핵심 내용은 웹 서비스 매개변수들에 대해 의미적으로(semantically) 같은 개념들을 묶고(clustering), 각 텀들 간의 계층관계(hierarchy)를 구축하여 텀들 사이에 숨겨져 있는 시멘틱 개념을 활용하는 것이다.

웹 서비스의 오퍼레이션 내에는 일반적으로 매개변수들이 몇 개 존재하지 않기 때문에 기존의 전통적인 클러스터링 알고리즘들은 직접 적용할 수 없다. 왜냐하면, IR 응용에서는 동의어가 동일한 도큐멘트에 발생하는 경향이 높은 반면에, 웹 서비스에서는 하나의 오퍼레이션 내에 같은 입출력 매개변수는 거의 발생되지 않기 때문이다. 따라서 기존의 기법과는 다른 클러스터링 알고리즘의 적용이 요구된다.

매개변수들을 토큰화하여 텀으로 분리한 후, 관련성이 많은 텀들에 대해 클러스터를 형성하면 이 클러스터는 각각의 단어가 아닌 하나의 의미있는 개념을 나타낸다. 이러한 클러스터는 "매개변수들이 동시에 자주 나타난다면, 그것들은 같은 개념을 나타내는 경향이 있다"는 가정 하에 하나의 특별한 연관규칙(association rules)[14, 15]에 따라 만들어 진다.

연관규칙 $\Re$은 조건부와 결과부로 구성되며 텀 t_1이 일어나면 t_2가 일어난다는 의미로 다음과 같이 표현될 수 있다.

$\Re$: $t_1 \rightarrow t_2$

따라서 연관규칙을 탐사하는 것은 적절한 텀 t_1과 t_2를 선택하는 문제로 볼 수 있으며 이를 위해 몇 가지 척도를 고려하고 있다. 우선 규칙 $\Re$에 대한 지지도(support)와 신뢰도(confidence)는 각각 다음과 같이 정의된다.

지지도 = 입출력에 t_1이 나타날 확률

$$= \frac{\| t_1 \text{을 포함하는 } IO \text{의 수} \|}{\| IO \text{의 전체 개수} \|}$$

신뢰도 = 입출력에 t_1이 주어졌을 때, t_2가 나타날 확률

$$= \frac{\| t_1, t_2 \text{를 둘다 포함하는 } IO \text{의 수} \|}{\| t_1 \text{를 포함하는 } IO \text{의 수} \|}$$

여기서 신뢰도가 임계치 δ보다 크면(즉 $t_1 \rightarrow$

t_2(신뢰도>δ)), 텀 t_1과 t_2는 밀접하게 연관되었다고 말할 수 있다. 임계치 δ 값은 기존의 agglomerative 클러스터링 알고리즘[16]으로 구할 수 있는데, 이 알고리즘은 결과적으로 높은 점수(score)를 갖도록 클러스터를 형성하는 것이 목표이다. 이때 높은 점수는 cohesion(한 클러스터 내의 텀들과의 응집력)은 높고, correlation(다른 클러스터 텀들 간의 상호관계)은 낮은 것을 의미한다.

$$\text{Score} = \frac{cohesion}{correlation}$$

$$\text{cohesion} = \frac{\| i \rightarrow j(\text{신뢰도} > \delta)\text{인 개수} \|}{\| C_1 \| \, \| (C_1 - 1) \|}$$

여기서 $i,j \in C_1$, $i \neq j$, C_1은 클러스터

$$\text{correlation} = \frac{\| i \rightarrow j(\text{신뢰도} > \delta)\text{인 개수} \| + \| j \rightarrow i(\text{신뢰도} > \delta)\text{인 개수} \|}{2 \| C_1 \| \, \| C_2 \|}$$

여기서 $i \in C_1$, $j \in C_2$, C_1, C_2는 클러스터

이러한 클러스터링 기법을 이용한 웹 서비스 탐색에서는 단순히 입출력 매개변수 텀들의 빈도수에 의존하는 것이 아니라 각 텀들 간의 상호연관성을 이용해 관련된 단어들 끼리 클러스터링 함으로써 보다 효과적인 웹 서비스의 검색이 가능하게 한다. 그러나 이 기법은 연관성 높은 단어들을 한 클러스터에 묶어서 단지 동일한(equivalent) 개념처럼 취급할 뿐 계층관계에 따라 사용자의 요구사항을 정확하게 표현하는 시멘틱 기능은 제공하지 못하고 있다.

계층관계 온톨로지 활용 기법은 사람들이 단어를 조합하여 복합단어로 된 매개변수를 만들 때 일반적으로 비슷한 패턴을 사용하는 경향이 있다[17, 18]는 관찰로부터 시작한다. 이러한 패턴들은 다음과 같은 형태로 나타난다.

1. 명사(1) + 명사(2)
   (예, CompanyID)

2. 접두사 + 명사(1) + 명사(2)
   (예, virtualAccountID)

3. 형용사 + 명사(1)
   (예, virtualAccount)

4. 명사(1) + 전치사 + 명사(2)
   (예, passwordOfAccount)

5. 동사 + 명사(1)
   (예, enrollAccount)

첫 번째 단계로 각 텀들의 상관관계를 취득하여 그들을 온톨로지에 저장한다. 변환 룰(rule)은 표 1과 같다.

(표 1) 상관관계 변환 룰

룰	패턴	상관관계
	Example	Example
1	명사(1) + 명사(2)	isProperty(매개변수, 명사(1))
	CompanyID	isProperty(CompanyID, Company)
2	접두사 + 명사(1) + 명사(2)	isProperty(매개변수, 접두사 + 명사(1)) subClassOf(매개변수, 명사(2))
	virtualAccount ID	isProperty(virtualAccountID, virtualAccount) subClassOf(virtualAccountID, ID)
3	형용사 + 명사(1)	subClassOf(매개변수, 명사(1))
	virtualAccount	subClassOf(virtualAccount, Account)
4	명사(1) + 전치사 + 명사(2)	subClassOf(매개변수, 명사(2))
	passwordOfAccount	subClassOf(passwordOfAccount, Account)
5	동사 + 명사(1)	subClassOf(매개변수, 명사(1))
	enrollAccount	subClassOf(enrollAccount, Account)

두 개의 온톨로지 개념은 다음 조건이 만족되면 매치된다.

(1) 어떤 개념이 다른 개념의 속성(property)일 경우(예, companyID는 company의 속성)
(2) 어떤 개념이 다른 개념의 자식관계(subclass)인 경우(예, virtualAccount는 Account의 자식관계)

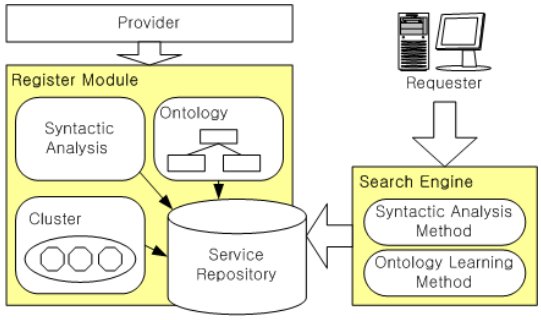
여기서, $Similarity(Q, S_k) = \begin{cases} 1 & \text{if } sucess \\ 0 & \text{if } fail \end{cases}$

위의 변환 룰을 적용함에 따라 관련 없는 개념들 사이의 매치를 피할 수 있다. 예를 들면, companyID와 countryID는 각각 다른 개념의 속성이므로 매치에서 배제된다. 계층관계 온톨로지 학습 기법은 관련 없는 개념들의 매치를 피할 수 있으므로, 매치되는 후보 집합들은 키워드 기반 탐색기법에 의해 생성되는 결과보다 더욱 정확한 매치를 얻을 수 있다.

5. 웹 서비스 매칭 알고리즘

본 논문에서 제안하는 구문 분석 및 온톨로지 학습 방법의 기본적인 원리는 그림 2와 같다. 먼저 사용자는 원하는 웹 서비스를 3장에서 설명된 구문 분석 방법에 의해 웹 서비스 저장소에서 찾는다. 그러나 이 방법에서는 텀들 사이의 상관관계를 알 수 없기 때문에 4장에서 기술된 온톨로지 학습 방법을 추가하여 시멘틱 검색이 가능하도록 한다.

온톨로지 학습 방법은 클러스터링 기법을 사용하여 사용자가 입력한 키워드뿐만 아니라 키워드가 포함된 클러스터 안의 모든 텀들에 대해 검색을 수행한다. 이때 클러스터는 상호 연관성이 큰 단어들로 묶여있다. 다음으로 계층관계 온톨로지 활용 기법에 의해 검색 키워드와 웹 서비스 간에 계층관계 조건이 만족되는지 체크된다. 따라서 검색 키워드와 웹 서비스 문서의 내용이 일치하지 않더라도 의미적으로 같은 웹 서비스를 검색할 수 있고, 검색된 웹 서비스들 중에서도 사용자가 원하지 않는 웹 서비스를 온톨로지를 통해 검색 결과에서 제거할 수 있다.



(그림 2) 구문 분석 및 온톨로지 학습 방법의 원리

본 기법은 우리가 이전에 제안한 "SOA 기반 웹 서비스 조합시스템[19]"을 효율적으로 구현하기 위해 개발되었다. [19]에서 사용자는 먼저 프로세스 디자이너를 사용하여 비즈니스 프로세스를 작성하는데, 웹 서비스 탐색이 요구되는 단계에서 사용자는 서비스 저장소로부터 적합한 후보 서비스들이 발견되도록 쿼리문을 작성한다. 그리고 난 후 탐색 엔진은 이 쿼리를 기반으로 웹 서비스 발견 과정을 수행해야 한다.

웹 서비스 발견 과정을 좀 더 자세히 살펴보면 쿼리 Q의 입력 매개변수를 사용하여 원하는 출력을 산출해 낼 수 있는 웹 서비스들을 찾는다. 이를 위해서 선택되는 웹 서비스는 반드시 쿼리의 출력항목을 포함하고 있어야만 하고, 이 서비스의 입력 매개변수는 쿼리의 입력항목에 포함되어 있어야만 한다. 이러한 과정을 기반으로 클러스터-온톨로지 웹 서비스 매칭 알고리즘을 작성하면 그림 3과 같다. 그림 3에서 Discovery( ) 함수는 쿼리문 Q를 서비스 저장소에 있는 모든 웹 서비스 S들과 비교한다. 이때 본 논문에서 제안한 구문 분석 및 온톨로지 학습 방법인 SynOntoMethod( ) 함수를 적용하게 된다. 이 방법에 의해 만일 매치가 발견되면 그 웹 서비스가 기록되고 우선순위에 의해 정렬된다. SynOntoMethod( ) 함수는 먼저

쿼리문 출력 매개변수 Q.Os를 웹 서비스 출력 매개변수 S.Os와 비교하여 유사도를 계산하고, 매치가 실패하지 않는다면 반대로 웹 서비스 입력 매개변수 S.Is와 쿼리문 입력 매개변수 Q.Is를 비교하여 유사도를 계산한다.

Algorithm: Matching algorithm for web services

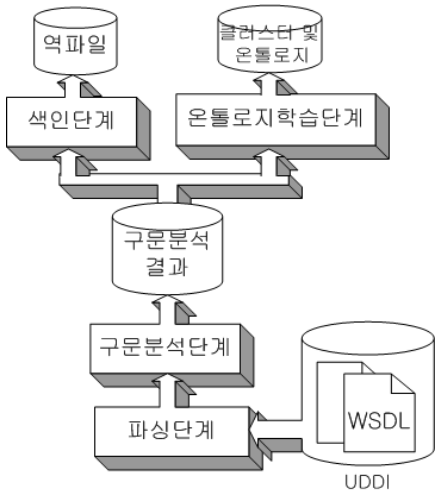
```
Input: query Q
Output: ranked list of matching services

  Discovery(Q) {
       for all S in ServiceRepository
           if SynOntoMethod(Q, S)
             then record.append(S)
           endif
       endfor
       return sorting(record)
  }
  SynOntoMethod(Q, S) {
       outputMatch(Q.Os, S.Os)
       inputMatch(S.Is, Q.Is)
  }
```

(그림 3) 클러스터-온톨로지 웹 서비스 매칭 알고리즘

6. 구현 및 실험 분석

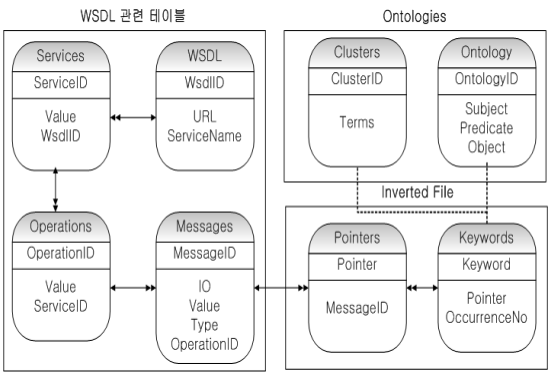
기존의 키워드 기반 탐색 방법은 검색이 필요할 때마다 UDDI로부터 정보를 가져와야 한다. 이러한 경우에 검색의 속도가 저하된다. 본 논문에서는 서비스 저장소에 검색에 필요한 부분을 별도의 테이블로 저장함으로써 검색의 효율을 높인다. 이러한 테이블들은 WSDL 파일을 파싱함으로써 얻어올 수 있으며 이들은 역파일(inverted file)을 생성하기 위한 정보로도 활용된다.



(그림 4) 데이터베이스 구축과정

데이터베이스 구축과정은 그림 4와 같이 파싱단계, 구문분석 단계, 색인단계, 온톨로지 학습단계로 나누어진다. 파싱단계에서는 WSDL의 Services, Operations, Messages(Inputs/Outputs)을 파싱하여 서비스와 오퍼레이션 이름, 입출력 매개변수 등을 넘겨준다. 구문분석 단계에서는 파싱된 입출력 매개변수들을 정형화하기 위하여 토큰화, POS 및 불용어 필터링, 약어 확장, 그리고 동의어를 처리한 후 구문분석 결과 테이블을 생성한다. 이후 색인단계와 온톨로지 학습단계로 나뉘어 지는데 색인단계에서는 검색 대상의 키워드 위치를 식별하도록 하며 단어의 빈도수 정보를 가지는 역파일을 생성한다. 온톨로지 학습 단계에서는 의미적으로 같은 개념들을 클러스터링하고 각 텀들 간의 계층관계를 취득하여 클러스터 및 온톨로지 테이블을 구축한다.

전체적인 데이터베이스 결과는 그림 5에서 WSDL 데이터의 E-R 모델로 표현하였다. 각각의 WSDL 파일은 여러 Service를 가질 수 있고, Service는 여러 Operation, Operation은 여러 Message를 가질 수 있다. 역파일은 각 Keyword에 대해 여러 Pointer를 가질 수 있고, 각 Pointer는 MessageID와 매핑된다. 클러스터와 온톨로지 테이블은 Keyword 테이블과 연관되어 온톨로지 학습 방법에 활용된다.

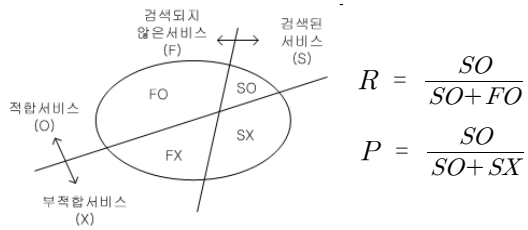


(그림 5) WSDL 데이터의 E-R 모델

실험 분석의 목적은 본 논문에서 제안하는 클러스터-온톨로지 웹 서비스 매칭 알고리즘의 우수성을 보이는 것이다. 전통적인 웹 서비스 탐색 방법은 UDDI를 이용한 키워드 기반 검색만 지원하고 있다. 이러한 키워드 기반 탐색 방법에 비해 클러스터-온톨로지 웹 서비스 매칭 방법이 얼마나 효율적으로 수행되는지 두 방법을 비교·분석한다.

평가 방법은 정보검색에서 가장 보편적으로 사용되고 있는 재현율과 정확률을 사용하고, 추가로 역화일에 의한 검색시간 효과를 측정한다. 재현율은 사용자의 질의에 적합한 웹 서비스를 얼마나 검색했는지를 나타내며, 정확률은 검색 결과 중에서 사용자 질의에 적합한 웹 서비스들이 얼마나 되는지를 나타낸다. 재현율과 정확률은 모두 높을수록 성능이 좋다고 할 수 있다. 하지만 이들은 서로 반비례의 관계가 있어 한쪽을 높이면 다른 한쪽이 내려가는 것이 보통이다. 재현율(recall) R과 정확률(precision) P는 그림 6과 같이 계산된다.

실험 분석을 위해 기존의 웹 서비스 저장소인 xmethods.net에서 508개의 WSDL 파일을 다운로드 받았다. 그림 4의 데이터베이스 구축과정을 거쳐 그림 5와 같은 테이블에 정보를 저장하였다. 만일 웹 사용자가 'ZipCode'라는 입력 매개변수를 사용해 City와 State를 찾는 웹 서비스를 검색한다면 이 데이터베이스에 존재하는 적합한 웹 서비스의 수는 508개 중 18개이다. ZipCode와 관련된 클러스터를 형성하면 {city, state, country, post, zip, code}와 같이 되고 계층관계 온톨로지는 isProperty(ZipCode, Zip)가 된다.

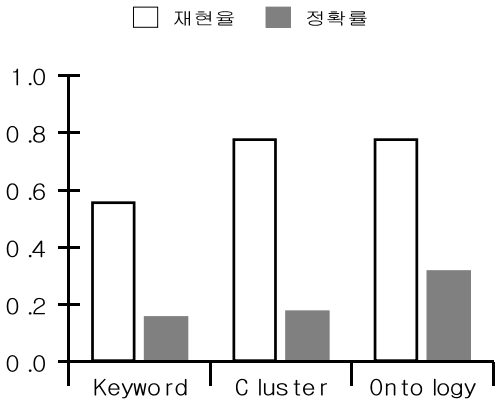


(그림 6) 재현율(R)과 정확률(P)의 계산

그림 7은 기존의 키워드 기반 검색 방법과 비교하여 클러스터-온톨로지 검색 방법의 성능 향상을 보여주고 있다. 클러스터-온톨로지 검색 방법은 클러스터링 기법과 계층관계 온톨로지 활용 기법의 효과를 분석하기 위해 이 두 기법을 각각 분리하여 재현율과 정확률을 측정하였다. 키워드 기반 검색 방법은 예측된 바와 같이 재현율과 정확률 모두 가장 낮다. 클러스터링 기법만 추가 되었을 경우에는 예를 들면 ZipCode 관련 클러스터 {city, state, country, post, zip, code}에 있는 텀들을 포함한 모든 웹 서비스들을 검색한다. 따라서 단순 키워드 검색만 사용했을 때보다 재현율은 개선되었으나 정확률은 거의 비슷하다. 이는 검색된 결과에서 적합한 웹 서비스들이 증가한 만큼 비례적으로 부적합한 서비스들도 증가하기 때문이다. 마지막으로 계층관계 온톨로지 활용 기법을 추가한 경우에는 ZipCode는 Zip의 속성이므로 Code만 포함하고 있는 웹 서비스들은 매치에서 배제된다. 따라서 검색 결과 중 부적합한 웹 서비스들이 줄어들어 정확률이 상승하게 된다. 실험 분석 결과 본 논문에서 제안한 클러스터-온톨로지 검색 방법(3번째 그래프)이 기존의 키워드 검색 방법에 비해 재현율, 정확률 각각 22%와 16% 개선된 것을 알 수 있다.

또한, 본 제안 방법은 사전에 오프-라인(off-line) 방식으로 역화일을 생성함으로써 검색 시간을 상당히 줄일 수 있다. 역화일을 사용함으로써 탐색

의 복잡성도 상당히 줄어들었고 검색 시간도 기존의 UDDI 기반 탐색 방법에 비해 평균 90~95%의 시간 감소를 이룰 수 있었다. 사전 오프-라인으로 역화일을 구축하는데 걸리는 시간은 3.20GHz Intel CPU에서 2분 이내에 처리 되었으며 쿼리는 순식간에 처리되었다.



(그림 7) 클러스터와 온톨로지를 적용한 실험 결과

한편, 기존의 전문가에 의한 수작업 온톨로지를 구축하는 방법과 본 논문의 자동구축 방법과의 차이는 수작업이 어렵기 때문에 다른 대안으로써 자동구축 방법을 제안했다는 의미에서 상호간의 성능 비교는 별 의미가 없다. 전문가들은 상황에 따라 다를 수 있겠지만 웹 서비스 제공자에 대한 정보, 서비스 처리를 위한 기능에 대한 정보, 서비스 특성에 관한 정보, 그리고 계층관계에 따른 시멘틱 정보를 자세히 기록할 수 있다. 이러한 정보들이 충실히 기술만 될 수 있다면 지금까지 제안된 시멘틱 매칭 알고리즘들[4, 5, 19]은 훌륭히 적용될 수 있을 것이다. 문제는 이러한 온톨로지 정보가 현재로서는 거의 존재하지 않으며 이들의 구축도 쉬운 일이 아닌데 있다. 일반적으로 웹 서비스 제공자는 자동적으로 생성되는 WSDL 파일과 간단한 웹 서비스 설명만 첨부하여 웹 서비스를 공개한다. 본 연구에서는 이러한 한정된 정보를 가지고 항목 간의 숨어 있는 시멘틱 정보를 찾아내어 온톨로지를 구축하고 이를 이용한 시멘틱 매칭 알고리즘을 적용하는 것이다.

7. 결 론

본 논문에서는 구문 분석 방법과 온톨로지 학습 방법을 혼합 사용한 보다 지능적인 웹 서비스 매칭 알고리즘을 제안하였다. 본 논문의 핵심 내용은 웹 서비스 매개변수들에 대해 의미적으로 같은 개념들을 클러스터링으로 묶고, 각 텀들 간의 계층관계 온톨로지를 구축하여 텀들 사이의 숨겨져 있는 시멘틱 개념을 활용하는 것이다. 따라서 검색 키워드와 웹 서비스 문서의 내용이 일치하지 않더라도 의미적으로 같은 웹 서비스를 검색할 수 있고, 검색된 웹 서비스들 중에서도 사용자가 원하지 않는 웹 서비스를 온톨로지를 통해 검색 결과에서 제거할 수 있다. 제안된 방법은 실험 분석을 통해 기존의 키워드 기반 검색 방법보다 성능이 우수함을 보였다. 향후 연구 과제로는 더욱 다양한 실험 분석을 통해 제안된 알고리즘의 완전성과 신뢰성을 보장하는 것이다.

참 고 문 헌

[1] http://www.uddi.org
[2] http://www.xmethods.com
[3] http://www.webservicelist.com
[4] M. Paolucci, T. Kawamura, T. R. Payne and K. Sycara, "Semantic Matching of Web Services Capabilities," Proceedings of the 1st International Semantic Web Conference(ISWC), 2002
[5] T. Syeda-Mahmood, G. Shah, R. Akkiraju, A. Lvan, and R. Goodwin, "Searching Service Repositories by Combining Semantic and Ontological Matching," Proceedings of IEEE International Conference on Web Services(ICWS), 2005
[6] R. Akkiraju, J. Farrell, J. Miller, M. Nagarajan,

M. Schmidt, A. Sheth and K. Verma, "Web Service Semantics - WSDL-S," http://www.w3.org/Submission/WSDL-S/, 2005

[7] OWL Services Coalition, "OWL-S: Semantic Markup for Web Services," OWL-S White Paper, http://www.daml.org/services/owl-s/1.0/owl-s.pdf, 2004

[8] M. Sabou, C. Wroe, C. Goble, and H. Stuckenschmidt, "Learning Domain Ontologies for Semantic Web Service Descriptions," Journal of Web Semantics, 3(4), 2005

[9] X. Dong, A. Halevy, J. Madhavan, E. Nemes, and J. Zhang, "Similarity Search for Web Services," In Proceedings of VLDB, 2004

[10] D. Shou and C. Chi, "A Clustering-based Approach for Assisting Semantic Web Service Retrieval," IEEE International Conference on Web Services, 2008

[11] A. Hess and N. Kushmerick, "Learning to Attach Semantic Metadata to Web Services," Proceedings of ISWC2003, 2003

[12] G. Miller and C. Fellbaum, "WordNet," http://wordnet.princeton.edu

[13] G. Salton and C. Buckley, "Term Weighting Approaches in Automatic Text Retrieval," Information Processing and Management, 24(4), 1988

[14] R. Agrawal, T. Imielinski and A. Swami, "Mining Association Rules between Sets of Items in Large Databases," Proceedings of the 1993 ACM-SIGMOD International Conference Management of Data, 1993

[15] D. Braga, A. Campi, S. Ceri, M. Klemetinen, and P. Lanzi, "Discovering Interesting Information in XML Data with Association Rules," SAC, Proceedings of the 2003 ACM Symposium on Applied Computing Table of Contents, pp. 450-454, 2003

[16] D. Kaufman and P. Rousseeuw, "Finding Groups in Data: An Introduction to Cluster Analysis," John Wiley & Sons, New York, 1990

[17] H. Guo, A. Ivan, R. Akkiraju, and R. Goodwin, "Learning Ontologies to Improve the Quality of Automatic Web Service Matching," Proceedings of IEEE International Conference on Web Services(ICWS), 2007

[18] P. Velardi, P. Fabriani, M. Missikoff, "Using Text Processing Techniques to Automatically Enrich a Domain Ontology," Proceedings of the ACM International Conference on Formal Ontology in Information Systems, 2001

[19] 이용주, "반자동 웹 서비스 조합을 위한 WS-BPEL과 OWL-S의 융합 시스템," 정보처리학회논문지D 제15-D권 제4호, pp. 569-580, 2008

◐ 저 자 소 개 ◑

이 용 주

1983년 울산대학교 산업공학과(학사)
1985년 한국과학기술원 산업공학과 정보검색전공(석사)
1997년 한국과학기술원 정보및통신공학과 컴퓨터공학전공(박사)
1985년~1989년 KIST 시스템공학연구소 연구원
1989년~1994년 삼보컴퓨터 근무
1998년~2007년 상주대학교 컴퓨터공학과 부교수
2008년~현재 경북대학교 이공대학 컴퓨터정보학부 교수
관심분야 : 웹 데이터베이스, 정보검색, 공간 데이터베이스
E-mail : yongju@knu.ac.kr