한국지능시스템학회 논문지 2008, Vol. 18, No. 2, pp. 175-182

사례 기반 추론을 이용한 서비스 컴포지션 연구

A Study on Service Composition Using Case-Based Reasoning

김건수 · 이동훈 · 박두경 · 이지형
Kunsu Kim, Dong-hoon Lee, Doo-kyung Park and Jee-Hyong Lee

성균관대학교 전자전기컴퓨터공학과

요 약

웹서비스와 상황인식서비스는 사용자의 요구에 따라 많은 종류의 서비스를 제공할 수 있어야 한다. 사용자의 요구는 날로 다양해지고 있고 그에 발맞추어 다양한 종류의 서비스들이 서비스 제공 업체에 의해 새롭게 시장에 등장하고 있다. 하지만 사용자의 새로운 요청에 맞추어 새로운 서비스를 생산하거나 일부 사용자의 요구때문에 새로운 서비스를 만드는 것은 비효율적이다. 그래서 최근에는 유저의 다양한 요청에 보다 유연하게 대응할 수 있는 기술들에 대한 연구가 이루어지고 있다. 그 중 단일 서비스의 조합을 통해 복합서비스를 제공할 수 있는 서비스 컴포지션이라는 기술이 주목을 받고 있다. 하지만, 기존 연구들은 다소 늦은 처리속도로 인해 빠른 응답이 필요한 실시간 상황인식서비스에는 부적합하다. 그래서 본 논문은 사례기반추론을 이용하여 기존 방법보다 빠른 서비스 컴포지션을 구현하는 방법을 제안한다. 그리고 제안하는 알고리즘을 검증하기 위해 사용자에게 이동서비스와 구매정보서비스를 이용하여 사용자의 이동 및 구매 요구를 만족하는 조합을 찾아주는 물건 구매 서비스 서비스를 구현하고 이 구현된 물건 구매 도우미 서비스에 제안하는 알고리즘과 기존 서비스 컴포지션 기법을 적용하여 각 알고리즘의 성능을 비교분석 한다.

키워드 : 사례기반추론, 서비스 조합, 군집화

Abstract

Context-aware service environment should provide many kinds of services according to users' requests. Users want a great variety of services. In response to their demands the service provider should make a new service every time. But making a new service every time may be inefficient even for a small number of users' requests. So, there are studies on how to efficiently support various and complex requests from users. In many researches, service compositions have lately attracted considerable attention. However, existing researches have mainly focused on Web services. So they are not proper to rapidly providing services in response to users' requests, especially in context-aware service environment. This paper proposes a rapid service composition using case-based reasoning. For evaluating the proposed algorithm we implement 'purchasing service agent'. With this system, we compare our algorithm and the existing service composition algorithms.

Key Words : Case-based Resoning, Service Composition, Clustering

1. 서 론

최근 유비쿼터스 컴퓨팅에 대한 많은 연구들이 진행 중이고, 그 중 상황인식 서비스에 대한 연구도 여러 분야에 걸쳐 이루어지고 있다. 이에 따라 다수의 서비스를 효과적으로 사용자의 요구에 맞게 제공하는 기술이 필요하게 되었고, 이를 위해 서비스 컴포지션[1] (service composition) 기술이 개발되었다. 서비스 컴포지션은 다양한 단일서비스(primitive service)들의 조합을 통해 사용자의 요구사항을 만족시키는 것을 목표로 한다. 서비스 컴포지션 기법으로는 Hierachical Task Network (HTN), Model Checking 등을 이용한 방법들이 제안되고 있지만 이런 기법들은 모두 탐색을 기반으로 하고 있어서 서비스의 수가 증가함에 따라 긴 탐색 시간을 갖게 된다. 유비쿼터스 환경에서는 상황인식 서비스의 빠른 응답시간을 요구하기 때문에, 실시간으로 응답이 가능한 빠른 서비스 컴포지션 기법이 필요한데, 이러한 기법들은 상황인식 서비스의 요구사항을 만족시키지 못한다.

본 논문에서는 사례기반추론을 이용한 서비스 매칭 기법을 이용하여 사용자의 요구에 보다 빠르게 응답할 수 있는 방법을 제안한다. 사례기반추론 기법은 크게 검색, 재사용, 수정, 저장 4단계로 나누어진다. 각 과정은 주어진 문제와 가장 일치하는 사례를 검색하는 과정 (이하, 서비스 매칭), 선택된 가장 비슷한 사례를 이용하여 현재 문제를 풀어보는 과정 (이하, 시뮬레이션), 문제가 해결되지 않을 경우 수정하여 새로운 해결방법을 모색하는 과정 (이하, 어뎁테이션), 마지막으로 새로 발견한 해결법을 기억하고 최적화하여 다음 문제 해결에 이용하는 저장 과정으로 이루어져있다. 또

접수일자 : 2007년 11월 18일
완료일자 : 2007년 12월 24일
본 연구는 21세기 프론티어 연구개발 사업의 일환으로 추친되고 있는 정보통신부의 유비쿼터스컴퓨팅 및 네트워크원천기반기술 개발사업의 지원을 받았습니다

한 검색시간 단축을 위해 클러스터링 기법을 이용하여 다양한 문제에 적용 가능한 소수의 사례들을 남기고 그와 비슷한 사례들은 제거하는 사례 관리 기법을 제안한다. 각 과정에 대한 자세한 설명은 3장에서 기술하고, 4장에서는 제안 알고리즘을 적용하여 구현한 물건 구매 도우미 서비스에 대하여 설명한다. 5장에서는 제안하는 알고리즘과 기존 알고리즘의 성능을 비교 검증하고, 끝으로 6장에서는 제안 알고리즘을 상황인식 서비스에 적용시키는 방법과 효과에 대해 기술하고, 향후 연구 진행방향에 대해 논의한다.

2. 관련 연구

하나의 서비스를 단일 서비스 (primitive service)로 정의하고 이러한 단일 서비스들 간의 연계를 고려해서 조합을 한 복합서비스 (composite service)를 생성하는 것이 서비스 컴포지션 기술이다. 이러한 서비스 컴포지션 기술에 대한 연구는 현재 여러 연구그룹에서 진행되고 있다. 대표적인 연구로서 Maryland 대학의 SHOP2[2] 기반의 웹 서비스 컴포지션 시스템이 있는데 이는 계층적 업무 네트워크 (Hierarchical Task Network)[3] 기반의 AI Planning 기술을 적용하여 OWL-S[4]로 의미가 기술된 웹 서비스 컴포지션을 수행하는 것이다.

본 논문에서는 위에서 설명한 서비스 컴포지션을 위해 사례기반추론기법을 사용하였다. 사례기반 추론은 인간의 지적 활동을 모델화한 것으로, 과거문제로부터 얻은 상황 경험이나 지식을 사례 데이터베이스로 구축하여 어떠한 상황이나 문제가 발생하면 기존의 사례 데이터베이스에서 가장 똑같거나 또는 가장 유사한 사례를 선택하여 그 사례가 제시하는 해결책으로 현 문제에 대한 답을 제시한다[5]. 지식을 'IF-THEN'의 규칙 형식으로 기술하는 것이 곤란한 경우에 유효한 추론 방식으로 주목받고 있다. 전문가 시스템은 대부분 규칙을 사용하여 추론하는 규칙 기반의 추론 (rule-based reasoning)을 채용하고 있다. 이 방식은 대상이 되는 문제에 관한 전문가와 상의하여 규칙을 추출하는 작업을 필요로 한다. 사례 기반의 추론을 하려면 규칙을 작성하는 대신에 사례를 수집하여 데이터베이스에 저장하면 된다. 사례기반추론을 실행하는 데 문제가 되는 것은 유사도의 정의와 얻어진 유사한 사례의 해답을 목적하는 문제에 맞추기 위한 수정 방법인데, 이를 위해 서비스 컴포지션에 사례기반추론을 적용한 연구가 있다. 주로 웹서비스를 위한 서비스 컴포지션에 사례기반추론을 적용한 연구들이다[6][7].

하지만 사례기반추론의 경우, 사례 기저에 쌓인 사례의 수가 늘어감에 따라 사례기반추론의 속도가 저하된다. 이를 극복하기 위한 방법으로 사례를 카테고리 별로 나눈 후 인덱싱을 통해 사례를 빨리 찾는 기법과 사례기저에서 사례의 수를 줄이는 연구가 있다. 인덱싱을 통해 사례를 빨리 찾는 기법은 category-exemplar[8]를 이용한 방법이 있다. 유사한 사례를 클러스터링을 이용하여 그룹화 한 후에 그 중에서 대표 사례를 하나 선정하여 인덱스로 이용하는 방법이다. 그리고 사례기저의 수를 줄이는 연구도 많은 연구자들이 클러스터링을 이용한 사례 관리 기법을 제안하였다. 주로 K-means 클러스터링 알고리즘[9]과 K-nn 클러스터링 알고리즘[10]을 이용하여 비슷한 사례끼리 그룹을 형성 후 유사한 사례 중에서 중요한 사례만 남기고 나머지 사례를 지우는 방식으로 접근하였고, 퍼지 클러스터링[11]을 이용하여서 접근하였다. 그리고 클러스터링 이후 Decision Forest[12]를 이용하여 사례에 대한 정보가 부족해도 비슷한 사례를 찾을 수 있는 연구가 있다.

3. 사례기반추론을 이용한 서비스 컴포지션

본 논문에서 제안하는 사례기반추론을 이용한 서비스 컴포지션 기법에 대한 전체적인 구성은 그림 1과 같다.

문제를 입력받으면 서비스 컴포지션 모듈은 사례기저 (Case Base)에서 가장 비슷한 사례를 찾는다. 완전 일치 하는 사례를 찾으면 사례의 해결방법을 그대로 새로운 문제에 사용하지만, 일치하는 사례가 없다면, 가장 비슷한 사례의 해결방법을 적용시켜 본다. 적용불가능하면 문제에 맞게 해결방법을 수정하고, 최적화 과정을 통해 새로운 해결방법이 이상이 없음을 파악한 뒤 해결법을 제공하고, 사례기저에 현재 문제와 함께 저장한다.

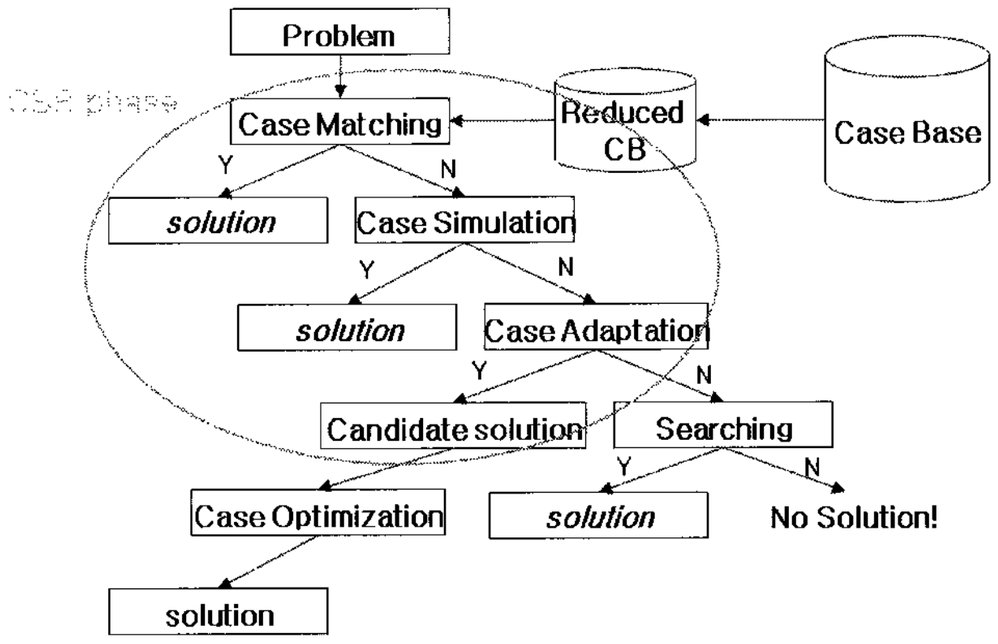


그림 1. 사례기반추론을 이용한 서비스 컴포지션
Fig. 1. Service composition using CBR

3.1 사례기반추론(Case-Based Resoning)

3.1.1 서비스와 사례

서비스는 단일서비스 (Primitive service)와 복합서비스 (Composite Service) 두 종류가 있다. 우선 단일서비스는 자원 (resource)을 이용하여 한 개 이상의 상태(state)를 변화시킬 수 있는 서비스를 말한다. 자원과 상태 정보는 Attribute-value 쌍으로 기술된다. 그리고 단일서비스는 선행상태 (precondition)와 후행상태 (postcondition)를 기본적으로 갖고 있다. 선행상태는 단일서비스를 수행하는데 있어서 만족해야하는 조건들의 집합이고, 후행상태는 단일서비스가 수행됨으로서 변화하는 정보의 집합이다. 그리고 복합서비스는 서비스 컴포지션의 결과로 생성되는 서비스이고, 특정 목표를 달성할 수 있는 단일서비스의 조합이며, 선행상태와 후행상태를 기본정보로 갖고 있다.

사례는 단일서비스와 복합서비스를 이용하여 실제 적용된 결과로 서비스와 마찬가지로 선행상태와 후행상태 정보와 해결한 문제에 대한 정보를 포함하고 있다.

3.1.2 매칭(Matching)

매칭은 주어진 문제의 속성값을 사례기저에 들어 있는 사례의 속성 값과 비교하는 과정이다. 비교를 위해서는 문제 도메인에 맞는 유사도 척도 (similarity measure)에 대

한 정의가 필요하다. 정의된 유사도 척도를 이용하여 문제의 현재 상태와 그리고 목표 상태의 각 속성 값을 사례의 선행상태와 후행상태와 비교한다.

매칭 결과 현재 문제와 일치하는 사례 혹은 가장 비슷한 사례가 선정되고, 일치하는 사례일 경우 사례의 해결방법이 주어진 문제의 해결방법으로 그대로 사용되나, 선정 된 사례가 가장 비슷한 사례일 경우 다음 과정인 시뮬레이션 단계로 넘어가게 된다.

3.1.3 시뮬레이션(Simulation)

시뮬레이션은 사례기반추론 Reuse 단계에 해당한다. 이 과정을 통해 가장 유사한 사례의 해결방법이 현재 문제에 그대로 적용 가능한지 알 수 있다. 다음과 같이 문제의 현재상태 (Sc)에서 사례의 해결방법인 복합서비스 (Csi)를 적용하여 목표상태에 도달할 수 있는지 검증하는 과정이다. 시뮬레이션 결과로 사례의 해결법이 현재 문제에도 적용가능하면 현재 문제에 대한 해결법으로 그대로 사용하고 그렇지 않으면 다음 과정인 어뎁테이션 단계로 들어가게 된다.

3.1.4 어뎁테이션(Adaptation)

어뎁테이션 단계는 사례기반추론의 revise과정에 해당한다. 이 단계에서는 주어진 문제와 가장 비슷한 사례의 해결법이 문제에 직접 적용이 불가능 할 때, 해결법을 주어진 문제에 맞게 수정하는 작업을 한다.

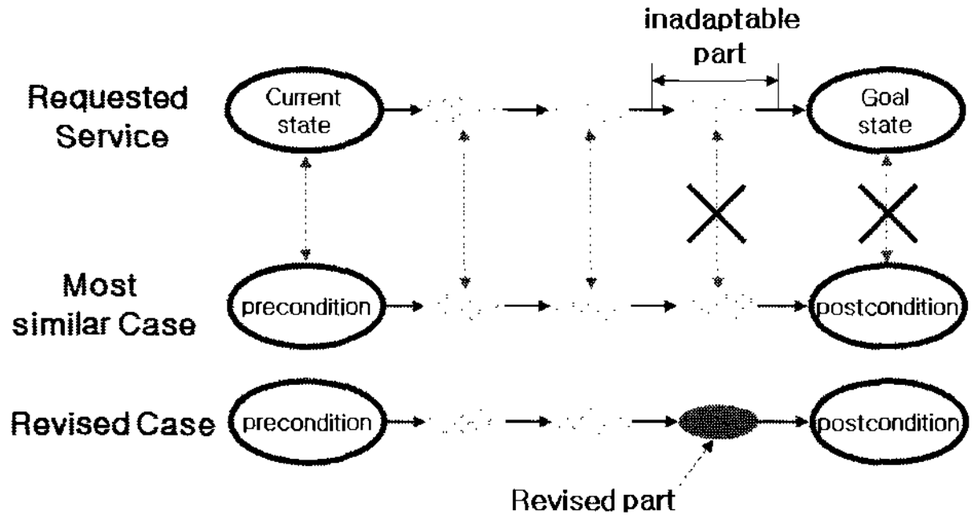


그림 2. Adaptation 모식도
Fig. 2. Diagram of adaptation

먼저 가장 비슷한 사례의 해결법을 순차적으로 현재 상태 (Sc)에서 부터 적용시킨다. 그 후 중간에 조건을 만족시키지 못하는 상태 (Si)에 도달하게 되면 바로 직전 상태 (Si-1)로 복귀 후 목표상태 (Sg)로 도달할 수 있는 사례를 매칭, 시뮬레이션 과정을 통해 재귀적으로 찾는다. 사례를 찾은 경우, Sc부터 Si-1까지 적용한 서비스와 Si부터 Sg까지 새로 발견한 서비스를 조합하여 새로운 조합서비스를 생성한다. 하지만 사례를 찾기 못한 경우, Si부터 Sg까지 탐색 과정을 이용하여 새로운 조합서비스를 생성한다. 이렇게 새롭게 생성된 해결법은 마지막 단계인 최적화 과정으로 넘어간다.

특정 자원의 부족으로 탐색 과정을 이용해도 해결법을 찾지 못하는 경우가 발생할 수도 있다. 그런 경우에는 중간 상태인 Si-1에서 그 직전상태 (Si-2)로 돌아가서 Si-2부터 Sg까지의 해결법을 찾는다. 이때 해결법을 찾는 과정은 위의 과정과 동일하다.

계속해서 해결법을 찾지 못하게 되면, 결국 중간 상태가 Sc까지 돌아오게 된다. 이런 경우에는 Sc부터 Sg까지 탐색 과정을 이용하여 해결법을 찾아본 후, 그래도 해결법을 찾지 못하면 최종적으로 해결 불가 메시지를 사용자에게 전달한다.

3.1.5 최적화(Optimization)

최적화과정은 어뎁테이션 과정의 일부이다. 사례기반추론의 특징상 비슷한 사례의 해결법을 새로운 문제의 해결에 그대로 적용하는데 오류는 발생하지 않지만, 자원의 낭비가 발생할 수도 있다. 따라서 최적화 과저을 통해 최종적으로 결정된 해결법의 오류뿐 만 아니라 해결법으로 결정된 서비스 컴포지션에 불필요한 서비스가 없는지 검사한다.

3.2 탐색(Searching)

탐색과정은 사례기반추론이 갖는 한계를 극복하기 위한 과정이다. 사례기반추론은 비슷한 사례가 사례기저에 존재하지 않을 경우 정상적인 추론을 할 수 없는 단점이 있다. 시스템이 관리자 없이 순수하게 자동으로 동작하기 위해서는 어떤 경우라도 해결책을 제시할 수 있어야 한다. 그래서 우리는 사례기반추론에 HTN과 Model Checking과 같은 기존 서비스 컴포지션에서 자주 이용한 기법인 탐색 과정을 추가하였다.

보다 효율적인 탐색을 위해 중요요소 (Critical Property)라는 속성을 두어, 서비스 컴포지션시 사용자의 특정 요구를 고려할 수 있다. 사용자의 현재 상태와 사용자의 목표상태를 비교하면 사용자가 원하는 서비스의 종류를 알 수 있고, 서비스 중 가장 중요요소에 영향을 많이 주는 서비스를 먼저 고려하여 나머지 서비스를 조합하면 보다 사용자의 요구에 맞는 조합서비스를 빨리 찾을 수 있다.

3.3 사례기저관리(Case Base Management)

사례기반추론은 사례기저에 쌓은 사례의 양이 늘어날수록 보다 정확한 해결법을 찾을 수 있지만, 역설적으로 많은 사례들과 현재 문제를 비교해야 하므로 가장 비슷한 사례를 골라내는 검색시간이 늘어나게 된다. 많은 사례들 중 일부는 서로 그 속성 값이 비슷한 경우가 있다. 이러한 비슷한 사례들 중 다른 문제에도 쉽게 적용될 수 있는 좋은 사례를 남기고 그와 비슷한 다른 사례를 지우면 추론의 정확도를 유지하면서 속도를 높일 수 있다.

본 논문에서는 사례기저를 관리하기 위해 클러스터의 직경을 제한하는 클러스터링 기법 (Limited Radius Clustering 이하 LRC)을 이용하였다.

비슷한 사례는 좌표 상에 모여서 나타나게 된다. 그래서 LCR는 다음 과정을 통해 사례기저에 있는 사례의 양을 감소시킨다. 먼저 지정된 거리 (Limited Radiuds) 안에 있는 사례들을 서로 비슷한 사례로 간주해서 하나의 클러스터로 묶은 뒤 그 중 클러스터의 중심에 가까운 사례를 우선적으로 남기고 그 사례의 일정 거리 (Limited Distance) 안에 있는 사례를 지운다. 그리고 클러스터에 남아 있는 사례들 중 일부를 무작위로 선택하여 역시 그 사례의 일정 거리 안에 있는 사례를 지우는 것을 반복하여 클러스터 내에 서로 일정거리 이상 떨어져 있는 사례들만 남기는 방식이다. 클러스터의 직경을 제한했기 때문에 일부 사례는 어떤 클러스터에도 속하지 않는 경우가 발생한다. 이러한 사례는 다른 사례들과 많이 다르기 때문에 추론의 성능을 향상시킬 수 있으므로 무조건 사례기저에 남기는 방식을 택했다. LCR을 이용한 2차원 사례 데이터에 대한 실험 결과는 그림 3과 같다.

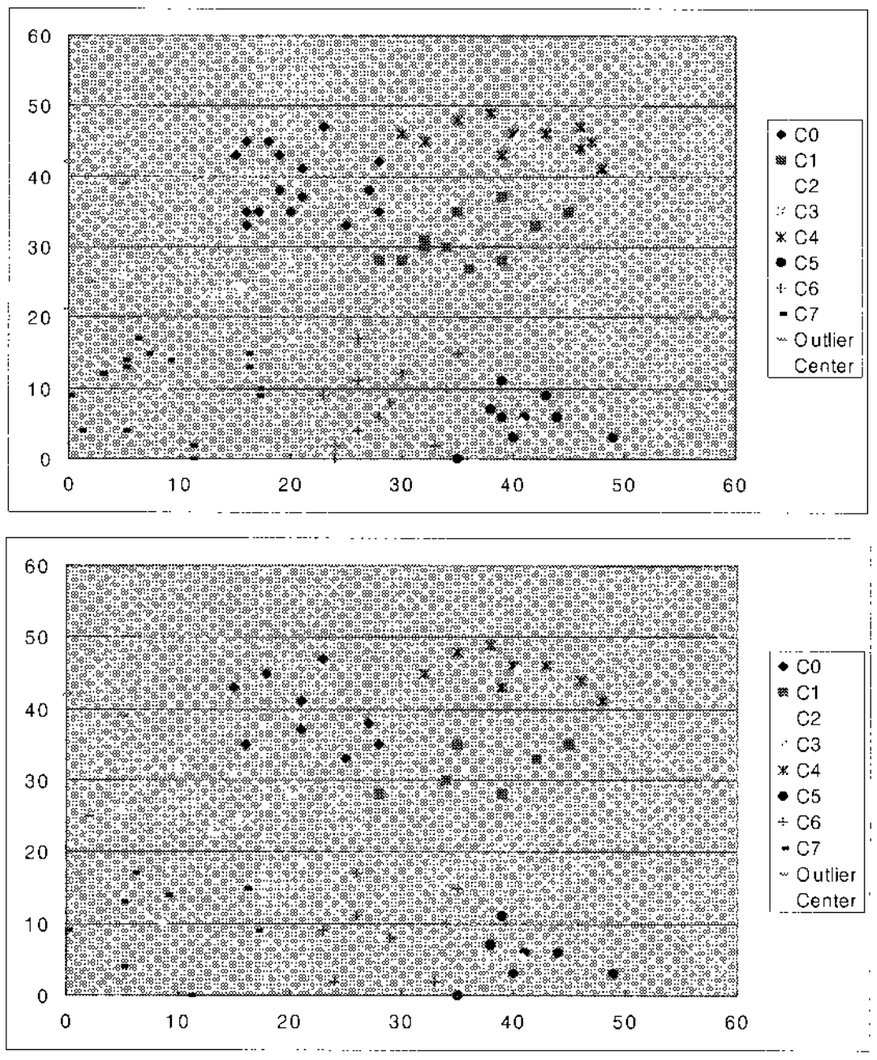


그림 3. LRC 수행 전 (상)과 수행 후 (하)
Fig. 3. Before LRC (top) & Afer LRC (bottom)

4. 구 현

본 논문에서는 제안하는 알고리즘의 검증을 위해 물건 구매 도우미 서비스를 구현하였다. 물건 구매 도우미 서비스는 현재 사용자의 위치에서 사용자가 구매하고자하는 물건을 입력 받아 사용자의 다음 스케줄까지 남은 시간과 사용자가 이용 가능한 금액을 고려하여 구매장소를 선택하고 구매장소까지 갈 수 있는 이동수단을 결정해주는 프로그램이다.

4.1 서비스

물건 구매 도우미 서비스는 최단거리 이동서비스와 최저가 검색 서비스 두 가지 종류의 서비스를 지원한다. 예를 들어, 사용자가 책과 같은 특정 물건을 구매해서 다음 스케줄의 목적지로 제한된 시간 안에 현재 보유한 금액을 고려하여 이동을 하고자 할 때, 가장 최적의 루트를 계산하여 사용자에게 제공하는 것이 그 목적이다.

4.1.1 이동 서비스

이동서비스는 사용자에 의해 주어진 조건에 따라, 현재 장소에서 목적지까지 지하철 또는 택시를 이용한 비용 및 시간을 계산한다. 실험의 편의를 위해서 아래와 같이 가정하였다.

- 이동 가능한 도시는 지하철역을 기준으로 서울/경기 전 지역에서 고르게 20개의 역을 지정하였다.
- 지하철을 타고 이동하는 경우 비용 계산 및 이동 시간 계산의 편의를 위해 갈아타는 경우는 배제하였고, 출발점에서 목적지까지의 직선거리를 지하철의 평균 속도(40km/h)로 달린다고 가정하여 이동하는데 걸린 시간을 계산하였다. 비용은 거리와 상관없이 현재 지하철 기본요금인 1000원으로 고정하였다.
- 택시를 타고 이동하는 경우, 출발점과 목적지까지의 직선거리에 지하철 평균 속도의 두배(80km/h)로 달린다고 가정하여 이동하는데 걸린시간을 계산하였다. 비용은 현재 국내 택시요금을 계산 방식인 거리-시간 병산제에서 택시가 15km 미만의 저속으로 달릴 때 추가적으로 발생하는 비용은 무시하고 계산하였다.

4.1.2 구매 서비스

구매서비스는 사용자에 의해 주어진 조건 및 사용자가 구매를 원하는 물건에 따라 구매 장소를 결정한다. 서비스의 구현을 위해 5종류의 책과 2종류의 게임을 판매하는 가상의 상점을 각각 10개씩 가정하였다. 각 상점의 위치 및 판매 물품에 대한 표가 아래 나타나있다.

표 1 서점정보
Table 1. Bookstore information
(품목가격단위 : 천원)

	서비스 이름	위도	경도	취급 품목				
				책1	책2	책3	책4	책5
1	교보문고1	37.342	126.584	15	9	35	22	-
2	교보문고2	37.305	127.060	15	9	35	22	-
3	교보문고3	37.295	127.014	-	9	-	22	5.5
4	영풍문고1	37.342	126.584	-	8.5	38	20	4.5
5	영풍문고2	37.295	127.014	13	8.5	38	20	4.5
6	반디& 루니스1	37.303	127.035	-	9	-	25	5.5
7	반디& 루니스2	37.342	126.584	-	9	37	25	5.5
8	리브로1	37.332	126.561	11	-	35	-	4.5
9	리브로2	37.295	127.014	-	8.5	35	-	4.5
10	리브로3	37.301	126.525	11	-	35	23	4.5

표 2 게임기 총판 정보
Table 2. Game Shop information
(품목가격단위 : 천원)

	서비스 이름	위도	경도	취급 품목	
				NDS	PSP
1	용산전자랜드1	37.315	126.575	156	269
2	국제전자상가1	37.294	127.005	158	270
3	테크노마트1	37.322	127.041	140	260
4	하이마트1	37.294	126.534	-	254
5	NDS마트1	37.315	126.575	145	-
6	용산전자랜드2	37.315	126.575	149	260
7	국제전자상가2	37.294	127.005	-	271
8	테크노마트2	37.322	127.041	149	-
9	하이마트2	37.294	126.534	-	258
10	NDS마트2	37.345	127.024	155	-

4.2 사례기반추론

물건 구매 도우미 서비스에 사례기반추론을 사용하기 위해서는 기본적으로 충분한 양의 사례를 사례기저에 보유하고 있어야 하기 때문에 임의로 100개에서 1000개의 사례를 생성하여 구현하였다.

4.2.1 문제 및 사례

물건 구매 도우미 서비스에서 사용자가 해결하기 원하는 문제는 <시작위치, 목표위치, 가용시간, 보유금액, 구매품목, 중요요소>와 같은 속성으로 구성되고, 각 속성의 특징은 다음과 같다.

- 시작 위치 및 목표 위치 : 각 위치의 위도/경도 정보
- 가용시간 : 사용자의 가용시간
- 보유금액 : 사용자가 보유한 금액
- 구매품목 : 구매해야하는 모든 물건 목록
- 중요요소 : 사용자가 설정한 중요 고려 요소. 시간 혹은 비용

물건 구매 도우미 서비스의 사례는 문제의 속성에 해결법이 추가된다. 해결법은 사례에서 해결하기 위해 생성된 복합서비스의 스케줄이고 이용서비스가 바로 이에 해당한다. 사례의 구성은 <시작위치, 목표위치, 소모시간, 소모비용, 구매품목, 중요요소, 이용서비스> 이고, 각 속성의 특징은 다음과 같다.

- 시작 위치 및 목표 위치 : 각 위치의 위도/경도 정보
- 소모시간 : 전체 복합서비스 동안 소모한 시간 정보
- 소모비용 : 전체 복합서비스 동안 소모한 금액 정보
- 구매품목 : 전체 복합서비스 동안 구매한 모든 물건 목록
- 중요요소 : 사용자가 요청한 중요 고려 요소. 시간 혹은 비용
- 이용서비스 : 전체 복합서비스 동안 이용한 이동서비스 및 구매서비스 목록

4.2.2 사례 비교

문제가 주어졌을 때, 조건이 일치하는 사례 혹은 가장 비슷한 사례를 선택하기 위해서는 유사도 척도(similarity measure)가 필요하다. 식(1)은 시간 유사도를 구하는 식이고, 식(2)는 비용 유사도를 구하는 식이다.

$$\text{시간유사도} = 1 - \frac{\text{사례소모시간}}{\text{문제보유시간}} \quad (1)$$

$$\text{비용유사도} = 1 - \frac{\text{사례소모비용}}{\text{문제보유금액}} \quad (2)$$

구매품목 속성에 대한 유사도도 식(3)과 식(4)와 같은 수식을 이용하여 계산하였다.

$$\text{사례 구매품목의 수} >= \text{문제 구매품목의 수}, \quad \frac{\text{사례구매품목과 일치하는 문제구매품목수}}{\text{사례구매품목 수}} \quad (3)$$

$$\text{사례 구매품목의 수} < \text{문제 구매품목의 수}, \quad \frac{\text{문제 구매품목과 일치하는 사례 구매품목수}}{\text{사례 구매품목 수}} \quad (4)$$

각 속성이 유사도에 모두 동일한 수준으로 영향을 줄 수 있도록 모든 속성 값을 0에서 100사이의 값을 갖도록 평준화하였다. 최종적으로 문제와 사례 사이의 유사도는 식(5)와 같이 계산된다.

$$\text{유사도 } S = \text{거리유사도} + \text{시간유사도} + \text{비용유사도} + \text{구매품목유사도} \quad (5)$$

4.2.3 탐색

탐색으로 문제를 해결할 경우, 사용자가 입력한 중요요소에 따라 문제를 푸는 방식이 비용 우선과 시간 우선 둘로 나뉜다. 두 방법 모두 전체 상점 목록에서 문제에서 구매해야하는 물건들을 보유한 상점들의 목록을 뽑는다. 그 후 비용 우선의 경우는 선택 된 모든 상점을 조합하여 출발 지점부터 목적지까지의 지하철과 택시를 이용한 이동을 모두를 고려해서 소모비용 및 소모시간을 계산한다. 그 중 허용된 비용 및 시간을 초과하지 않는 조합 중에서 가장 적은 비용이 든 조합을 선택한다.

시간 우선의 경우는 비용 우선의 경우와 동일한 과정을 거친 후 모든 루트에 대한 계산이 완료 되면, 그 중 허용된 비용 및 시간을 초과하지 않는 조합 중에서 가장 적은 시간이 든 조합을 선택한다.

4.2.4 최적화

그림 4(a)는 사례 어뎁테이션 적용 결과 하나의 상점에서 2개의 물건을 구매하는데 2번 방문하는 스케줄이 발생한 상황이다. 홍대입구에서 출발하여 PSP, NDS, Book1을 구매 후 왕십리까지 가는 문제를 사례 매칭을 통해 비슷한 사례를 찾았다. 이 경우 사례의 해결법을 그대로 적용하게 되면 이동서비스 (점선) 중에 허용된 시간을 초과하게 되기 때문에 물건 구매 도우미는 바로 전 단계로 돌아가 더 가까운 상점인 용산을 찾게 되었고, 용산에서 NDS를 구매 후 왕십리로 가는 새로운 스케줄을 생성하였다. 이 과정에서 불필요한 두번의 방문이 발생하게 되었고, 이를 해결하기 위해 최적화 과정에서 새로 생성된 스케줄의 경우에는 물건을 구매하는 상점의 중복이 생기면 스케줄 계산시 하나의 구매서비스로 인식하여 그림 4(b)와 같이 다시 스케줄을 계획한다.

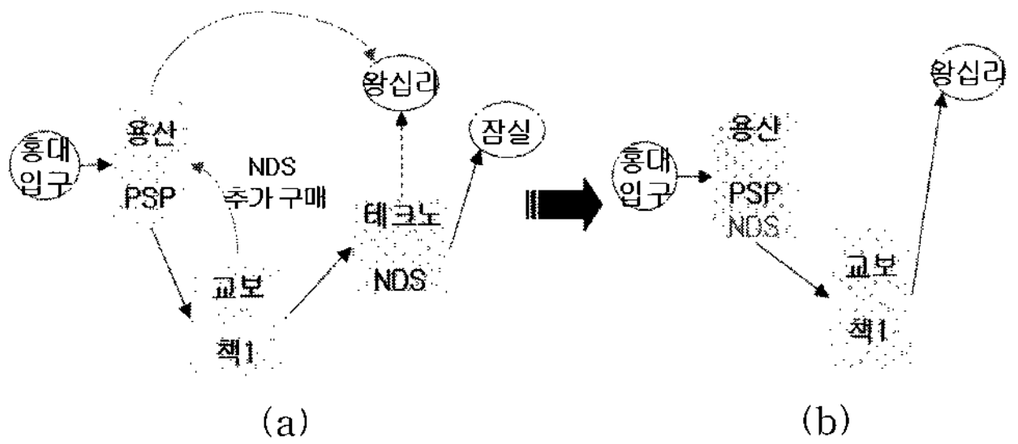


그림 4. 사례 어뎁테이션(a)과 최적화 결과(b)
Fig. 4. Case adaptation(a) & Optimization(b)

5. 실험 및 분석

본 논문에서 제안하는 알고리즘의 성능을 평가하기 위해 기존에 많이 사용하는 탐색기법을 이용한 서비스 컴포지션 기법과 비교하였다. 4장에서 정의한 20개의 서비스와, 임의로 생성한 1000개의 사례를 갖고 결과를 비교하였다.

먼저 문제 해결 시간을 각각 사례기반추론, LRC를 이용한 사례기반추론, 탐색을 이용한 추론 방법을 이용하여 측정하였다. 실험을 위해 15개의 문제를 생성하였는데, 문제의 특징은 다음과 같다.

- 1번~5번문제 : 1000개의 사례에서 임의 추출
- 6번~10번문제 : 1000개의 사례에서 추출 후 사례 정보의 속성을 임의로 한 개 변경
- 11번~15번문제 : 1000개의 사례에서 추출 후 사례 정보의 속성을 임의로 두 개 이상 변경

표 3 사례기반추론 결과
Table 3. Result of CBR

(단위 : 초)

	매칭	시뮬레이션	어뎁테이션	최적화	탐색	총 시간
1	1.494	-	-	-	-	1.494
2	1.495	-	-	-	-	1.495
3	1.498	-	-	-	-	1.498
4	1.498	-	-	-	-	1.498
5	1.497	-	-	-	-	1.497
6	1.490	0.041	-	-	-	1.531
7	1.494	0.037	-	-	-	1.531
8	1.495	0.037	-	-	-	1.532
9	1.492	0.039	-	-	-	1.531
10	1.491	0.044	-	-	-	1.535
11	1.497	0.037	4.451	-	4.144	10.129
12	1.494	0.041	-	-	-	1.535
13	1.492	0.041	1.013	-	-	2.546
14	1.496	0.036	-	-	-	1.532
15	1.490	0.044	-	-	-	1.534
					평균	2.161

초기 사례기저의 사례의 수가 1000개를 그대로 이용하여 사례기반추론을 한 경우, 그리고 LRC를 통해 사례의 수를 188개로 줄여서 사례기반추론을 한 경우, 그리고 탐색기법을 이용하여 문제를 풀었다. 사례기반추론을 이용하여 문제를 푼 결과가 표 3에 그리고 사례기반추론과 LCR을 이용하여 문제를 푼 결과가 표 4에 나타나 있다.

표 3에서는 문제를 푸는 각 단계별로 시간을 측정한 결과이다. 1번부터 5번 문제까지는 사례 매칭으로만 해결법을 찾은 것을 알 수 있고, 6번 부터 10번까지는 비슷한 사례를 찾아 시뮬레이션 결과 사례와 동일한 해결법이 적용가능한 경우다. 11번은 비슷한 사례의 해결법이 적용하는 것이 불가능해서 결국 탐색을 이용하여 문제를 푼 경우라고 볼 수 있다. 13번의 경우는 비슷한 사례의 해결법을 일부만 수정해서 문제가 풀린 것을 알 수 있다.

표 4 LRC를 이용한 사례기반추론 결과
Table 4. Result of CBR using LRC

(단위 : 초)

	매칭	시뮬레이션	어뎁테이션	최적화	탐색	Total
1	0.303	0.041	-	-	-	0.344
2	0.300	-	-	-	-	0.300
3	0.308	-	-	-	-	0.308
4	0.306	-	-	-	-	0.306
5	0.306	-	-	-	-	0.306
6	0.305	0.047	-	-	-	0.352
7	0.301	0.038	-	-	-	0.339
8	0.302	0.044	0.991	-	-	1.337
9	0.308	0.043	-	-	-	0.351
10	0.301	0.039	-	-	-	0.340
11	0.306	0.044	4.451	-	4.190	8.991
12	0.304	0.038	1.422	-	4.003	5.767
13	0.303	0.042	1.013	-	-	1.358
14	0.301	0.045	0.458	0.101	-	0.905
15	0.305	0.043	-	-	-	0.348
					평균	1.443

사례기반추론을 이용한 실험과 비교했을 때 LCR를 이용한 사례기반추론 결과는 매칭 단계에서 많은 속도의 개선이 이루어졌음을 알 수 있다. 이는 사례의 수가 줄었기 때문에 그 만큼 각 사례에 대한 유사도를 측정하는 시간이 줄어들었기 때문이다. 하지만 사례의 수가 줄었기 때문에 11번부터 15번까지 비슷한 사례를 찾을 수 없는 경우가 비교적 자주 발생할 수 있기 때문에 사례의 어뎁테이션이 더 자주 발생한다.

표 5 알고리즘 별 비교
Table 5. Agorithms comparison

(단위 : 초)

	CBR	CBR+LRC	Search
동일문제	1.496	0.313	4.919
수정문제 (속성 1개 수정)	1.532	0.544	4.788
수정문제 (속성 3개 수정)	3.455	3.474	4.684
평균	2.161	1.443	4.797

표 5는 각각의 실험에 대한 비교이다. 평균적으로 사례기반추론 알고리즘이 탐색을 이용한 알고리즘보다 좋은 성능을 보인다. 사례와 동일한 문제와 조금 문제를 수정한 경우는 사례기반추론 알고리즘이 탐색보다 3배 정도의 더 빠른 속도를 보이고, LRC를 이용한 사례기반추론의 경우 10배를 넘는 차이를 보이고 있다. 하지만 수정이 많이 된 문제의 경우 사례기반추론이나 LRC를 이용한 사례기반추론의 경우도 탐색에 가까운 결과를 보여주고 있다. 많이 수정한 문제라고 해도 비슷한 사례를 찾을 확률이 존재하기 때문에 탐색 알고리즘만을 이용한 것 보다는 나은 1초 정도 더 빠른 결과를 보여준다.

마지막으로 제안하는 알고리즘을 통해 생성한 복합서비스의 결과를 비교하였다. LCR을 이용한 사례기반추론으로 첫 번째 실험에서 사용한 15개의 문제를 풀어 그때 생성된 복합서비스의 총 소모비용과 소모시간을 같은 문제를 탐색으로 풀은 경우와 비교한 결과가 표 6에 나타나있다.

표 6 알고리즘 별 복합서비스 결과 비교
Table 6. Algorithms comparison of service's result

	LRC+CBR		Searching		Comparison	
	Time(분)	Money(원)	Time(분)	Money(원)	ΔT	ΔC
1	46	38800	46	38800	0	0
2	41	44000	41	44000	0	0
3	93	87000	73	87000	20	0
4	19	20200	19	20200	0	0
5	82	60000	82	60000	0	0
6	50	46000	50	46000	0	0
7	106	157000	91	154500	15	2500
8	84	227000	70	227000	14	0
9	121	265000	121	265000	0	0
10	65	272000	65	272000	0	0
11	78	288000	78	288000	0	0
12	130	288000	130	288000	0	0
13	32	400900	32	400900	0	0
14	36	424000	36	424000	0	0
15	43	425300	43	425300	0	0

총 15개의 문제 중 사례기반추론으로 문제를 풀었을 때

와 탐색 기법으로 문제를 풀었을 때 해결법이 다르게 나온 경우는 3번으로 전체 문제의 20%에 해당한다. 그리고 해결법이 서로 다른 문제들의 경우, 두 알고리즘간 해결법의 차이는 평균적으로 탐색으로 얻은 최적화 된 해결법의 21%에 해당하는 시간이 더 소모 되었으며, 비용은 약 1.6% 더 사용되었다. 이는 전체적으로 놓고 봤을 때, 사례기반추론을 이용한 결과가 최적화된 해결법과 큰 차이가 없음을 의미한다.

6. 결론 및 향후 연구

본 논문에서는 빠른 복합 서비스 제공을 위해 사례기반추론을 이용한 서비스 컴포지션 기법을 제안했다. 기존 탐색을 이용한 서비스 컴포지션 기술과는 달리 사례기반추론을 이용한 서비스 컴포지션은 간단히 과거의 사례만 비교하면 되므로 서비스의 수가 증가하더라도 복합서비스를 생성하는데 큰 영향을 받지 않는다는 장점이 있다. 하지만 사례의 수가 증가함에 따라 비슷한 사례를 찾는 시간이 오래 걸리는 문제가 발생하기 때문에, 우리는 추론 능력은 유지하면서도 사례의 수를 줄일 수 있는 직경제한 클러스터링(LRC) 기법을 제안하였다. LRC를 이용하면 가장 쉽게 사용될 수 있는 사례만을 남기고 그와 비슷한 사례를 지움으로서 사례의 수를 줄일 수 있다. 또한 사례기반추론의 성능에 가장 큰 영향을 미치는 것이 바로 수정 및 적용 단계이다. 우리는 사례 해결법의 수정을 최소화 하기 위해 사례 해결법을 현재 문제에 적용 불가능한 부분을 알아내어 그 부분만 수정하는 방법을 제안했다. 우리가 제안한 방법은 해결이 불가능한 상황까지 고려하여 점점 수정하는 부분을 확장해가면서 다시 사례기반추론을 적용하여 재귀적 문제를 풀어 나갈 수 있다. 이는 수정 때문에 발생 할 수 있는 문제풀이 시간을 가급적 줄여주는 효과가 있다.

본 논문에서 제안하는 방법은 기존의 탐색을 이용한 서비스 컴포지션보다 평균적으로 보다 나은 성능을 보여주었다. 하지만 사례기저에서 비슷한 사례를 찾을 수 없는 경우에는 오히려 탐색을 이용한 방법보다 문제를 푸는 시간이 더 오래 걸리는 모습을 볼 수 있었다. 앞으로 이를 해결하기 위해 LRC를 이용하여 사례기저를 축소시킬 때, 다음에 더 뽑힐 확률이 높은 사례를 남기는 방법에 대한 연구를 진행할 예정이다. 그리고 세번째 실험을 통해 사례기반추론에 의해 비슷한 사례의 해결법을 그대로 가져올 경우 비록 주어진 문제는 풀 수 있지만, 그 해결법이 최선의 방법이 아닐 수도 있다는 것을 발견했다. 그래서 다음에는 일정 간격으로 사례기반에 들어 있는 해결법이 각 사례에 최적화 된 것인지 검사를 한 후에 최적 해결법이 아닌 경우는 탐색을 통해 최적 해결법을 구한 후 교체를 하는 과정을 추가할 예정이다.

참 고 문 헌

[1] 김재홍, 하영국, 박천수, 장민수, 이미경, 윤여훈, 손주찬, "URC를 위한 명령 분석 및 서비스 계획 기술," *전자통신 동향 분석*, 제20권, 2호, pp. 67-75, 2005.

[2] D. S. Nau, "SHOP2 : An HTN Planning System," *Journal of Artificial Intelligence Research*, pp. 379-404, 2003.

[3] K. Erol, J. Hendler, D. S. Nau, "UMCP : A Sound and Complete Planning Procedure for Hierarchical Task-Network Planning," *Proceeding of the 2nd International Conference on Aitificial Intelligence Planning Systems*, pp. 95-102, 1994.

[4] OWL-S Home Page, "http://www.daml.org/services/owl-s/1.1/overview," 2004.

[5] 이재규, *전문가시스템 : 원리와 개발*, 법영사, 2004.

[6] O. Graciela Fragoso Diaz, R. Santaolay Salgado, I. Solis Moreno, G. Rodriguesz Ortiz, "Searching and Selecting Web Services using Case-based Reasoning," *ICCSA*, Vol. 3983, pp. 50-57, 2006.

[7] B. Limthanmaphon, Y. Zhang, "Web Service Composition with Case-based Reasoning," *Proceedings of the 14th Australasian database conference*, Vol. 17, pp. 201-208, 2003

[8] B. W. Porter, R. E. Bareiss, "PROTOS : An experiment in knowledge acquisition for heuristic classification tasks," *Proceedings of the 1st International Meeting on Advances in Learning*, pp. 159-174, 1986.

[9] N. Arshardi, I. Jurisica, "Data Mining for Case-Based Reasoning in High-Dimensional Biological Domains," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, pp. 1127-1137, 2005.

[10] Z. Li, Y. Liu, F. Li, S. Yang, "Case Base Maintenance based on Outlier Data Mining," *Proceedings of the 4th International Conference on Machine Learning and Cybernetics*, Vol. 5, pp. 2861-2864, 2005.

[11] 현우석, "퍼지 클러스터링을 이용한 사례기반 추론의 성능 개선," *한국퍼지및지능시스템학회 학술대회지*, pp. 100-103, 2002.

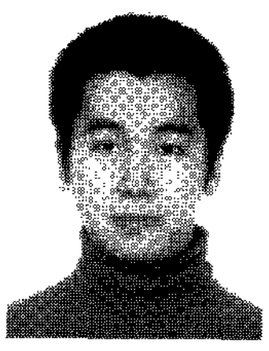
[12] Q. Yang, J. Wu, "Enhancing the Effectiveness of Interactive Case-based Reasoning with Clustering and Decision Forests," *Applied Intelligence*, Vol. 14, pp. 49-64, 2001.

저 자 소 개

김건수(Kunsu Kim)
2007년 : 성균관대학교 컴퓨터공학과 학사
2007년~현재 : 성균관대학교 전기전자컴터공학과 석사과정

관심분야 : 지능시스템, 시맨틱 웹, 온톨로지

E-mail : kkundi@skku.edu



이동훈(Dong-hoon Lee)
2005년 : 성균관대학교 컴퓨터공학과 학사
2007년 ~ 현재 : 성균관대학교 전기전자컴퓨터공학과 석사과정

관심분야 : 지능시스템, 기계학습, 시맨틱 웹

E-mail : idoun@skku.edu

이지형(Jee-Hyong Lee)
1993년 : 한국과학기술원 전산학과 학사
1995년 : 한국과학기술원 전산학과 석사
1999년 : 한국과학기술원 전산학과 박사
2002년 ~ 현재 : 성균관대학교 정보통신공학부 조교수

관심분야 : 지능시스템, 기계학습, 온톨로지

E-mail : jhlee@ece.skku.ac.kr

박두경(Doo-kyung Park)
2005년 : 성균관대학교 컴퓨터공학과 학사
2007년 : 성균관대학교 컴퓨터공학과 석사
2007년 ~ 현재 : 삼성중공업

관심분야 : Web service, Data mining

E-mail : harderme@skku.edu