IE Interfaces
Vol. 19, No. 3, pp. 190-201, September 2006.

웹 서비스 품질 지표 기반 컴포지트 웹 서비스 실행 계획 알고리즘

고종명 · 김창욱†

연세대학교 정보산업공학과

Heuristic Composite Web Service Execution Planning Algorithm Based on Quality of Service Criteria

Jong Myoung Ko · Chang Ouk Kim

Department of Information and Industrial Engineering, Yonsei University, Seoul, 120-749

With the rapid growth in the demand and supply of web services, the search for superior services has become a prominent issue. Furthermore, much emphasis and attention are being placed on composite web services, where individual services are combined together to form a single workflow to satisfy the demand of the customers. In today's era of excessive expansion of web services, this study intends to propose the execution plan architecture for composite web services to accomplish the following three goals. The first goal is to derive a feasible plan which maximizes user satisfaction for composite web services by implementing an execution plan architecture that reflects the Quality of Service criteria. Secondly, this study also aims at analyzing and selecting the indexes that adequately reflects its quality and its nonfunctional property. Finally, this study intends to apply the concept of constraint satisfaction problem and heuristics to reduce the execution duration of the process.

Keyword: composite web services, quality of service, constraint satisfaction problem, heuristic

1. 서 론

웹 서비스는 프로그램, 데이터베이스, 그리고 여러 비즈니스 기능들을 매핑 시켜주는 XML기반의 응용프로그램이다(Newcomer, 2004). 즉, 웹 서비스는 웹을 통하여 출판하고, 검색하고, 요청할 수 있는 정보와 표현방법을 지닌 컴포넌트 기반의 응용 프로그램이라고 정의할 수 있다. 웹 서비스는 적절한 서비스 제공자로부터 다양한 기능들을 실시간으로 찾을 수 있다는 장점이 있다. 또한 네트워크에 쉽게 접속할 수 있는 유비쿼터스 환경으로의 진행 속에서 웹 서비스는 기존의 서비스 형태보다 더 가치가 높은 서비스로 주목 받고 있다. 즉, 웹 서비스로 인한 새로운 수익 구조가 형성되고 새로운 비즈니스의 기회가 창출되고 있다. 최근에는 BSN(Business Service Network), Business process fusion, RTE(Real-time enterprise) 등의 환경에서의 웹 서비스 기반 운영 모형들이 활발히 제시되고 있으며 모바일 장치 기반의 응용모델도 등장하고 있다. 이러한 변화들 속에서 웹 서비스는 소프트웨어 지향적인 상호작용을 기반으로 점차 발전해 나가고 있다. 즉, 서비스 지향적인 아키텍처(SOA: Service-oriented architecture)를 통해 서비스들을 조합하여 제공하는 컴포지트 웹 서비스에 대한 관심이 높아지고 있다.

웹서비스의 발전, 일반적으로 말해 웹의 발전에 있어서는 Tim Berners-Lee를 중심의 월드와이드웹 컨소시엄(W3C)이 주도가 되었다. 그리고 현재는 컴퓨터가 정보의 의미를 이해하고 의미를 조작할 수 있는 시맨틱 웹(Semantic Web)으로 발전되고

†연락저자 : 김창욱 교수, 120-749 서울시 서대문구 신촌동 134 연세대학교 정보산업공학과, Fax : 02-364-7807, Email : kimco@yonsei.ac.kr
2005년 11월 접수, 1회 수정 후 2006년 6월 게재확정.

있다(Liu *et al*., 2004). 시맨틱 웹으로의 진행과 함께 웹서비스의 기술, 발견, 그리고 여러 웹서비스들의 컴포지션(Composition)이 새로운 이슈가 되고 있다. 그리고 기존의 웹서비스 역시 웹서비스간의 상호운영성이 높아지고 컴퓨터가 이해할 수 있는 시맨틱 웹서비스로 발전되고 있다(Cardoso, 2002; Cardoso and Sheth, 2003; Cardoso and Sheth, 2004).

본 논문에서는 새롭게 이슈가 되는 웹서비스 컴포지션을 다루고자 한다. 웹서비스 컴포지션은 사용자가 하나의 서비스를 받기 위해서 필요한 개별 서비스의 조합을 뜻한다. 또한 기존의 워크플로우처럼 프로세스 로직과 라우팅 규칙이 존재한다. 컴포지트 웹서비스(Composite web services)를 처리하기 위해서는 웹서비스 조합 생성 작업과 해당 개별 웹서비스에 적절한 실제 서비스의 매칭 작업이 중요하다. 앞서 제시했듯이 웹서비스 시장의 성장에 따라 웹서비스의 다양화, 복잡화, 혼합화가 진행된다면 그에 따라 사용자의 판단이 힘들 정도로 확장된 웹서비스 시대가 도래될 것이다. 따라서 사용자가 지정한 제약조건들 안에서 가장 알맞은 혹은 가장 우수한 웹서비스 제공자를 찾는 일이 매우 중요하다(Canfora *et al*., 2005; Liu *et al*., 2004; Zeng *et al*., 2004). 본 논문에서는 사용자가 지정한 제약조건들을 반영하여 개별 웹서비스들을 선택하고, 컴포지트 웹서비스 실행 계획을 도출하기 위한 아키텍처를 제안한다.

현재 웹서비스는 제공자의 경우 UDDI(Universal Description, Discovery and Integration)를 이용하여 웹서비스를 온라인 디렉토리에 등록, 광고한다. 사용자는 서비스를 검색해서 제공자와 계약을 맺고 서비스를 사용한다. 사용자의 입장에서는 서비스의 내용과 실제 서비스 이용을 위한 객체간의 통신 규약인 SOAP(Sample Object Access Protocol) 형태의 메시지 전달 정보도 중요하지만 웹서비스의 품질 정보(QoS : Quality of Service)도 중요하다. 오히려 수많은 서비스 안에서 양질의 서비스를 선택하기 위해서는 후자의 정보가 더 중요하다. 따라서 웹서비스의 비함수적인 속성을 고려한 품질 정보에 대한 연구가 중요하다. 그리고 웹서비스의 지속적인 공급과 이용을 위해서는 웹서비스 품질 정보의 지속적인 피드백이 이루어져야한다.

앞서 제시된 제안점들을 바탕으로 본 연구의 목적은 다음의 세 가지로 압축할 수 있다.

1.1 품질 정보가 반영된 웹서비스 실행 계획 아키텍처의 제안

웹서비스 실행 계획이 수행되기 위한 컴포지트 웹서비스의 명세화, 발견, 최적 실행 계획 도출 그리고 계획 실행의 과정들을 기반으로 한 품질 정보가 반영된 웹서비스 실행 계획 아키텍처를 제안한다. 품질 정보는 웹서비스의 발견과 최적 실행 계획 도출 과정에서 필요하다. 기존 웹서비스 명세화 문서가 가진 정보에 품질의 속성을 첨부함으로써 사용자들이 보다 양질의 서비스를 선택할 수 있게 된다. 그리고 사용자들이 실질적으로 중요시 하는 품질의 제약들이 반영된 실행 계획을 수립할 수 있다.

1.2 웹서비스 품질 판단 지표에 대한 선정

웹서비스를 구성하는 여러 속성들을 정리해보고 그 중에서 중요하게 고려되어야 하는 비함수적 속성들을 품질 정보 습득 및 판단을 위한 지표로 선정한다. 선정된 품질 판단 지표는 개별 웹서비스를 평가하기 위한 함수 모델에 사용된다.

1.3 웹서비스 실행 계획 수립 복잡도를 줄이기 위한 변형된 휴리스틱 알고리즘의 제안

웹서비스 실행 계획은 여러 웹서비스가 조합되어 있다. 각 개별 웹 서비스에는 적절한 후보 업체의 웹서비스가 필요하다. 가능한 계획의 모든 조합 중에서 최고의 계획을 고르기에는 시스템의 수행 시간에 무리가 있다. 따라서 적절한 휴리스틱 기법을 통해서 초기 실행 계획의 수립과 사용자가 지정한 제약식이 고려된 이웃해의 이동 즉 개선된 계획을 통해서 웹서비스의 실행 가능한 계획을 수립할 수 있다.

2. 관련 연구

2.1 품질 정보를 기반으로 한 웹서비스 컴포지션 연구

웹서비스의 품질 정보 연구로는 대표적으로 웹서비스 컴포지션을 위한 품질 정보 기반의 미들웨어 연구, LSDIS의 시맨틱 웹 프로세스 관련 연구 그리고 WSQoS 프레임웍에 대한 연구가 있다.

품질 정보 기반의 미들웨어 연구에서는 서비스의 비함수적인 속성을 기반으로 한 품질 정보 모델링을 통해 사용자 요구가 반영된 최적의 실행 계획 선택에 초점을 맞춘다(Canfora *et al*., 2005; Liu *et al*., 2004; Zeng *et al*., 2004). 그리고 국소 최적화 기법, 글로벌 계획 기법, 정수 계획법을 통하여 미들웨어의 핵심 부분인 QoS 기반의 컴포지션 알고리즘을 구현하고 있다. 웹서비스의 품질정보 기술을 위해서는 도메인, 서비스 품질 모델, 서비스 클래스로 이루어진 서비스 온톨로지를 이용하고 있다. 제공된 품질 정보는 서비스 중계자를 통해 UDDI 레지스트리 안에 함께 제공되며 서비스 컴포지션 관리자를 통해 실행 플래너(Planner)와 실행 엔진이 제어가 된다.

다음으로 LSDIS의 연구에서는 기존의 워크플로우 관리 시스템을 기반으로 하여 워크 플로우와 웹서비스 프로세스를 이루는 서비스들의 품질 지표를 자동적으로 평가하는 품질 모델을 제안하고 있다(Cardoso, 2002; Cardoso *et al*., 2002). 즉, 워크플로우의 여러 유형에 대한 품질 지표를 계산하기 위해서 SWR(Stochastic Workflow Reduction) 알고리즘을 사용한다. 추가적으로 웹서비스의 개발, 발견, 컴포지션 그리고 실행을 다루는 시멘틱 웹 프로세스와 컴포지션에 대한 연구도 있다(Cardoso and Sheth, 2003; Cardoso and Sheth, 2004). 특히 웹서비스 발견은 매칭함수를 이용하여 구문론적 유사성, 사용상(Operational) 유사성 그리고 시맨

틱적 유사성에 의해서 이루어진다. 그리고 컴포지트 서비스는 시맨틱 정보를 이용한 평가 등급에 따라 구성되고 있다. 컴포지트 서비스를 실행하기 위한 기술로는 BPEL4WS(Business Process Execution Language for Web Services)와 DAML-S (DAML Services)가 소개되고 있다(Russell and Norvig).

끝으로 WSQoS Framework에 대한 연구에서는 품질정보를 가진 웹서비스를 위한 기반 구조를 제안하기 위한 프레임웍을 소개하고 있다(Gram, 2004). 프레임웍 내의 품질 정보 관리를 위해서 품질 정보에 대한 정의, 품질 타입에 따른 품질 정보 구조와 온톨로지가 제안되고 있다. 프레임워크는 궁극적으로 품질 정보 관리자, 품질 정보 제공 중계자, 모니터링 기능을 통해 요구사항에 따른 서비스 탐색 및 실시간 적인 품질정보 매핑 그리고 사용자의 즉각적인 피드백을 목표로 한다.

2.2 웹서비스간 컴포지션 가능성이나 관계를 이용한 컴포지션의 연구

웹서비스의 품질 판단 정보가 아닌 웹서비스간 컴포지션 가능성(Composability)을 이용하거나 웹서비스간 관계를 분석한 컴포지션의 연구와 컴포지션 기술들을 이용한 시맨틱 웹서비스의 필터링과 선택에 대한 연구도 진행되었다.

컴포지션 가능성을 이용한 웹서비스 컴포지션에서는 WSDL (Web Service Description Language)과 DAML+OIL(DARPA Agent Markup Language + Ontology Interface Layer)을 통해 기술된 웹서비스간의 상호작용성과 컴포지션 가능성을 웹서비스간 구문론적 특징과, 시맨틱적 특징을 기준으로 판단하게 된다(Medjahed *et al*., 2003). 이 연구의 핵심은 자동적인 컴포지션이다. 그것을 실행하기 위해서 웹서비스의 명세화, 중계, 선택, 그리고 생성의 4단계를 제안하고 있다. 첫 번째 단계에서는 CSSL(Composite Service Specification Language)이라는 웹서비스 명세화 언어와 UDDI 비즈니스 저장소를 통해서 웹서비스를 기술하는 단계이다. 두 번째 단계는 명세화 된 웹서비스를 이용해 컴포지션 가능성 규칙에 따라 컴포지트 계획들이 생성되는 단계이다. 중계 알고리즘은 컴포지션 가능성 판단 함수, 웹서비스간의 선행관계, 그리고 컴포지션 템플릿을 이용해 실행 가능한 모든 컴포지트 계획의 후보들을 열거한다. 중계를 바탕으로 열거된 계획들은 전문가 그룹이나 시스템의 학습에 의해 서비스의 절차가 저장된 템플릿과의 관계성, 유사성, 적절성 등의 컴포지션 관련 품질 요소들을 통해 평가가 된다. 가장 평가 순위가 높은 계획이 생성단계를 통해 최종적으로 컴포지트 서비스로 생성되게 된다.

다음으로 웹서비스간의 관계를 분석한 컴포지션의 연구에서는 사례 기반 추론 기술을 통한 발견적 웹서비스 컴포지션 모델을 제안하고 있다(Limthanmaphon and Zhang, 2003). 이 연구에서는 웹서비스 간의 관계를 독립적 관계, 필요조건 관계, 병렬적 필요조건 관계, 병렬적 의존 관계, 대체 관계 그리고 중복 관계의 6가지로 분류하여 관계 유형별로 사례를 수집한다. 사례 기반을 통한 컴포지션 모델은 요구 분석자, 외부 공급되는 에이전트(agent), 서비스 컴포져로 이루어져 있다. 요구 분석자는 시스템을 통해 들어오는 사용자의 쿼리를 판단하여 에어전트에게 서비스 내용을 보낸다. 에이전트는 사례 기반 엔진과 UDDI를 통해 서비스를 발견하는 역할을 하며 컴포져는 서비스의 매개변수, 제약조건, 관계를 이용하여 서비스를 컴포지션한다. 요컨대, 이 연구는 앞서 제시한 연구와 더불어 시맨틱적으로 기술된 웹서비스의 세부적인 속성들을 파악하여 자동적으로 하나의 컴포지트 웹서비스를 생성하며, 유연한 컴포지션의 제공이 가능함을 제시하고 있다.

끝으로 시맨틱 웹서비스의 필터링과 선택에 대한 연구에서는 서비스의 필터링과 선택을 통한 컴포지션을 수행하기 위해 중계 알고리즘을 이용하여 목표지향적이고 상호작용적인 방법론을 언급하고 있다(Sirin *et al*., 2004). 서비스 컴포지션을 위한 목표 지향적 접근은 서비스 컴포지션의 각 단계마다 정방향 혹은 역방향으로 추론하여 컴포지션을 진행한다. 또한 사용자의 선호에 맞는 서비스를 걸러내기 위해 입출력 정보, 선행조건, 효과 등의 정보가 사용되고 있다. 이 연구에서는 BPEL4WS을 통한 웹서비스의 흐름 기술, 웹서비스 발견을 위해 DAML-S 중계 시스템과 시맨틱적인 표현력이 높은 OWL-S(Ontology Web Language-Services)를 언급하고 있다. 이런 기술들을 통해 서비스 정보를 저장하고 사용자 요구사항에 서비스의 프로세스를 매칭시키는 추론 엔진과 서비스 컴포지션 워크플로우를 생성하기 위한 컴포져(Composer)로 이루어진 하나의 프로토타입 시스템을 소개하고 있다.

2.3 타부서치(Tabu Search) 기법의 소개

최근 컴퓨터 성능의 발전에도 불구하고, 대형 최적화 문제 중에서 많은 문제들은 NP-hard로서 문제크기가 커짐에 따라 컴퓨터 용량과 계산시간의 한계를 갖게 된다. 본 논문에서 다루고자 하는 문제도 수많은 후보 서비스들 안에서의 실행 가능한 계획의 생성이므로 NP-hard의 범주에 속하며, 사용자의 제약식을 모두 만족해야하므로 제약식 만족 문제(Constraint Satisfaction Problem)의 범주에도 속한다(Nonobe and Ibaraki, 1998; Russell and Norvig).

NP-hard 문제에 대해서 빠른 시간에 근사적인 최적값을 도출하는 여러가지 휴리스틱 기법의 연구가 이루어지고 있다. 여러 기법들 중 다양한 문제에 적용가능하며 해공간의 탐색성능이 우수한 타부서치는 국소 최적해에 빠질 확률이 작고 유전자 알고리즘이나 시뮬레이티드 어닐링(Simulated Annealing)등의 전역적 탐색에 비해 특정 문제에 대한 지식을 활용하기에 유리하다. 일반적으로 타부서치는 해의 변경 속성이 타부 상태가 아닐 경우 해의 이동을 허락하고 선입선출 규칙에 의해서 타부 목록을 관리하며 해를 개선해가는 방법을 취한다. 즉, 타부 목록을 통해 해의 재방문이 방지되고 해의 변경 속성이 보관된다(Kim *et al*,, 1999).

다음으로 제약식 만족 문제의 경우에는 모든 제약식을 만족하는 것이 가장 관건이 된다. 제약식 만족 문제는 일반적으로

제한된 변수의 집합, 변수가 갖는 제한된 도메인, 제약조건의 집합으로 구성된다. 이 세 가지 구성요소를 통해 도메인 안에서 선정된 변수들이 모든 제약 조건을 만족하는지를 판단하게 된다. 제약식 만족 문제에 휴리스틱 기법을 응용하기 위해서 벌금 함수나 휴리스틱 각 요소의 유연한 설계 기법 등이 사용된다.

3. 웹서비스 품질 정보를 기반으로 한 웹서비스 실행 계획 아키텍처

3.1 웹서비스 실행 계획 아키텍처

기존의 웹서비스 실행 아키텍처는 UDDI 서버, 웹서비스 제공자와 사용자(요청자) 그리고 웹서비스 개발자로 구성된다. 웹서비스 요청자는 서비스의 요청을 위해 서비스를 중개하는 UDDI 서버를 통해 서비스를 검색한다. 요청된 서비스는 웹서비스 개발자와 계약 관계에 있는 웹서비스 제공자를 통해 출판되어 웹서비스 사용자에게 제공되고, 제공자와 사용자 간의 계약이 이루어진다. 이러한 기존 모델에 웹서비스의 발견과 요청 그리고 실행 계획의 도출에 있어서 품질 정보가 고려된 새로운 웹서비스 실행 계획 시스템의 아키텍처를 제안한다.

제안된 아키텍처는 동적인 웹서비스 환경에서 제안된 품질 레지스트리 모델과(Liu *et al*., 2004), 웹서비스의 실행 계획 도출을 위한 웹 프로세스 라이프 사이클을 고려한 시맨틱 웹 프로세스(Cardoso *et al*., 2002)에 기반하여 웹서비스 품질 정보가 반영된 실행 계획을 도출한다. 그리고 웹서비스의 명세화, 발견, 최적 계획도출 및 웹서비스의 실행의 전반적인 과정을 다루게 된다(Rao and Su, 2004).

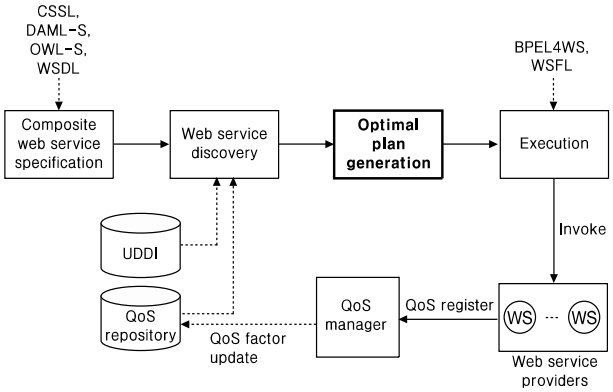


Figure 1. 컴포지트 웹서비스 실행 아키텍처.

아키텍처<Figure 1>는 크게 컴포지트 웹서비스 명세화 모듈(Composite web service specification), 웹서비스 발견 모듈(Web service discovery), 최적 웹서비스 실행 계획 도출 모듈(Optimal plan generation), 실행 모듈(Execution)의 핵심모듈들과 관리모듈인 QoS 관리자 모듈(QoS manager)로 구성된다.

명세화 모듈은 웹서비스 사용자와 직접 연결되는 부분이며 사용자의 서비스 요청에 적합한 컴포지트 웹서비스를 명세화해준다. 웹서비스의 명세화는 웹서비스를 논리적으로 기술하기 위해 사용된다. 그리고 사용자의 요청과 관련된 여러 도메인별 컴포지트 서비스는 실행계획 선택 모듈<그림2>을 통해 등록, 관리, 검색된다. 웹서비스를 명세화하기 위해서는 CSSL, DAML-S, OWL-S 등의 기술들이 이용된다. CSSL은 웹서비스 정의 언어인 WSDL와 형태가 비슷하며 웹서비스의 시맨틱적 특성을 표현할 수 있으며 컴포지트 서비스의 오퍼레이션들 간의 흐름을 명세화하는 장점이 있다(Medjahed *et al*., 2003). DAML-S는 시맨틱 웹서비스 구현을 위한 핵심적인 컴포넌트로 DAML+OIL 기반의 서비스 기술을 위한 온톨로지 언어이며 현재 OWL-S로 발전되어 제공되고 있다. OWL-S는 크게 3부분으로 구성된다. 첫째 서비스 프로파일(Service profile)은 웹서비스의 구조, 수행 기능, 특징, 서비스 실행의 인풋, 아웃풋을 정의한다. 둘째 프로세스 모델(Process model)은 서비스의 구체적인 프로세스 동작을 기술하기 위해 프로세스 흐름, 컴포지션 계층, 각 프로세스의 IOPE(Input, Output, Precondition, Effect)를 정의한다. 끝으로 서비스 그라운딩(Service grounding)은 웹서비스가 실제로 수행하기 위해 필요한 모든 정보를 기술하고 실질적인 동작을 처리한다(Martin *et al*., 2004).

다음으로 발견 모듈은 컴포지트 웹서비스의 개별 웹서비스 발견을 담당한다. 각 개별 웹서비스에 관련된 웹서비스 후보군을 발견하기 위해서 명세화 되어 기술된 웹서비스를 바탕으로 UDDI와 QoS 레파지토리에서 해당 후보 웹서비스의 정보를 추출한다<Figure 2>. QoS 레파지토리에 대해서는 먼저 SLA(Service Level Agreement)를 언급할 수 있는데, SLA는 사용자와 서비스 사이의 서비스 수준에 대한 협약을 의미한다(Zeng *et al*., 2004; Rao and Su, 2004). 즉, 서비스 제공자가 서비스의 사용자에게 제공해야 하는 서비스 특성 및 그와 관련된 성능 측정 기준을 명시한 문서를 말한다. 따라서 서비스의 신뢰성과 지속 발전 가능성을 고려할 때, 웹서비스 제공자는 SLA에 근거하여 품질에 관련한 정보, 서비스 클래스의 종류, 서비스의 이용 가능성, 고장 및 변경에 대한 배상 관련 정보, 그리고 서비스 성능 측정 방법 등의 정보들을 명시해 놓아야 한다. 명시된 QoS 정보는 QoS 관리자에 의해서 QoS 레파지토리에 등록되며 지속적인 업데이트가 이루어진다.

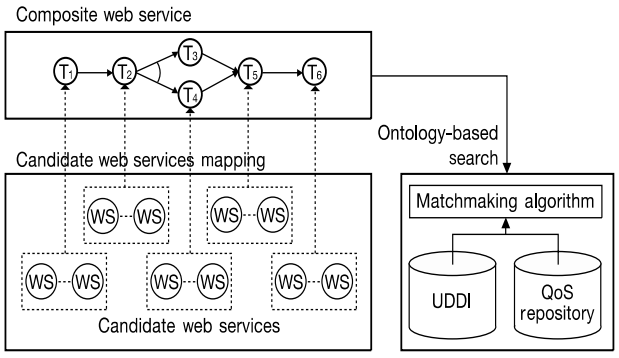


Figure 2. 후보 웹 서비스들의 발견.

후보 웹서비스의 발견이 이루어지면 도출 모듈에 의해서 사용자의 요구 사항과 제약 조건에 적합한 개별 웹서비스들로 컴포지트 웹서비스가 구성된다. 본 논문에서는 컴포지트 웹서비스가 실질적으로 완성되는 최적 웹서비스 실행 계획 도출 모듈에 대해서 집중적으로 다루고 있다. 그리고 계획 도출을 위해서 QoS 정보 기반의 제약식 만족 문제 컨셉과 몇몇 휴리스틱 기법을 이용하여 실행 가능한 계획을 찾기 위한 알고리즘을 제안하고자 한다.

계획 실행 모듈은 도출된 계획을 바탕으로 각 해당 웹서비스 제공자에게 서비스 제공을 요청한다. 서비스의 실행에 있어서는 서비스들의 기능들을 한 시퀀스로 엮어 자동적으로 실행 계획을 생성하는 일이 중요하다. 그것과 관련해 WSFL(Web Service Flow Language), BPEL4WS 등의 기술들이 이용되고 있다(Cardoso, 2002;Medjahed *et al*., 2003). WSFL은 컴포넌트 서비스들의 실행 시퀀스를 나타내는 플로우 모델(Flow model)과 컴포넌트 서비스들 간 상호작용의 기술과 각 서비스 오퍼레이션을 연결시켜주는 글로벌 모델(Global model)로 이루어져 있다. BPEL4WS는 실행 가능한 비즈니스 프로세스들과 프로세스의 규약들을 명세화하며 프로세스들의 시퀀스에 대해서 직접 실행이 가능하도록 각 프로세스를 요청하는 역할을 한다.

3.2 웹서비스의 품질 판단 지표

웹서비스의 품질을 판단할 수 있는 지표를 선정하기 위해서는 웹서비스의 함수적인 속성들 보다 비함수적인 속성들이 더 중요시 고려된다(O'Sullivan *et al*., 2002). 따라서 여러 고려사항들이 있겠지만 그중에서 서비스의 접근과 이용가능성 그리고 성능과 신뢰성 판단 등이 고려되어야한다.

품질이 고려된 웹서비스의 계획 수립을 위해 다음 6가지의 품질 판단 지표를 이용하고자 한다. 각 지표들은 웹 서비스들의 컴포지션을 수행하기 위해 가장 중점적으로 고려되어야 할 지표들을 위주로 선정되었다. 평가점수와 이용 빈도 지표는 초기 계획을 수립하는데 사용되며 나머지 지표들은 사용자가 지정한 제약조건과 직접적인 연관성이 있다.

- **이용가능도**(av: Availability) : 이용가능도는 웹서비스가 즉각적으로 사용할 준비가 되어있는지를 나타내는 상태를 의미한다. 즉 서비스가 이용 가능한가를 나타내며, Time-To-Repair (TTR)와 밀접한 관련성이 있다. 어떤 웹 프로세스의 후보 서비스의 이용가능도는 av = Uptime/(Uptime+Downtime)로 나타낸다.
- **소요 비용**(c: Execution cost) : 웹서비스를 사용하는데 소요되는 비용을 의미한다. 서비스 제공자의 정책 변경이나 서비스 개선 및 기타사항 등에 의해서 변경될 수 있다.
- **소요 시간**(t: Execution time) : 웹서비스를 실행하는데 소요되는 평균 시간을 의미한다. 각 유저들에게 있어서 웹서비스를 이용하는 환경은 동일하지 않다. 따라서 실행 시간은 사용자가 서비스를 요청했을 때 결과가 도출되는 시간을 기준으로 측정되어야 한다.
- **이용빈도**(f: Frequency) : 웹서비스의 이용 빈도를 의미한다. 여러 웹서비스가 컴포지션을 이루어 사용이 되든, 개별적으로 사용이 되든 이용 빈도수가 높다는 것은 사용자의 신뢰가 있음을 나타낸다.
- **평가점수**(rep: Reputation) : 실제 웹서비스를 이용한 사용자들에 의해서 평가가 되는 웹서비스의 평판에 대한 지표를 의미한다. 사용자 평가 점수 지표는 rep = (전체 평가점수 / 참여 인원)으로 나타낼 수 있으며 평가 범위는 [0, 10]으로 가정하였다.
- **실행성공률**(suc: Successful execution rate) : 사용자가 웹서비스 실행을 요청했을 때 실제적으로 웹서비스를 수행했는지를 나타내는 성공률 지표를 의미한다. 실행 성공률의 초기값은 1이며 전체 사용빈도에서 성공적으로 서비스가 수행한 정도로 나타낸다.

3.3 웹서비스의 평가 함수 모델

웹서비스 평가 함수는 웹서비스를 수행하는 여러 제공자들 중에서 가장 우수한 웹서비스 후보를 선정하기 위해서 필요하다. 앞에서 정의한 몇몇 품질 판단 지표와 기존 연구(Zeng *et al*., 2004)에서 제안된 국소 최적화 방법을 기반으로 웹서비스의 평가 함수를 다음과 같이 제안해 본다.

각각의 품질 판단 지표들이 서로 스케일이 다르며 실행 비용과 실행 시간 지표의 경우에는 값이 작을수록 좋고 이용가능도와 실행 성공률 지표의 경우에는 값이 클수록 좋다. 따라서 첫 번째 단계로 각 지표들의 스케일을 동일하게 하기 위한 표준 정규화의 작업을 수행해야 한다. 그리고 사용자가 지정한 각각의 품질 판단 지표에 대한 가중치를 이용하여 최종적인 서비스의 평가 점수를 측정한다.

표준 정규화를 통한 지표의 스케일링(Scaling)

한 개별 웹서비스를 수행하기 위한 웹서비스 후보가 m개 있다고 가정하자. 각 후보 서비스들은 앞서 정의한 각 품질 지표 정보를 가지고 있다. i번째 개별 웹서비스를 수행하기 위한 임의의 후보 서비스 s_{ij}는 소요비용 및 시간, 실행성공률, 이용가능도의 품질 정보에 대해서 $s_{ij} = \{c_{ij}, t_{ij}, suc_{ij}, av_{ij}\}$의 형태를 띤다고 가정할 때, 표준 정규화를 이용하여 $s_{ij}' = \{q_1, q_2, q_3, q_4\}$와 같이 변형한다.

$$q_1 = -\frac{c_{ij} - \text{소요 비용의 평균 값}}{\text{소요 비용의 표준편차}} \quad (1)$$

$$q_2 = -\frac{t_{ij} - \text{소요 시간의 평균값}}{\text{소요 시간의 표준편차}} \quad (2)$$

$$q_3 = \frac{suc_{ij} - \text{실행성공률의 평균}}{\text{실행성공률의 표준편차}} \quad (3)$$

$$q_4 = \frac{av_{ij} - \text{이용가능도의 평균}}{\text{이용 가능도의 표준편차}} \quad (4)$$

가중치 합산을 통한 웹서비스의 평가 점수 측정

$$score(s_{ij}) = w_1q_1 + w_2q_2 + w_3q_3 + w_4q_4 \quad (5)$$

w_k는 사용자가 지정한 가중치이고 $\sum_{k=1}^{4} w_k = 1$이 되며 점수가 높을수록 우수한 웹서비스 후보를 의미한다. 즉 가장 점수가 높은 후보 서비스는 사용자의 요구사항을 가장 잘 만족하는 서비스를 의미한다.

3.4 웹서비스의 실행 계획

웹서비스의 실행 계획은 선택된 후보 웹서비스를 통한 컴포지트 웹서비스의 완성을 의미한다. 웹서비스 실행 계획 완성을 위해 웹서비스 사용자는 컴포지트 서비스의 도메인을 선택해야 한다.

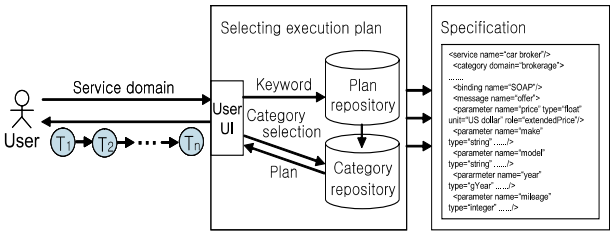


Figure 3. 웹서비스 실행계획 선택 모듈.

사용자는 <Figure 3>에서처럼 키워드 기반의 검색이나, 카테고리 안에서 선택 방법을 통해 원하는 웹서비스 실행계획을 선택하게 된다. 카테고리 저장소에는 서비스의 도메인별 분류 체계가 저장되어 있다. 웹서비스 계획 레파지토리(Plan repository)에는 각 도메인별 서비스간의 의미관계와, 도메인 간의 유사성 등에 대한 정보들이 저장된다. 그리고 그 정보에 따라 카테고리 저장소와 연계하여 해당 컴포지트 서비스를 찾아내게 된다. 실행계획이 선택되면 4장에서 제안될 알고리즘을 통해 각 후보 서비스들이 선택된다.

4. 휴리스틱 기반의 웹서비스 컴포지션 실행 계획 수립 모델

이제 실행 계획 모듈에서 선정된 웹서비스 실행 계획을 바탕으로 각 서비스를 계획에 적합하게 선택하는 과제가 남았다. 우수한 각각의 개별 웹서비스의 선택만으로는 사용자가 정해놓은 모든 제약식을 제대로 고려할 수 없다. 반면 가능한 모든 웹서비스 조합을 전부 고려하는 것은 수행 시간이 너무 많이 소요된다. 따라서 본 논문에서는 제약식을 만족하는 각 웹서비스들의 선택의 시간을 단축하기 위해서 휴리스틱 기법들을 이용한 웹서비스 컴포지트 모델을 제안한다.

먼저, 앞서 2.3에서 제시한 타부서치와 시뮬레이티드 어닐링 기법을 혼합하였다. 타부서치의 타부 리스트는 제약식 별로 각각 운영되며 각각의 리스트들 간 관계가 유연하도록 설계하였다. 타부상태를 나타내는 타부 터너(Tabu Tenure) 역시 제약식의 만족 정도에 따라서 유동적으로 변화되도록 설계 하였다. 끝으로 새롭게 얻어진 실행 계획이 전체 제약식 만족도를 높이고 실행 계획을 변경 시 발생 가능한 지역 최적화를 방지하기 위해 해의 이동을 확률적으로 수락하는 시뮬레이티드 어닐링 기법을 이용하였다.

4.1 기호 정리

웹서비스 컴포지션을 위한 실행 계획 수립 알고리즘에 사용되는 기호들을 정리하면 다음과 같다.

T_i: 컴포지션을 구성하는 i번째 컴포넌트 서비스
s_{ij}: i번째 컴포넌트 서비스를 수행하기 위한 j번째 후보 서비스
c_{ij}: s_{ij}의 소요 비용(execution cost)
t_{ij}: s_{ij}의 소요 시간(execution time)
av_{ij}: s_{ij}의 이용가능도(availability)
suc_{ij}: s_{ij}의 실행성공률(successful execution rate)
rep_{ij}: s_{ij}의 평가점수(reputation)
f_{ij}: s_{ij}의 이용빈도(frequency)
x_{ij}: s_{ij}의 결정변수(decision variable)
w_k: 사용자가 정한 k번째 제약조건의 가중치(weight)
C_k: k번째 제약조건의 제약값

4.2 수리적 모형

수리적 모형을 설계하기 위해서 각 개별 서비스에 대해서 m개씩 후보 웹서비스가 존재한다고 가정했다. 즉 $s_{i1} \sim s_{im}$는 T_i를 수행하기 위해서 제공되는 후보 서비스들을 나타낸다. 최종적으로 품질 판단 지표들을 이용하여 결정변수인 x_{ij}를 결정해야 한다. x_{ij}는 s_{ij}가 선택되었을 경우는 1값을 갖고 그렇지 않으면 0값을 갖는다.

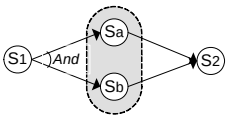
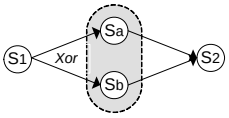
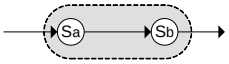
결론적으로 수리적 모형 설계의 궁극적인 목표는 모든 제약식을 어기지 않는 실행 가능한 계획을 구성하는 가장 적절한 각각의 개별 웹서비스를 찾는 것이다. 또한 발견될 수 있는 실행 가능한 계획 집합의 영역에 신속하게 도착해야 한다.

(1) 제약조건

제약조건은 웹 서비스의 할당에 대한 제약조건이 있으며 사용자가 지정하는 실행계획에 대한 실행 비용, 실행 시간, 실행 성공률 그리고 이용가능도에 대한 제약조건이 있다. 제약조건을 나타내기 위해서는 실행계획을 구성하는 웹 서비스들의 제약조건 관련 품질 지표들을 통합해야 한다. 품질 지표를 통합하는 방법은 다음과 같이 나타낼 수 있다.

• **웹 서비스 품질 지표의 통합** : 실행계획에서 나타날 수 있는 순차적인 패턴(Serial pattern), AND 병렬 패턴(AND-parallel pattern) 그리고 XOR 병렬 패턴(XOR-parallel pattern)에 대해서는 다음과 같은 지표 통합을 진행한다. XOR-parallel의 경우에는 양쪽 서비스 중에서 임의의 한 서비스가 수행되기 때문에 양쪽 모든 후보 웹 서비스를 후보로 포함한다고 할 수 있다.

Table1. 품질지표의 통합

	AND-parallel	
소요비용	$c=c_a+c_b$	
소요시간	$t=t_a+t_b$	
실행성공률	$suc=suc_a\times suc_b$	
이용가능도	$av=av_a\times av_b$	
	XOR-parallel	
소요비용	$\{s\}=\{s_a\}\cup\{s_b\}$	
소요시간		
실행성공률		
이용가능도		
	Serial	
소요비용	$c=c_a+c_b$	
소요시간	$t=t_a+t_b$	
실행성공률	$suc=suc_a\times suc_b$	
이용가능도	$av=av_a\times av_b$	

다음으로 k번째 제약 조건의 제약값을 C_k라 할 때 제약 조건은 다음과 같이 나타내어진다.

• **웹서비스 할당 제약 조건** : 각각의 개별 서비스에 는 1개의 후보 웹서비스만이 선택되어야 한다.

$$\sum_{j=1}^{m} x_{ij}=1,\ \forall i\in \text{실행계획내의 모든 } T_i \qquad (6)$$

• **실행 비용, 실행 시간 제약 조건** : 할당 된 웹서비스들의 실행 비용(7)과 실행 시간(8)은 각 웹서비스들의 해당 지표를 다 더한 것과 같으며 사용자가 지정해 놓은 제약값을 만족해야 한다. 각각의 제약 조건은 다음과 같이 나타낼 수 있으며 $P_k(x_{ij})$라는 함수로 간단히 표현하기로 한다.

$$\sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}x_{ij} \le C_1 \Leftrightarrow P_1(x_{ij}) \le C_1 \qquad (7)$$

$$\sum_{i=1}^{n}\sum_{j=1}^{m} t_{ij}x_{ij} \le C_2 \Leftrightarrow P_2(x_{ij}) \le C_2 \qquad (8)$$

• **실행 성공률, 이용가능도 제약 조건** : 할당 된 웹서비스들의 실행 성공률(9), 이용가능도(10)는 각 웹서비스들의 해당 지표의 곱으로 나타낼 수 있으며 사용자가 지정해 놓은 제약값을 만족해야 한다. 제약 조건은 다음과 같이 나타낼 수 있으며 마찬가지로 $P_k(x_{ij})$라는 함수로 간단히 표현하기로 한다.

$$\prod_{i=1}^{n}\left(\sum_{j=1}^{m} suc_{ij}x_{ij}\right) \ge C_3 \Leftrightarrow P_3(x_{ij}) \ge C_3 \qquad (9)$$

$$\prod_{i=1}^{n}\left(\sum_{j=1}^{m} av_{ij}x_{ij}\right) \ge C_4 \Leftrightarrow P_4(x_{ij}) \ge C_4 \qquad (10)$$

(2) 제약조건을 고려한 목적 함수

모든 제약식을 만족하는 실질적으로 실행 가능한 계획을 찾는 것이 목적이므로 목적 함수를 설정하는 데 있어서 앞서 언급된 제약 조건이 모두 고려된 목적 함수를 제안한다. 제안한 목적 함수는 다음과 같다.

$$\min\left[\sum_{k=1}^{2} w_k \cdot \max\left(\frac{P_k(x_{ij})-C_k}{C_k},0\right) + \sum_{k=3}^{4} w_k \cdot \max\left(\frac{C_k-P_k(x_{ij})}{C_k},0\right)\right] \qquad (11)$$

w_k는 사용자가 지정한 가중치이며 $\sum_{k=1}^{4} w_k = 1$이 된다.

실행 비용과 실행 시간 제약 조건은 제약값 이하가 되어야 하므로 제약값보다 클 경우, 목적 함수는 양의 값을 갖는다. 실행 성공률과 이용가능도 제약 조건은 제약값 이상이 되어야 하므로 제약값보다 작을 경우, 목적 함수는 양의 값을 갖는다. 제약 조건을 어기는 정도가 클수록 목적 함수의 값은 커진다. 목적 함수 값이 0이 되면 모든 제약식을 만족하게 되고 찾고자 하는 실행 계획이 도출된다.

(3) 실행 계획의 도출을 위한 휴리스틱 알고리즘

실행 계획을 도출하기 위해 임의의 초기 실행 계획을 선정하고 휴리스틱 기법을 이용해서 실행 계획을 구성하는 개별 웹서비스들을 변형시켜가면서 제약식을 만족하는 실행 계획을 찾는 방법을 제안한다.

초기 실행 계획의 선정

초기 실행 계획을 잘 선정할수록 최종 실행 계획에 도달하는데 유리하다. 보다 실질적인 초기 실행 계획 선정을 위해서 품질 판단 지표 중에서 이용 빈도수와 사용자에 의한 평가를 이용하였다. 즉 사용자의 이용과 평가가 높은 웹서비스들을 위주로 초기 실행 계획을 선정하였다. 만약 품질 판단 지표 값이 겹치거나 0인 경우는 랜덤하게 선택하였다. 선정된 실행 계획은 실행 계획 리스트(PL)에 저장한다.

이웃 실행 계획의 생성

선정된 실행 계획이 제약 조건을 어길 경우, 즉 목적함수 값이 0이 아닐 경우 실행 계획의 수정을 위해서 이웃 실행 계획을

생성해준다. 먼저 제약값을 어긴 제약 조건을 체크한다. 실행 계획의 수정은 제약조건에 대한 가중치가 목적 함수 값에 상대적으로 많은 영향을 준다는 가정 하에 w_k가 높은 순으로 수정을 해준다. 단, 여러 개의 제약 조건을 고려해야 되기 때문에 하나의 제약 조건을 만족시키기 위해 다른 제약식이 간과 되서는 안 된다. 따라서 실행 계획내의 웹서비스를 교체할 경우 웹서비스 평가 함수 모델과 제약식 만족문제의 컨셉을 이용하여 전체 제약 조건을 고려해준다.

이 과정을 하나의 예제 케이스를 통해 설명하겠다.

예제 사례

실행 비용(w_1)과 실행 시간(w_2) 관련 제약조건을 어겼을 경우 (단 $w_1 > w_2$)

- 단계 1 : 실행 계획($T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow ... \rightarrow T_n$)에 대해서 실행 비용이 높은 순서로 개별 서비스들을 정렬한다.
- 단계 2 : 정렬 결과가 $T_3 > T_5 > T_7 > T_4$...와 같다고 할 때 가장 비용이 높은 4개의 서비스 $\{T_3, T_5, T_7, T_4\}$를 임시 리스트인 타부 리스트1(TL1)에 집어넣는다.
- 단계 3 : 이 중에서 가장 개선이 필요한 2개의 개별 서비스인 T_3 , T_5는 다음 조건을 만족할 경우 웹서비스 후보를 웹서비스 평가 함수 값이 가장 좋은 후보($s_{i,new}$)로 교체한다.

  T_3: $c_{3,\,old} > c_{3,\,new}$이면 $s_{3,\,old}$를 $s_{3,\,new}$로 교체
  T_5: $c_{5,\,old} > c_{5,\,new}$이면 $s_{5,\,old}$를 $s_{5,\,new}$로 교체

  교체된 후보 서비스는 타부 터너 동안 타부 상태, 즉 변경이 요청되어도 값을 변화시키지 않는 상태가 유지된다.
- 단계 4 : 다음으로 실행 계획의 개별 서비스를 실행 시간이 가장 높은 순으로 정렬한다.
- 단계 5 : 정렬 결과가 $T_{10} > T_2 > T_4 > T_8 > T_9$...와 같을 때 가장 실행 시간이 큰 4개의 개별 서비스를 임시 리스트인 타부 리스트2(TL2)에 넣는다. 개별 서비스가 다른 타부 리스트에 겹치더라도 제외하고 않고 리스트에 집어넣는다. 즉 타부 리스트 2는 $\{T_{10}, T_2, T_4, T_8\}$의 개별 서비스 항목을 가지게 된다.
- 단계 6 : 이 중 가장 개선이 필요한 2개의 개별 서비스, T_{10}, T_2에 해당되는 웹서비스 후보를 웹서비스 평가 함수 값이 가장 좋은 후보로 교체하며 타부 터너 동안 타부 상태로 유지한다. 만약 가장 개선이 필요한 개별 서비스가 이미 타부 상태라면 4개의 개별 서비스 중에서 다음 순위의 개별 서비스를 교체하게 되고, 4개의 개별 서비스 중 타부 상태가 아닌 개별 서비스가 없다면 교체를 제외하고 다음 단계로 넘어간다.

새롭게 선정된 실행 계획은 실행 계획을 기억해 놓는 리스트인 실행 계획 리스트에 저장하고 목적함수의 개선 정도를 살핀다. 만약 목적 함수 값이 0일 경우에는 실행 가능한 계획이 도출된 것이므로 종료시킨다. 목적 함수 값이 줄어들었을 경우에는 새로운 실행 계획, 즉 후보 웹서비스의 교체를 받아들이고 그렇지 않은 경우에는 확률적으로 받아들인다. 즉, 새로운 실행 계획을 받아들이는 문제를 처리하기 위해서 시뮬레이티드 어닐링의 기법을 이용한다. 기존 실행 계획의 목적 함수 값을 $f_{old}(x_{ij})$, 새롭게 얻어진 실행 계획의 목적 함수 값을 $f_{new}(x_{ij})$라 할 때, (12)를 만족할 경우에만 실행 계획의 이동을 수락한다.

$$\exp[\{f_{old}(x_{ij}) - f_{new}(x_{ij})\} \times iter.] > random(0,1) \quad (12)$$

식에 반복 횟수(Iteration)를 첨부함으로 반복 횟수의 증가에 따라 안 좋은 실행 계획으로의 이동 확률을 점차 줄일 수 있다. 이 과정까지 끝나면 반복 횟수를 증가시키고 만약 지정해 놓은 최대 반복 횟수까지 도달한 경우에는 알고리즘을 강제로 종료시킨다.

타부 터너 선정과 실행 계획의 국지 순환 방지

타부 리스트에 속한 개별 서비스를 타부 상태로 유지하고 있는 기간인 타부 터너 역시 유동적으로 변화시킬 수 있도록 설계한다(Nonobe and Ibaraki, 1998). 즉 목적 함수의 개선률이 좋으면 타부 터너 값을 증가시켜 교체된 후보를 장기간 유지시키고, 개선률이 좋지 않으면 반대로 값을 줄여주었다. 본 논문에서는 타부 터너의 최대값을 5, 최소값을 1로 제한하고 변화 폭은 1씩 증가하거나 감소하는 방법을 사용하였다. 그리고 특정 실행 계획들이 지속적으로 국소 순환하여 실행 가능한 계획을 찾지 못하는 경우를 방지하기 위해 개별 서비스를 교체하는 과정에 있어서 랜덤 요소를 추가시켜 알고리즘을 진행하였다. 실제로 랜덤 요소를 추가할 경우 더 좋은 결과가 나타났다.

품질 판단 지표의 업데이트

최종적인 해가 도출 되었을 경우에는 이용가능도, 실행 성공률, 이용 빈도수, 사용자에 의한 평가값 등의 품질 판단 지표 항목들을 업데이트 해주어야 한다. 업데이트 된 지표 정보는 QoS 관리자에 의해서 관리되며 QoS 레파지토리에 저장해 놓는다.

4.3 동적인 웹서비스 환경을 고려한 알고리즘의 진행

웹의 동적인 환경은 웹서비스 제공자의 서비스 정보변경, 웹 기반 시설들의 오작동 등으로 인해 원활한 웹서비스 공급이 어렵게 되는 우연적인 상황을 의미한다(Zeng *et al*., 2004). 발생된 예외적인 문제를 처리하기 위해서 이미 실행된 부분을 제약 조건에 첨가하여 처리하는 방법과 예외 이벤트 처리기 중심의 대처 방법을 고려하였다. 하지만 신속한 예외처리를 위해서는 앞서 제시한 실행 계획 리스트를 이용하거나 예외사항에 대한 부분적인 탐색을 통한 더 근본적인 대안이 제시되어야 한다.

실행 계획 변경을 위한 첫 번째 방안은 사용자의 제약 조건 지정에 변화를 주는 것이다. 예컨대, 웹서비스 이용 가격이 90

만원(C_k)을 넘지 않아야 하지만 부득이 할 경우 100만원(C_k')까지는 감수할 수 있다는 식의 제약 조건을 설정한다. 하지만 C_k'가 너무 느슨해서는 안 된다. 예외 사항이 발생 시에는 실행 계획 리스트를 탐색하면서 다른 웹서비스 후보를 즉각적으로 채택할 수 있게 한다. 실행 계획 리스트는 저장 공간의 한계를 고려한다면 초기 실행 계획부터 새롭게 생성된 실행 계획 중 가장 우수한 몇 개의 실행 계획이 저장되어있다. 만약 실행 계획 리스트 안에서 적당한 웹서비스 후보가 없을 경우에는 개별 웹서비스 평가 함수를 이용하여 후보를 선택하고 제약조건 만족 여부를 살펴본다. 먼저 기존의 제약조건을 이용해보고 만족하지 않을 경우 새로운 제약조건을 이용해서 만족 여부를 살핀다. 만약 새로운 제약조건을 이용할 경우 다른 웹서비스 후보($s_{i,new}$)가 발견되면 다음 식을 통해서 후보 웹서비스를 수용 여부를 결정한다.

$$\text{소요 비용}: C_1' - \{P_1(x_{ij}) - c_{i,\,old}\} \geq c_{i,\,new} \quad (13)$$

$$\text{소요 시간}: C_2' - \{P_2(x_{ij}) - t_{i,\,old}\} \geq t_{i,\,new} \quad (14)$$

$$\text{실행 성공률}: \frac{C_3' \bullet suc_{i,old}}{P_3(x_{ij})} \leq suc_{i,\,new} \quad (15)$$

$$\text{이용가능도}: \frac{C_4' \bullet av_{i,old}}{P_4(x_{ij})} \leq av_{i,\,new} \quad (16)$$

위에서 제시한 4가지 조건을 모두 만족할 경우 웹서비스 교체를 받아들인다. 예외가 아닌 개별 서비스들은 이미 계약된 상태이므로 실제적으로는 예외로 판별된 개별 웹서비스만 교체하는 방법을 취해야 한다.

두 번째 방안은 계약 상태를 고려하지 않는 방법이다. 즉 예외가 발생한 시점에서부터 끝 시점까지의 웹서비스에 대해서 알고리즘을 이용하여 새로운 실행 계획을 수립하는 방법이다. 사용자가 지정해 놓은 제약조건은 예외 전까지의 서비스를 수행한 만큼 변화가 있으며 변화된 제약조건이 새로운 제약조건으로 알고리즘을 진행시킨다. 만약 n개의 개별 서비스 중 T_7에서 예외가 발생했을 경우 변경된 제약조건은 다음과 같다.

$$\text{소요 비용}: C_1' = C_1 - \sum_{i=1}^{6}\sum_{j=1}^{m} c_{ij}x_{ij} \quad (17)$$

$$\text{소요 시간}: C_2' = C_2 - \sum_{i=1}^{6}\sum_{j=1}^{m} t_{ij}x_{ij} \quad (18)$$

$$\text{실행 성공률}: C_3' = C_3 / \prod_{i=1}^{6}(\sum_{j=1}^{m} suc_{ij}x_{ij}) \quad (19)$$

$$\text{이용가능도}: C_4' = C_4 / \prod_{i=1}^{6}(\sum_{j=1}^{m} av_{ij}x_{ij}) \quad (20)$$

5. 실 험

제안된 알고리즘의 성능을 판단하기 위해 비교 실험을 시행하였다. 알고리즘은 C/C++ 프로그래밍을 통해 구현하고 AMD 1.8GHz, RAM 512Mb, Windows XP 환경에서 실험을 실시했다. 개별 후보 웹서비스들의 품질 정보는 서로 이율배반의 관계에 있는 지표들의 특성과 실생활에서 제공되는 서비스의 유형별 특성을 고려하여 생성하였다. 총 수행 시간을 기준으로 알고리즘의 성능을 판단하였다.

5.1 후보 서비스 교체를 통한 실행 가능한 계획의 탐색

제안한 알고리즘을 통한 실행계획의 탐색 과정은 다음과 같다. 예를 들어 총소요시간이 1500 이하, 총 소요비용이 280 이하, 이용가능도와 실행 성공률은 모두 0.25 이상을 만족해야하는 실행 가능한 계획을 탐색할 경우 <table 2>처럼 계획이 탐색된다.

Table 2. 실행 가능한 계획의 탐색 결과

반복 횟수	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	목적 함수값	수락 여부
1	1	5	1	2	7	5	10	1	3	3	0.372	O
2	1	5	1	2	7	5	10	1	3	3	0.372	O
3	1	5	4	2	7	6	4	5	3	3	0.215	O
4	1	4	4	2	7	6	4	5	3	3	0.234	O
5	1	4	6	2	7	6	6	5	3	3	0.197	O
6	7	2	6	2	7	10	6	7	3	3	0.104	O
7	7	2	6	2	7	10	10	7	3	4	0.049	O
8	7	2	6	4	7	10	1	7	3	4	0.325	X
9	8	2	6	2	7	7	4	5	3	7	0.015	O
10	8	2	6	2	7	7	4	5	3	7	0.015	O
11	8	2	6	2	7	10	4	5	3	7	0.016	O
12	**7**	**2**	**6**	**2**	**7**	**10**	**4**	**5**	**3**	**7**	**0.0**	완료

5.2 컴포지션내 개별 서비스 수의 증가와 후보 서비스 수의 증가에 따른 수행 시간 비교 실험

제안한 알고리즘의 성능 측정을 위해 기존 연구를 고려해 비교실험을 실시했다. 기존 연구로는 개별 서비스마다 가장 우수한 후보를 선택하는 국소 최적화 기법, 여러 실행 계획들 중에서 가장 우수한 실행 계획을 선택하는 글로벌 계획 방법 그리고 최적 계획을 찾기 위한 정수 계획법 등이 있다.

비교 실험을 위해, 초기해 선정 방법과 진행 알고리즘의 중요성을 고려하여 몇 개의 실험으로 나누고, 개별 서비스의 수와 후보 서비스의 수를 증가시키면서 실험을 진행했다. 초기해를 선정하는 데 있어서는 4.2.3에서 제안한 방법과 기존 연구의 국소 최적화 기법을 사용하였으며, 알고리즘의 진행에 있어서는 본

논문의 변형된 휴리스틱을 이용한 방법과 모든 개별 서비스를 랜덤 매핑 하는 방법 그리고 기존 연구의 정수계획법을 사용하였다.

Table 3. 실험의 구성

	초기 계획 선정	진행 알고리즘
실험1	이용 빈도, 평가점수 지표 이용	휴리스틱 알고리즘
실험2	Local optimization	휴리스틱 알고리즘
실험3	랜덤	랜덤
실험4	정수계획법	Branch & bound

초기해 선정 방법에 따라서 약간 다른 실행 가능 계획이 도출되었지만 한정된 제약조건으로 실행 가능한 계획을 찾는데 있어서 논문에서 제안한 알고리즘은 우수한 결과를 보여주었다. 물론 정수계획법은 안정적으로 실행 가능 계획을 찾아주지만 다른 실험들에 비해 너무 많은 수행시간이 걸렸다. 따라서 계산시간 단축을 위해서 가지치기(Branch & bound) 기법을 이용해 제약조건을 하한치로 지정하여 실행 가능한 계획을 탐색했다.

실험 결과에서 보는 것과 같이 제안한 알고리즘을 이용하여 진행할 경우에는 지정된 범위 내에서만 실행 계획을 탐색하기 때문에 단시간에 안정적으로 실행 가능한 계획을 탐색하는 것을 보여주었다. 하지만 모든 개별 서비스를 랜덤하게 매핑 하는 경우에는 후보서비스의 증가와 개별 웹서비스의 증가에 따라서 점차적으로 수행시간이 늘어남이 관찰되었다. 그리고 정수계획법의 경우에는 가지치기 방법을 이용하더라도 개별 서비스의 증가와 후보서 비스의 증가로 인해 늘어나는 결정변수의 수와 조합 가능한 실행 계획의 수의 급격한 증가로 인해서 가장 많은 수행시간이 소요되었다.

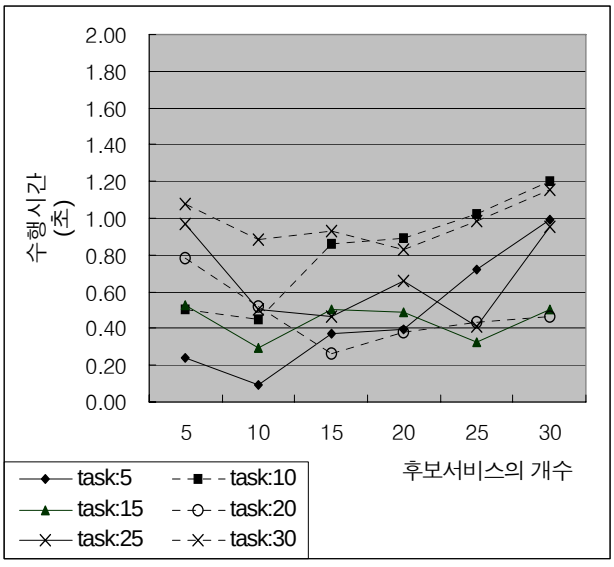


Figure 4. 실험1의 결과.

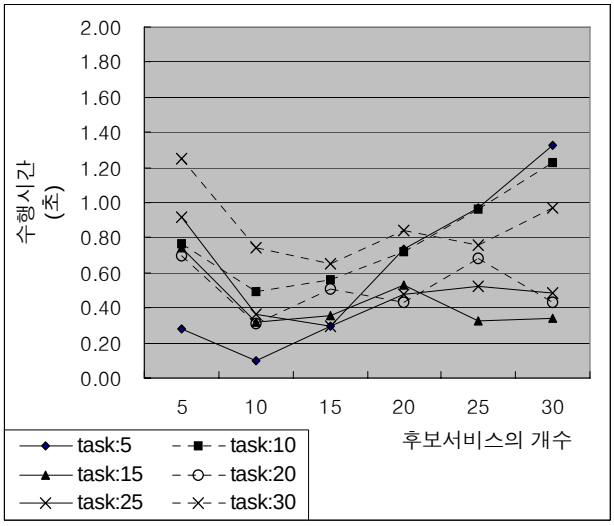


Figure 5. 실험2의 결과.

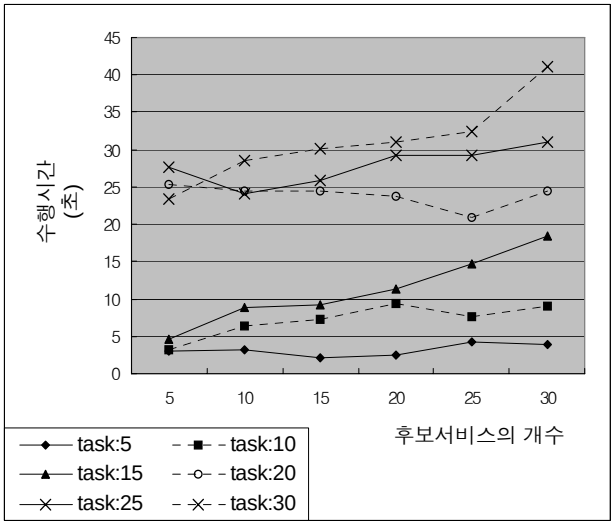


Figure 6. 실험3의 결과.

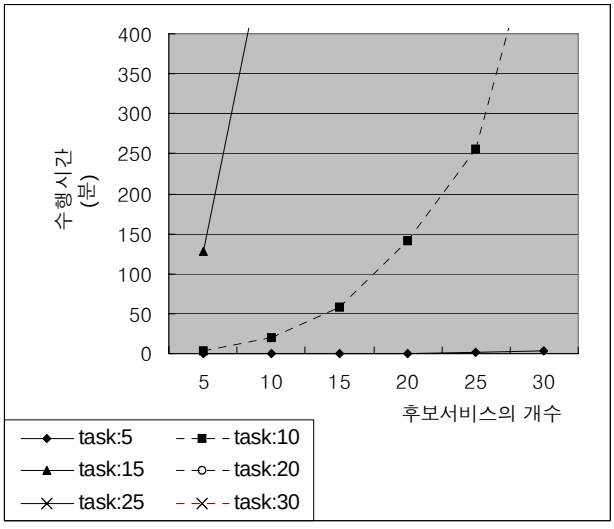


Figure 7. 실험4의 결과.

5.3 동적인 환경을 고려했을 경우의 수행 시간 비교 실험

4.3에서 제시한 동적인 환경을 가정하여 추가적인 실험을 실시하였다. 동적인 환경을 구성하기 위해 서비스 불능상태의 예외 이벤트를 발생시켰으며 예외 처리 핸들러를 통해 실행 계획의 변경을 통해 정상적인 서비스로의 진행이 가능한지를 파악하였다.

5.2의 정적인 환경에서의 실험과 비교하기 위해서 논문에서 제안하는 알고리즘을 이용하는 실험1과 두 가지 동적인 환경에서의 실험인 실험5와 6의 수행 시간을 비교하였다. 실험5는 4.3에서 제시한 첫 번째 예외처리 방법을 통해 개별 서비스의 교체로 실행 가능한 계획을 변경하는 것이고 실험6은 두 번째 방법을 통해 제안한 알고리즘을 통해 예외가 발생하는 시점의 서비스부터 나머지 모든 서비스의 계획까지 실행 가능한 계획을 재수립하는 것이다. 실험은 개별 웹서비스가 10개인 상황에서 각각의 후보서비스의 수를 5에서 30까지 5단위로 증가시키면서 실시하였다.

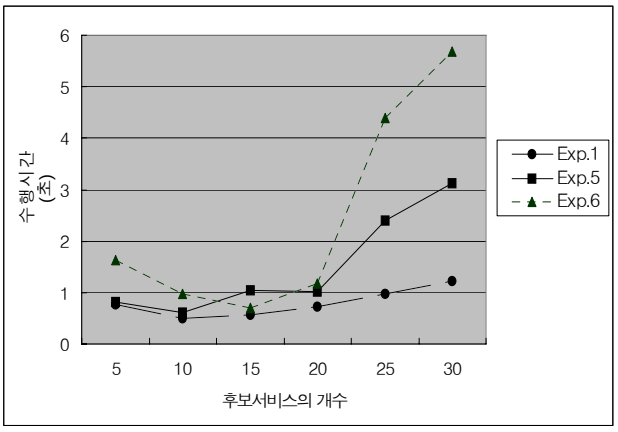


Figure 8. 동적인 환경과 정적인 환경의 비교 실험.

<Figure 8>의 그래프에서 나타나듯이 후보서비스의 수가 많아질수록 수행시간이 증가했다. 또한 예외 시점부터 모든 서비스를 교체하는 방법보다는 실행 계획 리스트의 탐색이나 개별 웹서비스 평가를 이용한 부분적인 웹서비스 교체의 방법이 더 우수하게 나타났다. 실제로 웹서비스의 동적 환경에 대한 예외처리는 웹서비스 운영과 이용 전반에 걸친 정책적인 문제로 제기될 수 있다. 따라서 정책적인 접근과 웹서비스 계약에 관한 약관을 고려하여 두 가지 예외처리 방법이 적절히 혼용되어야 할 것이다.

6. 결 론

본 논문을 통해 품질 정보가 반영된 웹서비스 실행 계획 아키텍처를 제안한다. 그리고 웹서비스의 품질 판단 지표에 대한 선정 및 연구 그리고 제안된 아키텍처 내의 실행 계획의 생성에 있어서 웹서비스 계획 수립 복잡도를 줄이기 위한 제약식 만족 문제 기반의 휴리스틱 알고리즘을 제안한다.

향후 웹서비스의 제공 및 이용에 있어서 적절한 서비스 탐색을 위한 품질 정보가 필요하게 되고 될 것이다. 또한 그런 환경에서 품질 판단 지표의 관리와 무결성 유지가 중요하게 다루어 질 것이다.

품질 정보가 반영된 아키텍처의 제안은 좀 더 실질적이고 적극적인 웹서비스 이용 환경을 구축하는데 도움이 된다. 또한 제약식 만족 문제 컨셉과 휴리스틱 기법의 이용은 사용자의 요구사항을 반영하는 제약 조건에 대하여 민감하게 반응하여 실행 가능한 계획을 신속하고 효과적으로 찾을 수 있는 장점이 있다.

차후의 연구 과제로는 웹서비스 간 컴포지션 가능성이나 관계 특징이 첨부된 알고리즘을 만드는 것이다. 즉, 그러한 관계 특징이나 컴포지션 가능성이 개별 서비스의 발견과 컴포지트 서비스를 형성할 때 이용될 수 있어야 한다. 그리고 웹서비스 간 의미론적인 요소들을 이용한 자동적인 웹서비스 컴포지션에 대한 새로운 알고리즘이 고려되어야 한다. 그러기 위해서는 웹서비스를 기술하는 기존의 언어에 의미론적 기술을 할 수 있는 표준화된 기법들이 첨가되어야 할 것이다. 마찬가지로 웹서비스의 표준에 민감하게 반응하고 대처할 수 있는 적응성과 지속성에 대한 연구도 함께 진행되어야 할 것이다.

참고문헌

Berners-Lee, T., Hendler, J. and Lassila, O.(2001), The Semantic Web, *Scientific America*, **284**(5), 28-37.

Canfora, G., Penta, M. D., Esposito, R. and Luisa Villani, M.(2005), An Approach for QoS-aware Service Composition based on Genetic Algorithms, *GECCO 2005*, 1069-1075.

Cardoso, J.(2002), Quality of Service and Semantic Composition of Workflows, PhD thesis, University of Georgia.

Cardoso, J., Miller, J., Sheth, A. and Arnold, J.(2002), Modeling Quality of Service for Workflows and Web Service Processes, Technical Report #02-002 v2, LSDIS Lab, Computer Science, University of Georgia.

Cardoso, J. and Sheth, A.(2004), Introduction to Semantic Web Services and Web Process Composition, First International Workshop, *SWSWPC 2004*, **3387**, 1-13.

Cardoso, J. and Sheth, A.(2003), Semantic Web Processes: Semantics Enabled Annotation, Discovery, Composition and Orchestration of Web Scale Processes, *Fourth International Conference on Web Information Systems Engineering*, 375-376.

Gram, A.(2004), "Building WS-QoS - compliant Web Services, Technical Report B-04-12, available at http://www.inf.fu-berlin.de/inst/ag-tech/wsqos/.

Kim, Y. G., Youn, B. S. and Lee, S. B.(1999), *Meta Heuristic*, 265-305, Younggimunhwasa.

Limthanmaphon, B. and Zhang, Y.(2003), Web Service Composition with Case-Based Reasoning, Fourteenth Australian Database Conference, Adelaide, Australia, *Conferences in Research and Practice in Information Technology*, 17.

Liu, Y., Ngu, A. H. H. and Zeng, L.(2004), QoS Computation and Policing in Dynamic Web Service Selection, Thirteenth World Wide Web Conference in New Yock City, 67-70.

Martin, D. et al. (2004), OWL-S: Semantic Markup for Web Services, available at http://www.daml.org /services/owl-s/.

Medjahed, B., Bouguettaya, A. and Elmagarmid, A. K.(2003), Composing Web services on the Semantic Web, *The VLDB Journal*, **12**, 333-351.

Nonobe, K., Ibaraki, T.(1998). A tabu search approach for the constraint satisfaction problem as a general problem solver. *European Journal of Operational Research*, **106**, 599-623.

Newcomer, E.(2002), Understanding Web Services, *Addison-Wesley*, 1-32.

O'Sullivan, J., Edmond, D. and Hofstede, A.T.(2002), "What's in a Service?", *Distributed and Parallel Databases*, **12**(2-3), 117-133.

Rao, J. and Su, X.(2004), A Survey of Automated Web Service Composition Methods, First International Workshop, *SWSWPC 2004*, **3387**, 43-54.

Russell, S. and Norvig, P., Artificial Intelligence, A Modern Approach, *Prentice Hall*, 137-155.

Sirin, E., Parsia, B. and Hendler, J.(2004), Filtering and Selecting Semantic Web Services with Interactive Composition Techniques, *IEEE Intelligent System*, **19**(4), 42-49.

Zeng, L., Benatallah, B., Ngu, A. H. H., Dumas, M., Kalagnanam, J. and Chang, H.(2004), QoS-Aware Middleware for Web Services Composition, *IEEE Transactions on Software Engineering*, **30**(5), 311-327.

고 종 명

연세대학교 컴퓨터산업공학과 학사
연세대학교 컴퓨터산업공학과 석사
현재 : 연세대학교 정보산업공학과 박사과정
관심분야 : 웹 서비스, 실시간 데이터 마이닝

김 창 욱

고려대학교 산업공학과 학사
고려대학교 산업공학과 석사
Purdue University 산업공학과 박사
현재 : 연세대학교 정보산업공학과 교수
관심분야 : 정보시스템 설계, RFID기반 물류 시스템, 웹 서비스