DOI: 10.3745/KIPSTD.2011.18D.2.133

온톨로지 학습에 의한 유사 웹 서비스 오퍼레이션 발견 방법

이 용 주†

요 약

시맨틱 웹 서비스 기술의 성공을 보장하기 위해서는 품질 좋은 온톨로지의 사용이 필수적이다. 하지만 온톨로지 사용의 중요성에도 불구하고 현재 웹 서비스를 위한 온톨로지는 거의 존재하지 않으며 이들의 구축도 쉬운 일이 아니다. 이러한 문제는 오늘날 웹 서비스의 확산과 발전을 가로막는 큰 저해 요인이 되고 있다. 본 논문에서는 웹 서비스를 개발할 때 자동으로 생성되는 WSDL 문서만 가지고 항목 간 숨어있는 시맨틱 정보를 찾아내어 온톨로지를 자동 구축하고, 이를 이용한 유사 웹 서비스 오퍼레이션 발견 방법을 제안한다. 핵심 내용은 WSDL 입출력 항목들로부터 의미적으로 같은 개념들을 묶고, 각 항목들 간의 계층관계를 형성하여 자동적으로 시맨틱 온톨로지를 구축한다. 그리고 새로운 유사도 측정 방법을 통해 우선순위별 유사 오퍼레이션을 발견하며, 발견된 오퍼레이션들 중 가장 적합한 오퍼레이션을 선택하여 웹 서비스 조합에 직접 활용할 수 있는 웹 서비스 오퍼레이션 검색 시스템을 구현한다.

키워드 : 온톨로지 학습, 유사 오퍼레이션 발견, 매개변수 클러스터링, 매개변수 패턴 분석, 유사도 측정

Discovery Methods of Similar Web Service Operations by Learning Ontologies

Yong-Ju Lee†

ABSTRACT

To ensure the successful employment of semantic web services, it is essential that they rely on the use of high quality ontologies. However, building such ontologies is difficult and costly, thus hampering web service deployment. This study automatically builds ontologies from WSDL documents and their underlying semantics, and presents discovery methods of similar web service operations using these ontologies. The key ingredient is techniques that cluster parameters in the collection of web services into semantically meaningful concepts, and capture the hierarchical relationships between the words contained in the tag. We implement an operation retrieval system for web services. This system finds out a ranked set of similar operations using a novel similarity measurement method, and selects the most optimal operation which satisfies user's requirements. It can be directly used for the web services composition.

Keywords : Learning Ontology, Similar Operations Discovery, Parameter Clustering, Parameter Pattern Analysis, Similarity Measurement

1. 서 론

최근에 웹(web)은 정적인 웹 문서에서 지적이고 동적인 데이터 통합 환경으로 발전되고 있으며 이러한 진전을 위해 두 가지 혁신적인 변화가 제시되고 있다. 첫 번째로 웹 서비스(web services) 기술은 웹 표준을 통해 다양한 플랫폼과 서로 다른 언어로 개발된 소프트웨어 컴포넌트들에 대하여 동일한 엑세스를 허용함으로써 이기종 소프트웨어 컴포넌트들 간의 상호운용성(interoperability)을 증가시키고, 다양한 소프트웨어 모듈 사이의 재사용(reuse)과 조합(composition)을 가능하게 한다. 두 번째로 시맨틱 웹(semantic web) 기술은 정보의 의미를 개념(concept)으로 정의하고 개념 간의 관계성을 표현함으로써 정보를 공유시키고, 웹 상의 정보를 수집하고 처리하기 위해 더 이상 인간의 전적인 개입이 요구되지 않는다. 따라서 각종 자동화된 에이전트를 통해 정확한 정보 검색, 새로운 지식의 생성, 최적의 서비스 제공 등을 가능하게 한다.

현재 웹 서비스 기술의 주된 단점은 서비스의 발견 및 조합이 아직까지 수작업으로 수행되고 있는 것이다[1]. 이는 오늘날 웹 서비스들이 수없이 증가되고 있는 상황에서 큰 부담이 되고 있다. 따라서 최근의 여러 연구들[2, 3, 4]에서

※ 이 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임(No. 2010-0008303).
† 종신회원 : 경북대학교 이공대학 컴퓨터정보학부 교수
논문접수 : 2010년 10월 13일
수 정 일 : 1차 2010년 12월 7일
심사완료 : 2010년 12월 14일

시맨틱 웹 기술을 이용하여 자동적인 웹 서비스 발견 및 조합을 실현하기 위해 기계 가독형으로 웹 서비스 기능을 묘사할 수 있는 메카니즘, 즉 온톨로지(ontology)를 활용하고 있다. 이에 따라 호환 가능한 웹 서비스들 간의 실시간 발견 및 조합이 가능하다. 하지만 이러한 시맨틱 웹 서비스 기술의 성공을 보장하기 위해서는 품질 좋은 온톨로지의 사용이 필수적이다.

최근 온톨로지 사용의 중요성에도 불구하고 현재 웹 서비스를 위한 온톨로지는 거의 존재하지 않으며 이들의 구축도 쉬운 일이 아니다[5]. 왜냐하면 현재 이들 온톨로지는 대부분 전문가의 수작업으로 구축되고 있으며 시간 및 인적 제약 때문에 실용적인 온톨로지를 구축하기가 쉽지 않다. 또한 현시점에서 매일 수백에서 수천 건씩 증가되고 있는 웹 서비스 전체에 대해 온톨로지를 수동 구축한다는 것은 거의 불가능하게 보이며, 이러한 문제는 오늘날 웹 서비스의 확산과 발전을 가로막는 큰 저해 요인이 되고 있다.

본 논문에서는 웹 서비스를 개발할 때 자동 생성되는 WSDL(Web Services Description Language) 문서만 가지고 항목 간 숨어있는 시맨틱 정보를 찾아내어 온톨로지를 자동 구축하고, 이를 이용한 유사 웹 서비스 오퍼레이션(operation) 발견 방법을 제안한다. 본 연구의 핵심 내용은 WSDL 입출력 항목들로부터 의미적으로 같은 개념들을 묶고(clustering), 각 항목들 간의 계층관계(hierarchical relationship)를 형성하여 자동적으로 시맨틱 온톨로지를 구축한다. 그리고 새로운 유사도 측정 방법을 통해 우선순위별 유사 오퍼레이션을 발견하며, 발견된 오퍼레이션들 중 가장 적합한 오퍼레이션을 선택하여 웹 서비스 조합에 직접 적용될 수 있는 웹 서비스 검색 시스템을 구현하는 것이다.

본 논문의 구성은 다음과 같다. 2장에서 본 논문과 관련된 연구들을 분석하고 3장에서 웹 서비스 구조 및 검색 유형을 간단히 살펴본다. 4장에서 온톨로지 학습 방법을 제안하고 5장에서 웹 서비스 오퍼레이션 발견 방법을 기술한다. 그리고 6장에서 실험 분석을 수행하고 7장에서 결론을 내린다.

2. 관련 연구

온톨로지 구축 방법에 관한 연구는 크게 두 가지 방향으로 추진되고 있다. 첫째는 전문가의 수작업으로 웹 서비스 저장소(registry)에 추가적인 시맨틱 정보를 주석처리(annotation)하여 온톨로지를 구축하는 방법이다(예, OWL-S, WSMO, SAWSDL). [6, 7]은 OWL-S 기반 시맨틱 웹 서비스 조합 시스템을 제안하였고, [8]은 WSMO 기반 시맨틱 SOA(Service Oriented Architectures) 프레임워크를 제안하였다. 그리고 [9]는 WSDL에 시맨틱 개념을 첨가한 SAWSDL을 제안하였다. 하지만 OWL-S, WSMO, 그리고 SAWSDL을 구축하기 위해서는 수작업에 의한 전문가의 주석처리가 필요하다. [10]은 OWL-S 기반 온톨로지에 전통적 구문(syntactic) 분석 방법을 첨가한 방식을 제안하였다. 비록 자동화된 구문 분석 방법은 제안하였으나 시맨틱 개념이 없어 추가로 수작업으로 온톨로지를 구축해야 하므로, 이 방법 또한 시간이 많이 소비되고 날로 증가하는 새로운 웹 서비스들에 대한 확장성에 문제가 많다.

두 번째 접근방법은 수작업으로 온톨로지를 구축하기가 어렵기 때문에 온톨로지 학습 방법에 의해 자동 구축하는 방법이다. [11]은 웹서비스들을 자동 분류하기 위해 Naive Bayes와 SVM 머신 러닝(machine learning) 방법을 제안하였다. 그렇지만 이 논문에서는 WSDL로부터 추출된 모든 용어(term)들을 단지 단어의 백(bag)으로만 취급할 뿐 계층관계와 같은 시맨틱 개념은 없다. [12]는 연관성이 높은 웹 서비스 매개변수들을 같은 개념으로 묶는 클러스터링(clustering) 메카니즘을 제안하였다. 이 방법에서는 재현율은 향상시킬 수 있으나 이와 비례하여 원하지 않은 결과도 증가하므로 정확률의 향상은 기대하기 어렵다. [13]에서는 온톨로지 학습을 위해 프로그램 소스, 도큐멘테이션, UML 다이어그램 등 다양한 소스를 고려하였고, 자연언어처리 기법을 이용한 웹 서비스 온톨로지 학습 프레임워크를 제안하였다. 그러나 이 논문에서는 WSDL은 취급하지 않았고 웹 서비스 매칭에 관한 연구는 없다. 최근에 [14]는 WSDL로부터 추출된 단어를 백으로 취급하지 않고 이들 간의 상관관계를 고려한 웹 서비스 매칭 방법을 제안하였으나, 이 방법은 실제 웹 서비스 환경에 적용했을 때 단지 오퍼레이션 1:1 부분 매칭에만 적용될 수 있다. 실세계 응용에서 웹 서비스 메타데이터는 이보다 더욱 복잡하고 다양한 구조로 구성되어 있다.

3. 웹 서비스 구조 및 검색 유형

기존의 전문가에 의한 수작업으로 온톨로지를 구축하는 방법에서는 보통 온톨로지 구축 툴(tool)을 사용하여 온톨로지를 만든다. 이때 온톨로지의 내용이 충실히 기술만 될 수 있다면 지금까지 제안된 시맨틱 매칭 알고리즘들[6, 7, 15]은 훌륭히 적용될 수 있을 것이다. 문제는 이러한 온톨로지 정보가 현재로서는 거의 존재하지 않으며 이들의 구축도 쉬운 일이 아니다. 일반적으로 웹 서비스 제공자는 UDDI(Universal Description Discovery and Integrity) 저장소 내에 자신의 웹 서비스와 관련된 WSDL 파일, 그리고 그 서비스에 대한 간단한 설명을 등록함으로 공개한다. WSDL에서 하나의 웹 서비스는 오퍼레이션들의 집합으로 구성되어 있고, 각 오퍼레이션은 다수의 입출력 매개변수(parameter)들로 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 저장소 내에 연락처나 카테고리와 같은 비즈니스에 관한 정보도 입력된다.
- 오퍼레이션 정보: 각 오퍼레이션은 WSDL 파일 내에서

오퍼레이션에 대한 이름과 이에 대한 텍스트 설명이 기술된다.
- 매개변수 정보: 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.

위와 같은 UDDI 정보로부터 온톨로지 구축에 필요한 정보를 대부분 취득할 수 있다. 다만, WSDL 파일에서는 구문 정보만 있을 뿐 온톨로지에서 요구되는 동일(equivalent) 또는 계층(hierarchy) 관계와 같은 객체지향 클래스에 대한 정보(예, Customer equivalent Person 또는 Sedan subClassOf Car)가 없다. 따라서 본 연구에서는 이러한 WSDL 구문 정보만 가지고 항목 간의 숨어 있는 시맨틱 정보를 찾아내어 온톨로지를 자동 구축하고 이를 이용한 유사 웹 서비스 오퍼레이션 발견 방법을 구현한다.

먼저 본 연구의 목표를 설명하기 위해 하나의 전형적인 시나리오를 고려한다. 일반적으로 웹 서비스 사용자는 UDDI 저장소에서 키워드 입력에 의해 웹 서비스 탐색을 시작한다. 탐색된 결과 중에서 자세히 검토될 필요가 있는 웹 서비스들을 결정한다. 그 후 이들 웹 서비스 내에 있는 오퍼레이션들을 살펴보고 입출력 매개변수들을 조사한다. 만일 사용하고 있는 검색 엔진에서 “try it” 기능을 제공하고 있다면 입력 값을 기입하고 그 서비스를 실행해 볼 수 있다. 이 단계에서 어떤 이유로 그 서비스가 사용자의 요구에 적합하지 못할 수도 있다. 간혹 사용자는 이런 번거로운 작업들을 다른 웹 서비스 모두에 대해 반복적으로 처리하는 것은 원치 않을 것이다. 따라서 본 연구의 목표는 사용자가 마치 웹 서비스를 사전에 자세히 살펴본 것 같이 원하는 순서대로 정렬된 좀 더 지능적인 탐색 방법을 제공하는 것이다. 사용자들이 시도하기를 원하는 웹 서비스에 대한 검색 유형은 다음과 같이 세 종류로 요약될 수 있다.

- 유사 오퍼레이션 발견: 하나의 오퍼레이션이 주어졌을 때 이와 유사한 오퍼레이션들을 찾는다. 직관적으로 비슷한 입력과 비슷한 출력을 가지고 있고 이들 입출력 사이의 관계가 비슷하다면 두 오퍼레이션은 유사하다고 할 수 있다. 참고로 본 논문에서는 유사한 웹 서비스 발견은 고려하지 않는다. 이는 너무 포괄적이어서 별로 사용되지 않기 때문이다.
- 유사 입출력 발견: 비슷한 입력(또는 출력)을 가진 오퍼레이션을 찾는다. 예를 들면, 오퍼레이션1(입력: ZipCode, 출력: City)과 오퍼레이션2(입력: PostalCode, 출력: Weather)는 비슷한 입력을 가진 오퍼레이션이다.
- 오퍼레이션 조합: 하나의 오퍼레이션으로는 다양하고 복잡한 사용자 요구사항을 충분히 만족시켜줄 수 없을 때 오퍼레이션 조합을 통해 새롭고 유용한 솔루션을 얻는다. 예를 들면, 여행 예약 서비스는 항공기, 숙박, 그리고 관광 등 다수의 솔루션들이 적절히 결합되어야 한다.

본 논문에서는 위의 요구사항을 쉽게 해결할 수 있도록 다음과 같은 두 가지 문제에 초점을 맞춘다. (1) 오퍼레이션 매칭: 오퍼레이션에 관한 질의 템플릿(query template)이 주어졌을 때 이와 유사한 오퍼레이션 리스트를 반환한다. 참고로 유사 입출력 발견 문제는 이 매칭 방법에 포함될 수 있다. (2) 오퍼레이션 조합: 자동화된 메카니즘으로 오퍼레이션 워크플로우(workflow)를 생성한다. 워크플로우의 시작 단계에서부터 점차적으로 하나씩 오퍼레이션이 추가된다. 각 단계에서 사용자는 유사 오퍼레이션을 탐색하고 그가 의도한 바를 잘 이룰 수 있는 가장 적합한 오퍼레이션을 선택한다.

4. 온톨로지 학습 방법

본 연구의 핵심 내용은 웹 서비스 매개변수들에 대해 의미적으로(semantically) 같은 개념들을 묶고, 각 단어들 간의 계층관계를 구축하여 단어들 사이에 숨겨져 있는 시맨틱 개념을 활용하는 것이다.

입출력 매개변수들 간에 유사성을 발견하는 것은 쉬운 일이 아니다. 왜냐하면, 매개변수 이름은 복합단어, 약어, 개발자의 명명(naming) 습관 등으로 인해 매우 다양해 질 수 있다. 따라서 WordNet[16]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(Term Frequency/Inverse Document Frequency) [17]와 같은 전통적인 정보 검색(information retrieval) 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 토큰화(tokenization)가 요구된다. (2) 올바른 매치를 발견하기 위해서는 스테밍(stemming) 뿐만 아니라 시소러스(thesaurus)를 통한 단어의 뜻도 고려되어야 한다. (3) 다수의 매개변수들에 대한 몇 개의 부분 일치도 고려해야만 한다. (4) 기존의 전통적인 클러스터링 알고리즘과는 다른 새로운 클러스터링 알고리즘의 적용이 요구된다.

4.1 매개변수 클러스터링

매개변수를 토큰화하여 용어들로 분리한 후, 관련성이 많은 용어들에 대해 클러스터(cluster)를 형성하면 이 클러스터는 각각의 단어가 아닌 하나의 의미 있는 개념을 나타낸다. 이러한 클러스터는 “매개변수들이 동시에 자주 나타난다면, 그것들은 같은 개념을 나타내는 경향이 있다”는 가정 하에 하나의 특별한 연관규칙(association rules)[18, 19]에 따라 만들어 진다.

연관규칙 $\mathcal{R}$은 조건부와 결과부로 구성되며 용어 X가 일어나면 Y가 일어난다는 의미로 $\mathcal{R}$:X⇒Y와 같이 표현될 수 있다. 따라서 연관규칙을 탐사하는 것은 적절한 용어 X와 Y를 선택하는 문제로 볼 수 있으며 이를 위해 몇 가지 척

도를 고려하고 있다. 우선, 용어 X와 규칙 $\mathcal{R}$에 대한 지지도(support)는,

$$\text{supp(X)} = \frac{\| IO(X) \|}{\| IO \|}$$

$$\text{supp}(\mathcal{R}) = \text{supp}(\text{X} \cup \text{Y}) = \frac{\| IO(X \cup Y) \|}{\| IO \|}$$

여기서, $\| IO \|$는 IO(Input/Output)의 전체 개수, $\| IO(X) \|$는 X를 포함하는 IO의 개수, $\| IO(X \cup Y) \|$는 X와 Y를 동시에 포함하는 IO의 개수를 나타낸다. 신뢰도(confidence)는 입출력에 X가 주어졌을 때, Y가 동시에 나타날 확률로 계산되며, 신뢰도가 큰 규칙일수록 그 의미가 크다고 하겠다.

$$\text{conf}(\mathcal{R}) = \frac{supp(X \cup Y)}{supp(X)} = \frac{\| IO(X \cup Y) \|}{\| IO(X) \|}$$

여기서 신뢰도가 임계치 δ보다 크면(즉, conf($\mathcal{R}$:X⇒Y)> δ) 용어 X와 Y는 밀접하게 연관되었다고 말할 수 있다. 임계치 δ 값은 관련성이 많은 용어들과 관련 없는 용어들을 잘 분리할 수 있는 최적의 값을 수작업으로 선택하는데, 이 값은 적용되는 실험 데이터에 따라 다른 값들을 가질 수 있다. 그러나 이것은 데이터셋(dataset) 구축에 비해 그다지 번거로운 작업은 아니다. 이 알고리즘은 결과적으로 높은 점수(score)를 갖도록 클러스터를 형성하는 것이 목표이다. 이때 높은 점수는 cohesion(한 클러스터 내의 용어들과의 응집력)은 높고, correlation(다른 클러스터 용어들 간의 상호관계)은 낮은 것을 의미한다.

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

여기서, cohesion과 correlation은 다음 식의 평균값

$$\text{cohesion}_{i,j} = \frac{\| conf(R: i \Rightarrow j) > \delta) \|}{\| C_1 \| \| (C_1 - 1) \|}$$

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

$$\text{correlation}_{i,j} = \frac{\| conf(R: i \Rightarrow j) > \delta \| + \| conf(R: j \Rightarrow i) > \delta \|}{2 \| C_1 \| \| C_2 \|}$$

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

연관규칙 탐사 문제는 Apriori[20] 알고리즘을 사용하여 효율적으로 처리될 수 있다. 그러나 클러스터링은 기존의 알고리즘을 바로 적용할 수가 없다. 본 논문에서는 계층적 결합 클러스터링(hierarchical agglomerative clustering) 알고리즘[21]을 확장하였는데, 기존의 클러스터링 알고리즘들은 대부분 저차원 데이터를 위해 설계되어 있어서, 데이터의 차원이 매우 큰(예, 수천 개의 상이한 용어들을 가지는 텍스트 문서) 경우 차원의 저주(curse of dimensionality) 문제가 대두된다. 이러한 문제를 해결하는 하나의 방법으로써 빈발 패턴 기반 클러스터링(frequent pattern based clustering) 방법이 적용될 수 있다. 빈발 패턴 기반 클러스터링의 아이디어는 발견된 빈발 패턴이 또한 클러스터링을 의미하기도 한다는 것이다. 즉 고차원 용어 벡터 공간을 클러스터링하는 대신에 저차원의 용어 집합만을 클러스터 후보로 간주하는 것이다. 따라서 본 논문에서 제안되는 알고리즘은 Apriori 알고리즘의 연관규칙 결과에서 먼저 신뢰도를 내림차순으로 정렬한 다음 지지도를 내림차순으로 정렬한다. 그리고 각 단계에서 가장 높은 점수의 규칙을 선택한다. 이때 그 규칙을 구성하는 조건부와 결과부가 서로 다른 클러스터에 속하면 본 알고리즘은 그 클러스터를 결합한다. 최종적으로 우리의 클러스터링 알고리즘은 높은 점수를 갖도록 형성하는 것이 목표이므로 클러스터 점수는 위의 식과 같은 cohesion/correlation 방식으로 계산된다.

매개변수 클러스터링의 처리 과정을 확인하기 위해 본 논문에서는 먼저 기존의 웹 서비스 저장소인 xmethods.net[22]에서 WSDL 파일을 다운로드 받아 사전 작업 처리과정을 거쳐 본 알고리즘이 수행될 수 있는 데이터 파일을 준비하였다. xmethods.net에서는 약 600여개의 WSDL 파일이 존재하고 있으나 본 실험에서는 모든 파일을 이용하지 않고 zip, weather, address에 관한 50개의 파일에 대해서만 알고리즘을 적용시켰다. 이는 너무 관련 없는 웹 서비스들 간의 클러스터링은 결과가 거의 없거나 별 의미가 없을 수 있기 때문이다. 본 연구에서는 실험 파일에 대해 최저 지지도와 최저 신뢰도를 변경시켜가며 클러스터 결과를 도출 분석하였는데, 그 결과 최저 지지도 10%, 최저 신뢰도 80%에서 관련성 있는 용어들과 관련 없는 용어들을 잘 분리할 수 있는 하나의 적절한 임계치가 될 수 있음을 알 수 있었다. 이로부터 추출된 연관 규칙들은 code,city⇒zip state⇒city zip,state⇒city area⇒zip area⇒city zip,area⇒city city,area ⇒zip area⇒zip,city country⇒city zip⇒code zip,city⇒state 가 되었고, 최종 클러스터 결과는 {zip, city, area, state}, {zip, code}, {country, city}의 3개 최적 클러스터가 생성되었다.

이러한 클러스터링 기법을 이용한 웹 서비스 탐색에서는 단순히 입출력 매개변수 용어들의 빈도수에 의존하는 것이 아니라 각 용어들 간의 상호연관성을 이용해 관련된 단어들끼리 클러스터링 함으로써 보다 효과적인 웹 서비스 탐색을 가능하게 한다. 그러나 이 기법은 연관성 높은 단어들을 한 클러스터에 묶어서 단지 동일한(equivalent) 개념처럼 취급할 뿐 계층관계에 따라 사용자의 요구사항을 정확하게 표현하는 객체지향 클래스 기능은 제공하지 못하고 있다. 따라서 본 논문에서는 다음의 매개변수 패턴(pattern) 분석 기법을 추가로 제안한다.

4.2 매개변수 패턴 분석

본 기법의 주된 아이디어는 매개변수 내의 각 단어들 사이의 상관관계를 취득하고, 비교되는 단어들이 서로 유사하고

상관관계가 조건에 일치한다면 그 비교를 매치하는 것이다. 이러한 구축 기법은 “사람들이 단어를 조합하여 복합단어로 된 매개변수를 만들 때 일반적으로 비슷한 패턴을 사용한다[14, 23]”는 관찰로부터 시작한다. 일반적으로 사람들이 어떤 한 개념을 표현할 때 다양한 방법들이 있을 수 있지만 공간적인 제약 하에서는 비슷한 패턴을 사용하는 경향이 있다.

이러한 패턴들을 조사하기 위해 4.1에서 만든 실험 데이터에 POS(part-of-speech) 형태소 분석기[1]를 적용시킨 결과 <표 1>과 같은 형태를 취하였다. 분석 결과 단지 하나의 토큰으로 구성된 매개변수(예, City)가 38%로 가장 많았고, 명사+명사(37%), 명사+명사+명사(14%), 동사+명사(6%), 명사+전치사+명사(3%), 형용사+명사(1%), 그리고 기타(1%) 순으로 나타났다.

〈표 1〉 복합단어 매개변수의 패턴

규칙	패턴	출현수(%)	예
1	Noun1+Noun2	470 (37%)	CompanyID
2	Noun1+Noun2+Noun3	175 (14%)	TelephoneAreaCode
3	Verb+Noun	70 (6%)	enrollAccount
4	Noun1+Preposition+Noun2	40 (3%)	PasswordOfAccount
5	Adjective+Noun	15 (1%)	virtualAccount

시맨틱 웹의 논리적 기반이 되는 온톨로지는 “특정분야에 대한 공유할 수 있는 개념들의 정형화된 기술”로 정의되며, 개념은 그것의 특징을 설명하기 위한 속성(property)과 개념 사이의 상-하위(subclass) 관계로 표현된다. 따라서 본 절에서는 첫 번째 단계로 각 단어들 간의 속성 및 상-하위 관계를 취득하여 그들을 온톨로지에 저장한다. <표 1>과 같은 관찰로부터 WSDL 파일의 매개변수에 대한 본 논문에서의 변환 규칙(rule)은 다음과 같다.

규칙1(Noun1+Noun2): 매개변수가 명사1의 속성(property)이 된다.
- Parameter propertyOf Noun1
- 예, CompanyID → CompanyID propertyOf Company 형태가 된다.

규칙2(Noun1+Noun2+Noun3): 매개변수가 명사1의 속성이 된다.
- Parameter propertyOf Noun1
- 예, TelephoneAreaCode → TelephoneAreaCode propertyOf Telephone

규칙3(Verb+Noun): 매개변수가 명사의 자식관계(subclass)가 된다.
- Parameter subClassOf Noun
- 예, enrollAccount → enrollAccount subClassOf Account

규칙4(Noun1+Preposition+Noun2): 매개변수가 명사2의 속성이 된다.
- Parameter propertyOf Noun2
- 예, passwordOfAccount → passwordOfAccount propertyOf Account

규칙5(Adjective+Noun): 매개변수가 명사의 자식관계가 된다.
- Parameter subClassOf Noun
- 예, virtualAccount → virtualAccount subClassOf Account

위와 같은 규칙을 사용하여 온톨로지가 구축되고 나면, 다음 단계는 질의문에 의해 두 개념 간 매칭을 시키는 것이다. 즉 두 개의 온톨로지 개념이 다음 조건을 만족하면 매치된다.

- 어떤 개념이 다른 개념의 속성일 경우 (예, CompanyID propertyOf Company)
- 어떤 개념이 다른 개념의 자식관계인 경우 (예, virtual Account subClassOf Account)

이러한 조건을 적용함에 따라 관련 없는 개념들 사이의 매치를 피할 수 있다. 예를 들면, CompanyID와 CountryID는 각각 다른 개념의 속성이므로 매치에서 배제된다. 매개변수 패턴 분석 기법은 관련 없는 개념들의 매치를 피할 수 있으므로, 매치되는 후보 집합들은 클러스터링 기법에 의해 생성되는 결과보다 더욱 정확한 매치를 얻을 수 있다.

5. 웹 서비스 오퍼레이션 발견 방법

5.1 유사 오퍼레이션 발견

동적으로 웹 서비스를 발견하기 위해 본 절에서는 오퍼레이션의 유사도가 어떻게 측정되는지 기술한다. 하나의 오퍼레이션 템플릿이 주어졌을 때, 이 템플릿과 저장소 내의 임의의 웹 서비스 오퍼레이션과의 유사도는 오퍼레이션 텍스트 설명이나 입출력 매개변수 유사도에 깊은 관련이 있다.

질의 템플릿 Q는 3장의 웹 서비스 구조로부터 다음과 같이 5개의 튜플(tuple)로 표현될 수 있다.

$$Q = < \text{name, description, Is, Os, category} >$$

여기서, Is는 입력항목, Os는 출력항목을 표시하고 category는 NAICS와 UNSPSC와 같은 분류시스템을 사용한다. 각 항목들을 특성별로 분류하면 name과 description은 텍스트로 표현되어 있으며, Is와 Os는 온톨로지로 구축되어 있고 category는 코드 분류 체제를 사용하고 있다.

유사도를 측정하기에 앞서 정확성을 향상시키기 위해 각 단어들에 대해 다음과 같은 전처리 과정을 수행한다. 먼저 복합단어로 구성된 매개변수들을 파싱(parsing)하여 용어들로 분리하고 스테밍과 불용어(stop-word) 필터링을 수행한다. 그 다음 단어 내 약어들이 확장되고 동의어 리스트를 발견하기 위해 시소러스가 사용된다.

1) 본 논문에서는 CRFTagger[24] 형태소 분석기를 사용하였다.

본 연구에서 유사 오퍼레이션 발견 과정은 다단계 매칭 방법에 의해 수행된다. 각 매칭 단계에서 질의 템플릿과 저장소 내의 웹 서비스 사이의 일치 여부가 판단되는데, 이를 위해 먼저 사용자로부터 작성된 질의 템플릿을 어떠한 순서로 수행할지에 대한 질의 계획을 세운다. 질의 계획을 세울 때는 질의 처리의 효율을 고려해야 하는데, 매칭 요소의 특성상 매칭 속도가 빠른 항목이 있고 매칭 속도가 느린 항목이 있으며, 코드별 1차 필터링(filtering) 단계와 제약조건에 따른 2차 정제(refinement) 단계가 있기 때문이다. 따라서 계산량이 많은 온톨로지 매칭은 다른 항목들보다 늦게 처리하고, 분류 코드나 텍스트의 경우 적은 계산량으로 높은 질의 처리 효율을 가지고 있으므로 다른 항목들보다 먼저 처리한다.

코드 일치 방법: 질의 템플릿에 카테고리(category) 정보가 있으면 이는 분류코드로 이루어져 있으므로 코드 일치(exact) 방법을 사용하여 매칭을 수행한다. 코드 일치 방법은 분류 코드에 속하는 서비스들만 빠르고 쉽게 필터링하여 다음 매칭 단계가 효율적으로 수행되도록 탐색 범위를 한정시킨다.

이름 및 설명 유사도: 오퍼레이션의 이름과 설명은 텍스트로 구성되어 있다. 텍스트에 대해 완전 일치 방법을 사용하면 너무 제한된 탐색이 이루어질 수 있기 때문에 좀 더 유연한 매칭을 위해 정보검색 분야에서 널리 사용되고 있는 TF/IDF 방법을 사용하여 매칭의 유사도를 구한다.

TF/IDF 방법에서 각 단어의 가중치(term weight) w_{ik}는 해당 문서에서 각 단어의 빈도(TF)와 역문헌빈도(IDF)의 곱으로 나타낸다[17]. 즉,

$$w_{ik} = \frac{F_{ik}\log(N/n_k)}{\sqrt{\sum_{k=1}^{t}(F_{ik})^2[\log(N/n_k)^2}}$$

여기서, F_{ik}는 i번째 문서에서 단어 k의 빈도이며, N은 전체 문서의 수, n_k는 k가 출현한 문서의 수이다. 이를 이용한 질의와 문서 간의 유사도 similar(Q, D_i) 측정은 코싸인(cosine) 방법으로 측정된다. 즉,

$$similar(Q,D_i) = \frac{\sum_{j=1}^{t} w_{qj} \times w_{ij}}{\sqrt{\sum_{j=1}^{t}(w_{qj})^2 \times \sum_{j=1}^{t}(w_{ij})^2}}$$

여기서, $Q = w_{q1}, w_{q2}, \ldots w_{qt}$이고 $D_i = w_{i1}, w_{i2}, \ldots. w_{it}$이다.

위 식을 이용하여 오퍼레이션 이름과 설명 유사도를 각각 계산할 수 있으며, 전체적으로 질의 템플릿 Q와 저장소에 있는 임의의 오퍼레이션 O간의 텍스트 유사도 Txt Similar(Text Similarity)는 다음과 같이 계산된다.

$$TxtSimilar = \frac{\alpha(NamSimilar) + \beta(DesSimilar)}{\alpha+\beta}$$

여기서, NamSimilar은 이름 유사도, DesSimilar 설명 유사도이며, α와 β는 각각의 가중치이다. 결과 값은 0과 1 사이의 실수값을 리턴한다.

매개변수 유사도: 입출력 매개변수에 대해서는 4장에서 서술된 바와 같이 클러스터링과 패턴 분석 기법이 적용된다. 패턴 분석 기법은 정제 단계에서 활용되므로 따로 설명하고 여기서는 단지 클러스터링 기법만 고려한다. 오퍼레이션 내에는 매개변수들이 몇 개 존재하지 않으며, 하나의 오퍼레이션 내에 같은 매개변수는 거의 발생되지 않기 때문에 기존의 TF/IDF 방식을 그대로 적용하기는 어렵다. 따라서 관련성이 많은 용어들을 개념으로 클러스터링하고 이런 용어들을 단어의 백으로 재배치한 후 TF/IDF 방법을 적용한다.

하나의 질의와 저장소로부터 매치되는 임의의 후보 오퍼레이션 쌍을 (Q, O)라 하고, 매개변수 Q와 O에는 각각 m과 n개의 매개변수들이 있다고 가정하자.

$$Q = q_1, q_2, \cdots q_i, \cdots q_m$$
$$O = o_1, o_2, \cdots o_j, \cdots o_n$$

Q의 q_i와 O의 o_j 간의 매치를 고려할 때, 클러스터링 기반 매개변수 유사도 CstSimilar(Cluster Similarity)는 다음과 같이 매개변수 쌍들의 평균값으로 계산된다.

$$CstSimilar = \frac{2 \times \sum_{i=1}^{m} PairSimilar(q_i, o_j)}{m+n}$$

여기서, $PairSimilar(q_i, o_j) = \max\{similar(q_i, o_j)\}$ *for all* $1 \le j \le n$, $i = 1, 2, \cdots m$ 이다.

마지막으로 전체적인 유사도는 다음과 같이 TxtSimilar와 CstSimilar의 제곱근으로 계산된다.

$$Similarity = \sqrt{TxtSimilar \times CstSimilar}$$

매개변수 패턴 분석: 위의 유사도 측정 방법에서는 매개변수 패턴 분석 기법은 고려하지 않았으나, 이를 통한 온톨로지를 활용함으로써 검색 결과의 정확률을 향상시킬 수 있다. 매개변수 유사도를 계산할 때 계층관계 온톨로지 개념에 위배되는 매개변수 쌍들을 배제하여 사용자가 만족할 수 있는 품질 좋은 것만 정제한다. 즉 매치되는 매개변수 쌍들 중에서 4.2절에서 설명한 패턴 상관관계를 조사하여 부모 클래스가 일치하지 않을 경우 이 쌍들을 검색 결과에서 배제한다.

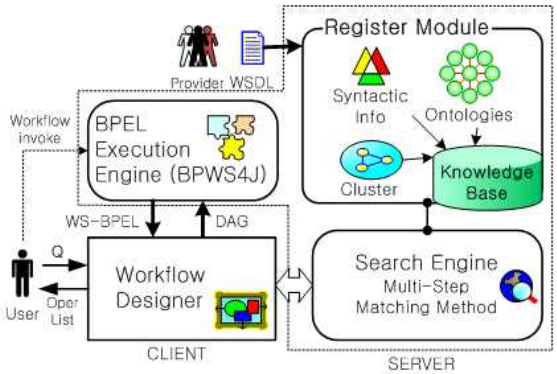
$$PairSimilar(q_i, o_j) = PairSimilar(q_i, o_j) \times match(q_i, o_j)$$

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

이상의 유사도 측정 방법을 적용한 Similarity 값은 우선순위 리스트를 위해 정렬될 수 있다.

5.2 오퍼레이션 조합

최근에 실시간적으로 요구가 있는 즉시 서비스들을 조합하는 동적 웹 서비스 조합은 가장 큰 관심사 중 하나이다. 특히, 현존하는 하나의 웹 서비스로는 사용자의 요구사항을 만족시켜줄 수가 없을 때 그러한 요구를 만족시키기 위해 몇 개의 서비스들을 동적으로 결합해야 하는 문제가 큰 이슈로 부각되고 있다. 본 논문에서는 비즈니스 분야에서 성공적으로 활용되고 있는 워크플로우 기법을 적용하여 동적 웹 서비스 조합 시스템을 구현하였다. 전체 시스템 구조는 (그림 1)과 같다.



(그림 1) 동적 웹 서비스 조합 시스템의 구조

본 연구에서는 동적 웹 서비스 조합 시스템을 구현하기 위해 순차적 목표 지향 접근방법을 제안한다. 워크플로우의 시작 단계에서부터 점차적으로 하나씩 새로운 오퍼레이션들이 추가된다. 각 단계에서 새로운 오퍼레이션을 워크플로우에 첨가시킬 때 사용자는 먼저 질의 템플릿 Q를 생성한다. 일단 Q가 생성되면 이는 웹 서비스 탐색 모듈로 보내지고, Q와 기존 웹 서비스들 간의 유사도 측정에 따라 우선순위가 매겨진 오퍼레이션들이 반환된다. 이때 사용자는 그가 의도한 바를 가장 잘 이룰 수 있는 적합한 오퍼레이션을 하나 선택한다.

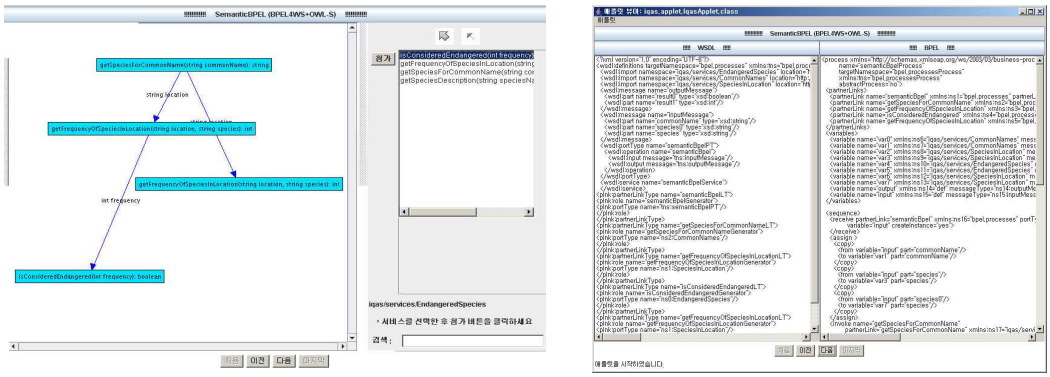
일단 워크플로우가 완성되고 나면 워크플로우 실행 엔진에 의해 이들 프로세스들이 자동으로 수행된다. 웹 서비스 기반 워크플로우는 전통적인 워크플로우와는 달리 인터넷상에 수 많은 공급자와 수요자가 관련되어 있기 때문에 여러 사이트에 분산 저장되어 있는 이질적 서비스들을 통합·관리할 필요성이 있다. 이러한 필요성에 따라 최근엔 WS-BPEL, WSCI, BPSS 등 여러 가지 웹 서비스 조합언어가 발표되었으나 이들 언어들은 시맨틱 개념을 지원해 주지 못하고 수동으로 작업을 해야 하는 단점이 있다. 따라서 본 연구에서는 5.1과 같은 온톨로지 기반 유사 오퍼레이션 발견 기법을 제안하였고, 완성된 워크플로우는 WS-BPEL (Web Services Business Process Execution Language) 스펙으로 변환되어 BPEL 실행 엔진[2]에서 오퍼레이션 조합이 실행되는 시스템을 개발하였다.

워크플로우를 표현하는 가장 자유스러운 방법은 사이클이 없는 방향성 그래프(DAG: Directed Acycle Graph) 방식이다. 여기서 정점(vertex)은 태스크를 표현하고 간선(edge)은 데이터 흐름을 나타내며 이 그래프는 DAG=(V, E), 즉 정점들의 집합 V와 간선들의 집합 E로 구성된다. 구현된 디자이너는 “drag & drop” 방식으로 워크플로우를 작성함으로써 사용자에게 고수준의 인터페이스를 제공한다. 이러한 인터페이스를 구현하기 위해 클라이언트 측 프로그래밍은 자바 애플릿(applets)을 사용하는데, 애플릿은 사용자가 웹 브라우저로 웹 사이트를 접속할 때 사용자의 PC로 프로그램이 자동 다운로드된 후 실행되기 때문에 서버의 로드를 줄여줄 수 있다. 하나의 문제는 애플릿은 클라이언트 측 샌드박스(sandbox) 환경에서 작동되기 때문에 파일을 읽을 때 극히 제한된 권한을 가진다. 따라서 WSDL 파일의 읽기/쓰기 작업은 모두 서버 측에서 수행된다. 이를 위해 서블릿(servlets)은 애플릿과 동일한 codebase에 존재해야 한다. 예를 들면, 애플릿이 ‘http://somehost:port/path/applet’에서 발견된다면 서블릿은 ‘http://somehost:port/path/servlet’에서 발견된다.

워크플로우 디자이너는 마법사처럼 만들어져 있다. 여러 판넬에 의해 다음/이전 화면을 보여줄 수 있으며 이러한 워크플로우 디자이너 진행 과정은 (그림 2)에서 보여주고 있다. 사용자로부터 질의 템플릿이 주어지면 5.1에서 설명된 유사 오퍼레이션 발견 기법을 적용하여 이용 가능한 오퍼레이션 리스트를 보여주고, 이들 중 가장 적합한 오퍼레이션을 선택한다. 이러한 방식으로 점차적으로 하나씩 새로운 오퍼레이션들이 첨가되고 나면 최종적으로 워크플로우가 완성된다. 비록 WS-BPEL 스펙에는 다양한 프로세스 제어 구성자들을 명시하고 있지만 본 구현에서는 단지 연속적 태스크 수행 및 웹 서비스 호출 기능만 구현되었으며 while과 if 문과 같은 흐름 제어를 처리하기 위해서는 추가 개발이 필요하지만 본 구현만으로도 DAG 표현은 충분히 지원될 수 있다.

클라이언트 측에서 DAG가 일단 완성되고 나면 이를 서버 쪽에 전송하고 서버에서는 DAG를 WS-BPEL 스펙으로 변환한 후 BPEL 실행 엔진에 배치(deploy)시켜야 한다. 이를 위해 먼저 클라이언트 프로그램에서는 완성된 DAG를 binary 형태로 서블릿에 전달한다. 서블릿은 binary 형태로 된 그래프를 받아서 그래프 파서(parser)를 호출한다. 그래프 파서는 DAG를 파싱하여 관련된 WSDL과 WS-BPEL 문서를 자동으로 생성하고 난 다음 서버 쪽의 BPEL 배치함수를 부른다. 일단 워크플로우가 BPEL 실행 엔진에 배치되고 나면 그 워크플로우의 수행은 실행 엔진의 호출 방법에 따르면 된다. 예를 들면, JSP(Java Server Page)로 구현된 사용자 인터페이스를 통해 클라이언트 측에서 워크플로우를 실행하거나 본인의 OS 환경에 맞는 스크립트를 사용하면 된다.

2) 본 논문에서는 공개 소프트웨어인 IBM BPWS4J 실행 엔진을 사용하였다.



(그림 2) 워크플로우 디자이너 실행 과정

6. 실험 분석

실험 분석의 목적은 본 논문에서 제안하는 온톨로지 학습과 유사 오퍼레이션 발견 방법의 우수성을 보이는 것이다. 전통적인 웹 서비스 탐색 방법은 UDDI를 이용한 키워드 기반 검색만 지원하고 있다. 이러한 키워드 기반 검색 방법에 비해 온톨로지 학습에 의한 유사 오퍼레이션 발견 방법이 얼마나 효율적으로 수행되는지 이들 두 방법을 비교·분석하였다.

실험은 MS Window Server 2003 버전에 Apache Tomcat 5.5와 Axis 1.3이 설치된 3.20GHz Intel CPU에서 수행되었다. 본 논문에서 제안하는 온톨로지 학습과 유사 오퍼레이션 발견 방법은 자바 언어(JDK1.5)로 구현되었으며, 기존에 이미 구현되어 배포되고 있는 오픈 소스 알고리즘들을 적극 활용하였다. 토큰화, 불용어 필터링 등 데이터 전처리 과정은 OpenNLP[25]를 사용하였고, POS 형태소 분석기는 CRFTagger[24]를 이용하였다. 연관규칙 및 클러스터링 알고리즘은 각각 최신의 Apriori-T[26]와 ClusterLib[21] 알고리즘을 수정·확장하였다. 그리고 웹 서비스 조합을 위한 워크플로우 실행은 IBM BPWS4J 엔진에서 수행되었으며, 워크플로우 디자이너 구현을 위한 그래픽 툴은 opengraph graphing 팩키지를 사용하였다.

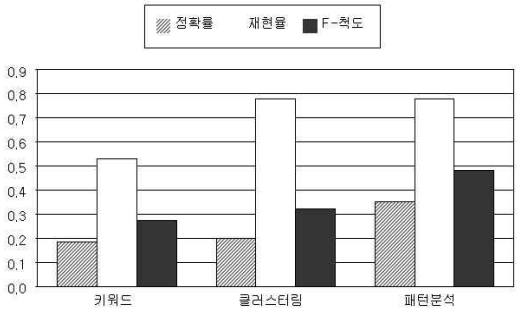
평가 방법은 정보 검색에서 가장 보편적으로 활용되고 있는 정확률, 재현율, 그리고 F-척도를 사용한다. 정확률은 검색 결과 중에서 사용자 질의에 적합한 웹 서비스들이 얼마나 되는지를 나타내며, 재현율은 사용자의 질의에 적합한 웹 서비스를 얼마나 검색했는지를 나타낸다. 재현율과 정확률은 모두 높을수록 성능이 좋다고 할 수 있으나, 이들은 서로 반비례의 관계가 있어 한쪽을 높이면 다른 한쪽이 내려가는 것이 보통이다. F-척도는 정확률과 재현율을 대체하는 하나의 척도로써 정확률과 재현율의 가중치 조화 평균이다. 정확률(precision) P와 재현율(recall) R, 그리고 F-척도(F-measure) F는 다음과 같이 계산된다.

$$P = \frac{C}{C+I} \qquad R = \frac{C}{C+M} \qquad F = \frac{2PR}{P+R}$$

여기서, 검색된 서비스들 중 적합 서비스 수는 C(correct), 부적합 서비스 수는 I(incorrect), 검색되지 않은 서비스들 중 적합 서비스 수는 M(missing)으로 표시되며, 적합 서비스는 실험 데이터를 작성하는 전문가에 의해 체크되는데 이 수는 C+M과 같다.

첫 번째 실험은 4.1에서 만든 실험 데이터를 이용해 zip, weather, address에 관한 질의를 수행하여 온톨로지 학습 방법의 성능을 분석한다. 예를 들어, 웹 사용자가 ZipCode라는 입력 매개변수를 사용해 City와 State를 출력하는 오퍼레이션들을 검색한다면 ZipCode와 관련된 클러스터는 {zip, city, area, state}와 {code, zip}와 같이 되고, City와 State는 각각 {city, zip, area, state, country}와 {state, zip, city, area}와 같이 된다. 또한 이와 관련된 계층관계 온톨로지는 ZipCode propertyOf Zip이 이용된다.

(그림 3)는 기존의 키워드 기반 검색 방법과 비교하여 온톨로지 학습 방법의 성능 향상을 보여주고 있다. 온톨로지 학습 방법은 매개변수 클러스터링과 매개변수 패턴 분석으로 구성되어 있는데 이들의 효과를 분석하기 위해 두 기법을 각각 분리하여 정확률, 재현율, 그리고 F-척도를 측정하였다. 키워드 기반 검색 방법은 예측된 바와 같이 정확률, 재현율, F-척도 모두 가장 낮다. 클러스터링 기법만 추가 되었을 경우에는 단순 키워드 검색만 사용했을 때보다 재현율은 개선되었으나 정확률은 거의 비슷하다. 이는 검색된 결과에서 적합한 오퍼레이션들이 증가한 만큼 비례적으로 부적합한 서비스들도 증가하기 때문이다. 예를 들면, ZipCode는 관련된 클러스터인 {zip, city, area, state}에 있는 모든 용어들에 대해 오퍼레이션 검색을 수행하기 때문이다. 다음으로 매개변수 패턴 분석 기법을 추가한 경우에는 예를 들어, ZipCode는 Zip의 속성이므로 Code만 포함하고 있는 웹 서비스들은 매치에서 배제된다. 따라서 검색 결과 중 부적합한 웹 서비스들이 줄어들어 정확률이 상승하게 된다. 이에 따라 F-척도도 가장 좋은 결과를 얻게 된다. 실험 분석 결과 본 논문에서 제안한 온톨로지 학습 방법이 기존의 키워드 검색 방법에 비해 정확률, 재현율, F-척도 각각 17%, 25%, 21% 개선된 것을 알 수 있다.



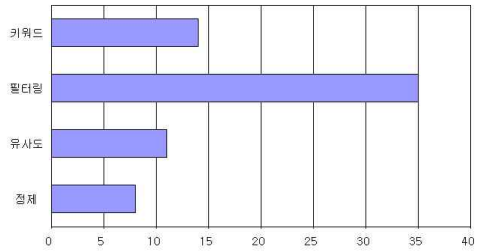
(그림 3) 온톨로지 학습 방법의 성능 분석

두 번째 실험은 제안된 유사 웹 서비스 오퍼레이션 발견 방법의 성능을 분석하는 것이다. 이 방법은 항목 특성별로 ① 필터링(코드 일치 방법) 단계, ② 텍스트(이름 및 설명, 그리고 매개변수) 유사도 측정 단계, ③ 정제(패턴 분석) 단계로 구성되어 있다. 실험은 기존의 키워드 기반 검색 방법과 이들 다단계 매칭 방법에 대해 각각 검색된 웹 오퍼레이션들의 수, 그리고 정확률, 재현율, F-척도를 측정하였다. 첫 번째 실험과 다른 점은 코드와 이름/설명 텍스트 검색이 추가되고, 전체 과정에 대한 단계별 성능을 분석·조사하는 것이다. 이에 대한 결과는 (그림 4)과 (그림 5)와 같다.

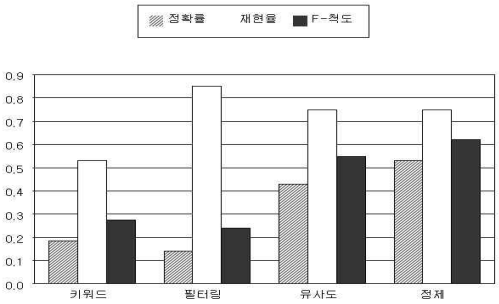
먼저, 필터링 단계에서는 탐색 효율을 향상시키기 위해 NAICS 분류체계에 따라 전체 웹 서비스들 중에서 후보 웹 서비스 집합을 단순히 필터링시키는 것이다. 실험 결과 이 단계에서 검색된 웹 서비스 오퍼레이션들의 수는 키워드 검색보다 훨씬 많다. 이는 xmethods.net에 등록된 웹 서비스들이 아직까지는 다양한 산업분야에 걸쳐서 개발된 것이 아니라 주로 컴퓨터 분야에 치우쳐 있어서 이러한 내용을 위주로 검색을 수행했기 때문이다. 향후 웹 서비스 개발이 다양한 분야로 확산되면 필터링은 큰 효과를 볼 수 있을 것이다. 이 단계에서는 최종 결과 가능성이 있는 후보 서비스들을 임시 저장소에 여과시키는 과정이므로 재현율은 상당히 높은 반면 정확률은 상당히 낮은 경향을 보이고 있다.

다음으로, 필터링 결과를 가지고 이름 및 설명, 그리고 매개변수에 대한 유사도 측정 기법을 수행하여 유사도가 0인 서비스들을 제외시킨 우선순위별 리스트를 구하는 것이다. 유사도가 0인 것은 질의와 전혀 관계없는 서비스들이므로 이들을 제거하면 정확률을 향상시킬 수 있다. 실험 결과 이 단계에서 검색된 오퍼레이션들의 수는 상당히 많이 줄어들었으며, 정확률, 재현율, 그리고 F-척도가 키워드 검색에 비해 상당히 개선된 것을 알 수 있다. 기존의 전통적인 방법들과의 큰 차이점은 이 단계에 온톨로지 학습 방법에서 제안된 매개변수 클러스터링의 성능 향상이 포함되어 있는 점이다.

마지막으로 정제 단계에서는 계층관계 온톨로지 개념에 위배되는 서비스들을 제거하여 품질 좋은 서비스들만 선택하는 단계이므로 정확률 향상이 기대된다. 이는 (그림 3)의 첫 번째 실험에서 이미 증명된 바 있다. 이 단계에 대한 실험 결과 검색된 오퍼레이션들의 수는 예측된 바와 같이 앞 단계보다 약간 줄어들었다. 이로 인해 재현율과 정확률을



(그림 4) 단계별 검색된 오퍼레이션들의 수



(그림 5) 단계별 정확률, 재현율, F-척도

다시 계산한 결과 재현율은 전 단계와 비슷하고 정확률은 조금 상승되었다. 마지막 단계에서 정확률, 재현율, 그리고 F-척도는 기존의 키워드 검색 방법에 비해 각각 35%, 22%, 35% 개선되었다. 이는 첫 번째 실험보다 각각 18%, -3%, 14% 개선된 결과인데, 재현율이 조금 낮게 나타나는 것은 이름 및 설명 유사도 측정에서 M이 약간 발생하기 때문이다. 그러나 본 실험에서는 데이터 특성상 필터링 효과가 그렇게 크지 않았지만 다양한 산업분야에 대한 적용은 1단계 필터링의 효과에 의해 2~3 단계 복잡한 계산식의 실행 속도 면에서 큰 성능 향상을 기대할 수 있다.

7. 결 론

본 논문에서는 온톨로지 학습에 의한 유사 웹 서비스 오퍼레이션 발견 방법을 제안하였다. 본 연구의 핵심 내용은 WSDL 입출력 항목들로부터 의미적으로 같은 개념들을 클러스터링으로 묶고, 각 항목들 간의 계층관계를 형성하여 자동적으로 시맨틱 온톨로지를 구축하는 것이다. 그리고 오퍼레이션 발견 과정은 다단계 매칭 방법에 의해 우선순위별 유사 오퍼레이션들이 발견되며, 발견된 오퍼레이션들 중 가장 적합한 오퍼레이션을 선택·조합하여 동적 웹 서비스 오퍼레이션 조합 시스템을 구현한다. 본 연구에서 제안된 온톨로지 학습 및 유사 웹 서비스 오퍼레이션 발견 방법은 실제 웹 서비스 저장소인 xmethods.net으로부터 WSDL 파일을 다운로드 받아 실험 분석을 수행한 결과 기존의 키워드 기반 검색 방법보다 상당한 성능 향상을 보였다.

본 논문에서 제안된 온톨로지 학습 방법은 적용되는 데이터셋에 따라 최저 지지도 및 최저 신뢰도를 데이터 구축자가 결정해야 하고, 복합단어 매개변수의 패턴이 응용분야에

따라 약간 달라질 수 있다. 실제로 최저 임계값 결정은 매칭 결과에 그다지 민감하지 않고 데이터 구축 작업에 비해 그렇게 번거로운 작업도 아니다. 다만 매개변수 패턴 분석은 다양한 데이터셋에 대한 다양한 변환 규칙들의 생성이 향후 연구 과제로 요구된다. 또한, 본 연구에서 구축되는 온톨로지는 동일(equivalent), 속성(property), 그리고 상-하위(subclass) 관계로만 표현되는데, 향후 이들뿐만 아니라 다른 문헌상에 나와 있는 인과관계 및 다양한 토폴로지(topology) 관계와 같은 관련 연구가 추가로 수행될 필요가 있다. 지금까지 웹 서비스는 주로 SOAP(Simple Object Access Protocol)을 이용해 구현되어 왔으나 최근에는 보다 단순하고 가벼운 REST(Representational State Transfer) 기반의 웹 서비스가 널리 이용되고 있다. 따라서 기존의 SOAP 서비스와 함께 RESTful 웹 서비스도 함께 지원하는 통합 프레임워크에 관한 연구도 필요하다.

참 고 문 헌

[1] Wikipedia, Semantic Web Services, http://en.wikipedia.org/wiki/Semantic_Web_Services.

[2] B. Khalid and B. Marco, "Ontology-Based Description andDiscovery of Business Processes," BPMDS 2009 and EMMSAD 2009, LNBIP 29, pp.85 - 98, 2009.

[3] S. Nathalie, et al., "Service Finder: Realizing Web Service Discovery at Web Scale(First Design of Service-Finder as a Whole)," http://www.service-finder.eu/, June, 2008.

[4] A. P. Sheth, K. Gomadam, and A. Ranabahu, "Semantics Enhanced Services: METEOR-S, SAWSDL and SA-REST," IEEE Data Engineering Bulletin, Vol.31, No.3, pp.8-12, September, 2008.

[5] K. Belhajjame, S. M. Embury, N. W. Paton, R. Stevens, and A. C. Goble, "Automatic Annotations of Semantic Web Services Based on Workflow Definitions," ACM Transactions on the Web 2 (2): pp.1 - 34, April, 2008.

[6] E. Sirin, B. Parsia, and J. Hendler, "Composition-driven Filtering and Selection of Semantic Web Service," In AAAI Spring Symposium on Semantic Web Services, 2004.

[7] K. Verma, K. Gomadam, A. Sheth, J. Miller, and Z. Wu, "The METEOR-S Approach for Configuring and Executing Dynamic Web Processes," Technical Report, LSDIS Lab, University of Georgia, 2005.

[8] T. Vitvar, M. Zaremba, M. Moran, M. Zaremba, and D. Fensel, "SESA: Emerging Technology for Service-Centric Environments," IEEE Software, Vol.24, No.6, pp.56-67, 2007.

[9] J. Kopecky, T. Vitvar, C. Bournez, and J. Jarrell, "SAWSDL: Semantic Annotations for WSDL and XML Schema," IEEE Internet Computing, Vol.11, No.6, pp.60-67, November/December, 2007.

[10] 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.

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

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

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

[14] 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.

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

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

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

[18] 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.

[19] 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.

[20] R. Agrawal and R. Srikant, "Fast Algorithm for Mining Associations Rules," In Proceedings of the 20th VLDB Conference, Santiage, Chile, Sept., 1994.

[21] http://niels.drni.de/s9y/pages/clusterlib.html

[22] http://www.xmethods.net

[23] 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.

[24] http://crftagger.sourceforge.net/

[25] http://opennlp.sourceforge.net/

[26] F. Coenen, "The LUCS-KDD Apriori-T Association Rule Mining Algorithm," http://www.csc.liv.ac.uk/~frans/KDD/Software/Apriori-T/aprioriT.html, 2004.

이 용 주

e-mail : yongju@knu.ac.kr

1985년 한국과학기술원 산업공학과 정보검색전공(석사)

1997년 한국과학기술원 정보및통신공학과 컴퓨터공학전공(박사)

1985년~1989년 시스템공학연구소 연구원

1987년~1988년 일본 IBM TRL 연구소 연구원

1989년~1994년 삼보컴퓨터 근무

1998년~2007년 상주대학교 컴퓨터공학과 부교수

2008년~현 재 경북대학교 이공대학 컴퓨터정보학부 교수

관심분야 : 웹 데이터베이스, 정보검색, 시맨틱 웹 서비스