한국지능정보시스템학회논문지 제12권 제2호 2006년 6월(pp.67~81)

차세대 웹을 위한 SWRL 기반 역방향 추론엔진 SMART-B의 개발*

송용욱
연세대학교 원주캠퍼스 경영정보학과
(yusong@yonsei.ac.kr)

홍준석
경기대학교 경영정보학과
(junehong@kyonggi.ac.kr)

김우주 · 윤숙희
연세대학교 정보산업공학과
(qorwkr@nate.com)

이성규
동부정보기술주식회사 보험운영팀
(oop760@dongbu.com)

현재의 웹이 HTML을 바탕으로 인간 사용자와의 인터페이스에 초점을 맞추고 있는데 비하여, 차세대 웹은 XML 및 XML 기반 각종 표준들을 바탕으로 소프트웨어 에이전트 간의 상호작용에 초점을 맞추어 나가고 있다. 차세대 웹에서 소프트웨어 에이전트의 두뇌 역할을 수행하기 위하여 추론엔진은 차세대 웹의 표준 언어인 시맨틱 웹(Semantic Web)을 충실히 이해할 수 있어야 한다. 이를 위한 기초 작업의 일환으로 OWL(Web Ontology Language)과 RuleML(Rule Markup Language)을 조합한 SWRL(Semantic Web Rule Language)이 W3C에 제안된 바 있다. 본 연구에서는 SWRL을 규칙 표현 방법으로 사용하고, OWL을 사실 표현 방법으로 사용하는 역방향 추론엔진인 SMART-B(SeMantic web Agent Reasoning Tools -Backward chaining inference engine)를 개발하고자 하였다. 이를 위하여 SWRL 기반 역방향 추론을 위한 요구 기능을 분석하고, 기존 역방향 추론 알고리즘에 차세대 시맨틱 웹의 요구 기능을 반영한 역방향 추론 알고리즘을 설계하였다. 또한, 유비쿼터스 환경에서의 각종 플랫폼 간의 독립성과 이식성을 확보하고 기기 간의 성능 차이를 극복할 수 있도록 사실 베이스 및 규칙 베이스의 관리도구와 역방향 추론 엔진 등을Java 프로그래밍 언어를 이용하여 단위 컴포넌트의 형태로 개발하였다.

논문접수일 : 2005년 11월 게재확정일 : 2006년 6월 교신저자 : 송용욱

1. 서론

웹(Web)의 등장과 함께, 기존에 독립형(Stand-alone) 시스템으로, 또는 서버의 백 오피스(Back-office) 시스템으로 사용되던 추론엔진들이 웹 기반 시스템으로 상용화되어 고객 지원, 교육, 위험 관리 등 다양한 분야의 웹 사이트에서 성공적으로 활용되고 있다. (Blaze Advisor, Exsys, Expertise2go, ILog 등) 또한, (Kim, *et al.*, 2005), 송용욱 등(2003)은 서버형 추론엔진의 서버 부담을 줄이기 위한 방안으로 HTML 기반 추론 방법을 제시하고 의료 진단(Song *et al.*, 2003) 등의 분야에 적용한 바 있다. 그러나, 이러한 추론 엔진 또는 추론 방법들은 HTML에 의한 디스플레이를 바탕으로 추론 서비스를 제공하기 위한 것들로서, 인간 사용자에 대한 웹 서비스를 주목표로 하고 있다.

현재의 웹이 HTML을 바탕으로 인간 사용자와의 인터페이스에 초점을 맞추고 있는데 비하여,

* 본 연구는 2005년도 정보통신부 선도기술개발사업 "차세대 웹을 위한 시맨틱 서비스 에이전트 기술 개발"의 일환으로 KT 지원을 받아 수행되었음.

차세대 웹은 XML 및 XML 기반 각종 표준들을 바탕으로 소프트웨어 에이전트 간의 상호작용에 초점을 맞추어 나가고 있다. 사용자의 편의성을 증대하고 나아가 비즈니스 모델로 연결시키기 위한 자동화된 서비스의 일환으로 키워드 검색, 비교 구매 검색, 자동 협상 등 다양한 분야에서 소프트웨어 에이전트가 활용되고 있으며, 이들에 대한 양적 수요뿐만 아니라 이들의 질적 우수성에 대한 요구도 나날이 증가하고 있다. 이러한 소프트웨어 에이전트들이 웹 상의 정보를 쉽게 이해하고 컴퓨터와 사람 간의 협동 작업을 원활하게 하기 위하여 시맨틱 웹(Semantic Web)이 제안되었으며, 이와 관련하여 RDF(Resource Description Framework), RDF-Schema, OWL(Web Ontology Language) 등이 개발되었다.

RDF, RDF-Schema, OWL 등은 소프트웨어 에이전트에 의한 정보의 의미적 처리를 가능하게 함으로써 서로 다른 응용시스템 간의 정보를 효과적으로 탐색하고 정보를 자동으로 교환하며 정보의 재사용성을 증대시킬 수 있으나, 기존 정보를 바탕으로 새로운 정보를 도출하기 위한 지식의 표현과 활용 측면에서는 OWL의 Ontology 추론 외에는 이렇다 할 기능이 없는 실정이다. 이를 보완하기 위하여 W3C는 OWL과 RuleML(Rule Markup Language)을 조합한 지식표현 방법으로 SWRL (Semantic Web Rule Language)을 제안한 바 있다.

따라서, 본 연구에서는 SWRL을 규칙 표현 방법으로 사용하고, RDF를 사실 표현 방법으로 사용하는 역방향 추론엔진인 SMART-B(SeMantic web Agent Reasoning Tools - Backward chaining inference engine)를 개발함으로써 차세대 웹에서 소프트웨어 에이전트의 두뇌 역할을 수행할 수 있도록 하고자 한다. 이를 위하여 SWRL 기반 역방향 추론을 위한 요구 기능을 분석하고, 기존 역방향 추론 알고리즘에 차세대 시맨틱 웹의 요구 기능을 반영한 역방향 추론 알고리즘을 설계한다. 또한, 유비쿼터스 환경에서의 각종 플랫폼 간의 독립성과 이식성을 확보하고 기기 간의 성능 차이를 극복할 수 있도록 사실 베이스 및 규칙 베이스의 관리도구와 역방향 추론 엔진 등을 Java 프로그래밍 언어를 이용하여 단위 컴포넌트의 형태로 개발한다.

본 논문은 다음과 같이 구성되어 있다. 2절에서는 시맨틱 웹의 개념과, 추론을 위한 사실베이스 및 규칙베이스 관점에서 RDF, SWRL 등의 의미를 설명하고 관련 추론엔진들을 살펴본다. 3절에서는 시맨틱 웹 기반 역방향 추론 알고리즘을 설명하며, 4절에서는 시맨틱 웹 기반 역방향 추론 엔진인 SMART-B의 구조와 개발 상의 현안들을 설명한다. 마지막으로 5절에서는 결론과 향후 연구 방향에 대해 기술한다.

2. 관련 연구

2.1 시맨틱 웹

시맨틱 웹을 한마디로 정의하자면 "컴퓨터가 정보의 의미를 이해하고 의미를 조작할 수 있는 웹"(Berners-Lee, T., 2001)이라고 할 수 있다. 웹의 발달은 세계 곳곳에 산재된 정보를 인터넷이라는 환경을 통해 접근하는 방법을 제공하였다. 하지만 무수히 많은 정보의 홍수 속에서 사용자가 원하는 정보를 찾는 것은 거의 불가능하다. 따라서 사람이 정보를 찾는 것이 아닌 컴퓨터가 정보를 검색 할 수 있는 기술이 필요하게 되었다. 하지만 기존의 표현만을 위한 HTML 문서를 통해서는 문서간의 관계, 의미 등을 표현할 수가 없었다.

그래서 출현하게 된 새로운 패러다임이 시맨틱 웹이다. 시맨틱 웹은 메타데이터의 개념을 통해 웹 문서에 시맨틱 정보를 덧붙이고 이를 이용하여 소프트웨어 에이전트나 프로그램 등이 이러한 의미를 자동으로 추출할 수 있는 웹 환경을 가리킨다.

시맨틱 웹에서 데이터의 형식에 대한 내용을 기술하고 있는 RDF의 기본적 패턴은 XML 데이터를 Resource/Property/Value의 트리플(Triples) 형태로 구조화하는 것이라고 할 수 있다. 여기서 설명하는 트리플은 표현하고자 하는 해당 데이터의 객체를 Resource로 식별하고, 포함하는 Property와 그 Property가 가지는 Value를 묶은 하나의 개별 데이터 세트와 같은 개념이다(RDF Primer).

추론을 수행하기 위해 필요한 요소는 우선 사실 정보를 저장하고 있는 사실베이스와 규칙을 저장하고 있는 규칙베이스이다. 사실 정보 저장부분은 RDF를 이용하여 각 정보를 트리플 형식으로 표현한다. 그리고 규칙 부분은 시맨틱 웹 규칙 언어인 SWRL을 사용한다. SWRL은 조건부인 몸체(body) 부분과 결론부분인 머리(head) 부분으로 구성되며 각 부분은 Atom이라는 단위 트리플로 구성된다. 이렇게 추론은 사실베이스와 규칙베이스를 기반으로 일치하는 트리플 패턴을 찾아내는 방식으로 수행된다. 본 절에서는 사실베이스 표현을 위한 RDF와 규칙베이스 표현을 위한 SWRL을 설명한다.

2.1.1 RDF(Resource Description Framework)

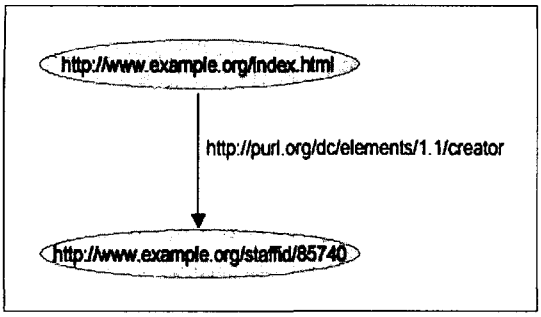
시맨틱 웹에서 메타데이터 표현을 위한 문서 표현 언어로써는 RDF, OWL 등이 있다. 그 중에 가장 기본이 되는 메타데이터를 처리하기 위한 언어로는 RDF라는 정보 리소스를 기술하는 언어가 있다. RDF는 메타데이터를 처리하기 위한 기초 언어로서, 웹에서 기계가 이해할 수 있는 정보를 교환하는 어플리케이션 간에 상호운용성을 제공한다. RDF의 목표 중 하나는 표준화되고 상호운용성 있는 방식으로, XML(Extensible Markup Language)에 기초한 데이터에 의미를 지정하도록 하는 것이다. (RDF Primer) RDF/XML은 XML 구문을 이용해 메타데이터를 표현하고, RDF 스키마는 XML 문서의 요소들 사이의 구조를 정의해 주는 XML 스키마나 DTD와는 달리 RDF 문장에서 쓰여진 어휘들의 정의를 위한 메타 언어로서의 역할을 한다. RDF 스키마에는 미리 정해진 몇 개의 어휘들이 있어서 RDF 문장에서 쓰이는 어휘 사이의 관계를 의미적으로 정의 내리는 데 사용한다(김홍기, 2002).

2.1.1.1 RDF 구문

RDF 문서의 문법은 XML에 기초한다. RDF는 객체지향적 접근의 지식표현 방식을 취하며, 세 개의 요소로 이루어진 트리플 구조를 기본으로 한다. 즉, RDF 데이터 모델에서 기술되는 URI를 갖는 모든 객체를 가리키는 Resource, Resource의 속성명을 의미하는Property, 속성값에 해당하는 Value의 구조를 가진다. RDF의 가장 기본적인 구조는 [그림 1]과 같다.

```
<?xml version="1.0">
<Class rdf:ID="Resource"
  xmlns:rdf="http://www.w3.org/1999/02-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns="uri">
  <property>value</property>
  <property>value</property>
  ...
</Class>
```

[그림 1] RDF의 기본 구조



[그림 2] RDF 데이터 모델

RDF 데이터 모델은 RDF 모델과 구문 명세서에 대한 네임스페이스를 정의하고 RDF 스키마의 정보를 갖는 네임스페이스를 정의하는 것으로 시작되며 그 이후에 URI를 갖는 자원과 속성, 속성값 등을 기술한다. 자원을 정의하는 Resource는 URI로 독자성(identity)을 지니는 데 URI 끝의 # 은 부분 식별자(fragment identifier)로서 특성의 전체 URI를 얻기 위해 로컬명으로 네임스페이스와 연결할 때 사용한다. RDF에서 리소스의 개념은 매우 중요한 요소이다. 독자성을 갖고 있는 개체이면 어떤 것이라도 리소스에 속하며 URI는 이것을 설명하기 이전에 단순히 지칭하기 위한 인식자로 사용된다. RDF 데이터 모델은 [그림 2]와 같이 트리플을 표현하기 위해 노드(Node)와 호(Arc)로 이루어진 그래프를 이용함으로써 메타데이터를 정의하기 위한 추상적이고 개념적인 틀을 제공한다.

2.1.1.2 RDF 모델의 예

RDF 표현을 위한 간단한 표현 방식의 예를 들면 <표 1>과 같다.

<표 1> RDF 표현 예제

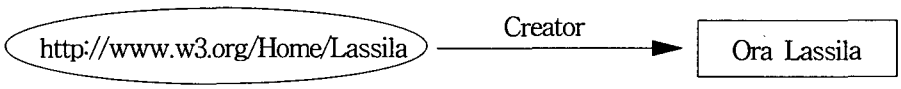
Subject(Resource)	http://www.w3.org/Home/Lassila
Predicate(Property)	Creator
Object(Value)	"Ora Lassila"

위 예제는 [그림 3]과 같이 노드(Node)와 호(Arc)로 구성된 RDF 그래프로 표현할 수 있다. 이 그림에서 노드는 Resource를 표현한 것이고, 호는 이름이 부여된 Property를 표현한다. Value의 경우에는 노드와 리터럴(Literal)을 표현한 직사각형으로 표현할 수 있다.

2.1.2 SWRL(Semantic Web Rule Language)

2.1.2.1 Rule 언어

SWRL이란 OWL의 하부 언어인 OWL DL 및 OWL Lite와 RuleML의 하부 언어인 Unary/Binary Datalog RuleML을 통합한 규칙 표현 언어이다(SWRL). SWRL로 표현하는 규칙은 기본적으로 조건(if)과 결론(then) 간의 관계를 표현하는 것을 전제로 한다. 또한, SWRL을 통하여 기존의 Horn Logic을 포함하고 OWL Axiom을 확장함으로써 유사 Horn 규칙을 OWL 기반의 지식베이스와 통합시켜 추론을 이루게 된다. SWRL을 통한 규칙 구조 표현은, 의미를 포함한 정보로부터 유추 가능한 새로운 정보들을 찾아내는 과정이라는 의미에서 볼 때 시맨틱 웹 활용 기술의 중요한 부분이 될 것이다.



[그림 3] RDF 그래프

```
<swrl:Variable rdf:ID="VARIABLE">
...

<swrl:Imp rdf:ID="RULE-NAME">
<swrl:body>
    <swrl:AtomList>
        <rdf:first>
            <swrl:ClassAtom>
                <swrl:argument1>
                <swrl:PropertyPredicate>
            </swrl:ClassAtom>
        OR
            <swrl: DatavaluedPropertyAtom>
                <swrl:argument1>
                <swrl:PropertyPredicate>
                <swrl:argument2>
            </swrl: DatavaluedPropertyAtom>
        OR
            <swrl: IndividualPropertyAtom>
                <swrl:argument1>
                <swrl:PropertyPredicate>
                <swrl:argument2>
            </swrl: IndividualPropertyAtom>
        OR
            <swrl: BuiltinAtom>
                <swrl: builtin>
                <swrl:arguments>
            </swrl: BuiltinAtom>
        </rdf:first>
        <rdf:rest>
            <swrl:AtomList>
            ...
            </swrl:AtomList>
        </rdf:rest>
    </swrl:AtomList>
</swrl:body>

<swrl:head>
    <swrl:AtomList>
    ...
    </swrl:AtomList>
</swrl:head>
</swrl:Imp>
```

[그림 4] SWRL의 기본 구조

2.1.2.2 SWRL 규칙 표현 구조 및 사용 방식

먼저 SWRL의 기본적인 구조를 살펴보자. [그림 4]에 나타난 SWRL의 기본 구조를 간략하게 살펴보면, 크게 변수 선언, 머리(head 규칙에서의 then 부분), 몸체(body 규칙에서의 if 부분)의 세 가지 부분으로 나뉘어진다. 변수 선언 부분에서는 머리나 몸체에 나타나는 모든 변수를 선언하며, 머리와 몸체에서는 And로 연결된 복수 개의 Atom들을 AtomList내에 표현한다.

Atom은 변수를 포함한 트리플, 다시 말하면, Resource/Property/Value 중 일부가 변수인 트리플을 말하며, SWRL로 규칙을 표현하고자 할 때 사용할 수 있는 Atom은 다음과 같은 7가지가 있다.

- swrl : ClassAtom
- swrl : DataRangeAtom
- swrl : DatavaluedPropertyAtom
- swrl : DifferentIndividualsAtom
- swrl : IndividualPropertyAtom
- swrl : SameIndividualAtom
- swrl : BuiltinAtom

SWRL에서 내장 함수(Built-in function)는 BuiltinAtom으로 표현되며, 현재 약 70 가지 정도가 정의되어 있다(SWRL). SWRL로 아래와 같은 규칙을 표현한 예제가 [그림 5]에 나타나 있다.

$$P(?x, ?y) \wedge P(?y, ?z) \rightarrow P(?x, ?z)$$

규칙
$P(?x, ?y) \wedge P(?y, ?z) \rightarrow P(?x, ?z)$

SWRL 표현

```
<swrl:Variable rdf:ID="z"/>
  <swrl:Variable rdf:ID="y"/>
  <swrl:Imp rdf:ID="Rule">
    <swrl:body>
      <swrl:AtomList>
        <rdf:first>
          <swrl:DatavaluedPropertyAtom>
            <swrl:argument2 rdf:resource="#y"/>
            <swrl:propertyPredicate rdf:resource="#P"/>
            <swrl:argument1>
              <swrl:Variable rdf:ID="x"/>
            </swrl:argument1>
          </swrl:DatavaluedPropertyAtom>
        </rdf:first>
        <rdf:rest>
          <swrl:AtomList>
            <rdf:first>
              <swrl:DatavaluedPropertyAtom>
                <swrl:propertyPredicate
                                rdf:resource="#P"/>
                <swrl:argument2 rdf:resource="#z"/>
                <swrl:argument1 rdf:resource="#y"/>
              </swrl:DatavaluedPropertyAtom>
            </rdf:first>
            <rdf:rest rdf:resource="http://www.w3.org/
                        1999/02/22-rdf-syntax-ns#nil"/>
          </swrl:AtomList>
        </rdf:rest>
      </swrl:AtomList>
    </swrl:body>
    <swrl:head>
      <swrl:AtomList>
        <rdf:rest rdf:resource="http://www.w3.org/
                        1999/02/22-rdf-syntax-ns#nil"/>
        <rdf:first>
          <swrl:DatavaluedPropertyAtom>
            <swrl:argument2 rdf:resource="#z"/>
            <swrl:argument1 rdf:resource="#x"/>
            <swrl:propertyPredicate rdf:resource="#P"/>
          </swrl:DatavaluedPropertyAtom>
        </rdf:first>
      </swrl:AtomList>
    </swrl:head>
  </swrl:Imp>
```

[그림 5] 규칙의 SWRL 표현 예제

2.2 추론 엔진

OWL과 SWRL을 활용한 역방향 추론엔진 개발 설명에 들어가기 앞서, OWL과 같은 시맨틱 웹을 위한 표현 언어들을 활용하는 추론엔진이나 관련 기술들을 살펴보도록 하자.

2.2.1 Jena

Jena는 시맨틱 웹 어플리케이션 구축을 위한 Java 프레임워크로서, 규칙기반 추론을 포함하여 RDF, RDFS와 OWL을 위한 프로그램 구축 환경을 제공한다.

주요 특징으로는 RDF API(Application Program Interface) 제공, RDF/XML내에서의 RDF, N3와 N-Triples를 읽고 쓰는 기능, OWL API 제공 기능 등이 있다. 또한, 주 기억장치 및 보조 기억장치용 RDQL(RDF를 위한 질의 언어)를 제공하고 있다.

2.2.2 F-OWL

F-OWL은 웹 온톨로지 언어인 OWL을 위한 온톨로지 추론 엔진이다. F-OWL에서의 온톨로지 추론 메커니즘은 Flora-2(F-logic), HiLog와 Transaction Logic등과 같은 통합언어를 XSB 추

론 엔진으로 변환해 주는 객체 지향 지식 기반 언어와 어플리케이션 플랫폼을 사용한다.

주요기능으로는 W3C의 표준 OWL 언어에 기반한 온톨로지 모델 정의를 사용하는 추론, Flora-2에서 제공하는 공리 규칙(Axiomatic Rule)을 사용한 지식 일관성 검사, Java 어플리케이션 통합을 위한 개방된 어플리케이션 프로그래밍 인터페이스 지원 기능 등이 있다.

2.2.3 Pellet

Pellet은 표현형(Expressive) Description Logic을 위해 개발된 Tableaux 알고리즘에 기초한 OWL DL 추론을 위한 시스템이다. 또한 분류 확인을 위한 완전한(Complete) 문법을 가진다. Pellet는 OWL Lite와 OWL DL을 위한 결정가능하고 유효한 추론 서비스를 제공한다. 그리고 불명확한 노드를 처리하기 위해 휴리스틱을 사용함으로 OWL Full 온톨로지를 처리하는 특징이 있다.

주요기능으로 온톨로지 분석 및 보상, 자료형 추론, Entailment, Conjunctive ABox 질의의 처리 등이 있다.

2.2.4 Euler

Euler는 증명에 기초한 로직(Logic)을 지원하는 추론 엔진이다. Euler는 Euler 경로 탐색을 사용한 역방향 추론기이며, 주어진 일련의 사실과 규칙이 주어진 결론을 지원하는가의 여부를 판단하는 시스템이다.

2.2.5 Hoolet

Hoolet은 OWL 온톨로지에 관해 추론하기 위하여 First Order Prover를 사용하는 OWL DL 추론기이다. 온톨로지는 OWL API를 사용함으로 구문분석 되고, 자동 정리 증명(Automated Theorem Proving)을 위한 Problem Library의 형태로 변형된다. 그런 다음 First-Order Classical Logic을 위한 정리 증명기(Theorem Prover)을 사용하여 일관성을 확인한다.

3. 시맨틱 웹 기반 역방향 추론

시맨틱 웹 기반 역방향 추론은 트리플로 표현된 사실(Fact)들과 트리플을 조건절 또는 결론절로 갖는 규칙들을 조합하여 새로운 사실 트리플을 도출해내는 것이다. 이때, 규칙의 조건절 또는 결론절은 변수를 가질 수 있으며 이를 감안한 매칭(Matching) 및 변수 바인딩(Variable binding)이 이루어져야 한다. 또한, SWRL은 내장 함수를 표현하며, 내장 함수의 인수(Argument)에도 변수가 있을 수 있으므로 이를 감안하여야 한다. 역방향 추론을 위한 기존의 Backtracking 알고리즘을 트리플 및 내장함수를 고려하여 변형한 Backtracking 알고리즘은 [그림 6]과 같다.

```
boolean BackwardInference(Goal goal)
{
Stack searchStack = ∅
SearchNode node
Vector newNodes
Atom atom
boolean bSuccess = FALSE
m_result = ∅
node = new CSMARTBSearchNode()
node.m_subgoal = goal
node.m_variableMap = ∅
node.m_bTraversed = FALSE
searchStack.push(node)
```

```
While ((node = searchStack.first())  NULL)
{
If (node.isTraversed() = TRUE)
{
   searchStack.pop()// discard the first node
}
Else if (node.m_subgoal = Ø ) // It is true
{
   m_result = m_result
      { node.m_variableMap.getVariableBinding(goal) }
   searchStack.pop()                // discard node
   bSuccess = TRUE
}
Else
{
   For each atom in node.m_subgoal
   {
      If (atom.getType() = TRIPLE
         OR atom.isCalculatable(node.m_variableMap))
      {
         Break
      }
   }
   If (atom = NULL)          // all is not calculatable
   {
      searchStack.pop()          // dead-end
   }
   Else IF (atom.getType() = TRIPLE)
   {
      If ((newNodes = atom.find(m_rulebase,
            m_factbase, node.m_variableMap)) = Ø )
      {
         searchStack.pop()          // dead-end
      }
      Else
      {
         searchStack.push(newNodes)
         node.setTraversed(TRUE)
      }
   }
   Else   // getTYPE() = BUILT_IN
   {
      If (atom.calculate(&curentNode.m_variableMap)
         = FALSE)
      {
         searchStack.pop()          // dead-end
      }
      Else              // change node
      {
         node.m_subgoal.remove(atom)// discard atom
      }
   }
}
}
Return bSuccess
}
```

[그림 6] 트리플용 역방향 추론 알고리즘

위 알고리즘을 설명하기 위하여, "(1 A 2), (2 A 3), (3 A 4), (4 A 5)"라는 4개의 사실과 "If (?x A ?y) and (?y A ?z) then (?x A ?z)"라는 1개의 규칙이 있는 경우 "(?p A ?q)"라는 Goal을 설정하여 추론을 진행한 Backtracking 탐색 트리의 예제를 이용하도록 하겠다. 여기서, ?x, ?y, ?z, ?p, ?q 등은 변수이다.

[그림 7]에서 이 트리의 루트 노드에는 목표(Goal) (?p A ?q)와 현재까지 알려진 사실, 즉 "?p와 ?q의 값은 모른다"라는 것이 표시되어 있다. 목표 (?p A ?q)에 대해서 먼저 4개의 사실들을 적용하면 루트 노트 아래 검게 표시된 왼쪽 4개의 노드에서처럼 하위 목표(Subgoal)는 없고, ?p, ?q가 각각 (?p=1, ?q=2), (?p=2, ?q=3), (?p=3, ?q=4),

(?p=4, ?q=5)인 4개의 결론을 얻는다.

한편, 루트 아래 오른쪽 노드에는 규칙 "If (?x A ?y) and (?y A ?z) then (?x A ?z)"에 대하여 목표 (?p A ?q)을 적용한 결과가 나타나 있다. 여기서, ?p는 ?x에 바인드(Bind)되고, ?q는 ?z에 바인드 되어 새로운 하위 목표 "(?p A ?y1) (?y1 A ?q)"이 만들어지고, ?p와 ?q의 값은 아직 알려지지 않은 상태이다. SWRL이 Horn Logic을 지원하므로 조건절의 각 Atom들은 And 관계로 연결된다. 따라서, 하위 목표의 표현 시 And를 따로 나타내지 않고, "(?p A ?y1) (?y1 A ?q)"와 같이 표시하였다. 또한 변수 ?y의 경우 노드가 확장됨에 따라 또 다른 ?y 변수들이 계속 나타나게 되므로 그것들과 구분하기 위하여 ?y1으로 표시하였다.

위 하위 목표에 4개의 사실과 1개의 규칙을 적용한 것이 아래의 5개 노드로 나타난다. 예를 들어, 사실(1 A 2)를 하위에 적용한 결과 ?p에는 1이 바인드되고 ?y1에는 2가 바인드 된다. 이에 따라 하위 목표에서 값이 확정된 (?p A ?y1), 즉 (1 A 2)가 삭제되고, (?y1 A ?q)만 (2 A ?q)로 변경되어 하위 목표로 남는다. 그리고, ?p의 값은 1이라는 것이 알려졌으므로 노드의 아랫부분에 그 사실이 표현된다. 사실을 적용한 나머지 3개의 노드들도 마찬가지로 처리된 것이 그림에 나타나 있다.

규칙 "If (?x A ?y) and (?y A ?z) then (?x A ?z)"을 첫 번째 하위 목표 "(?p A ?y1)"에 적용한 결과는 오른쪽 마지막 노드에 표시되어 있다. 이때, ?p는 ?x에 바인드되고 ?y1은 ?z에 바인드되어 "(?p A ?y2) (?y2 A ?y1)"로 변환되고, 기존 두 번째 하위 목표 (?y1 A ?q)에 덧붙여져 새로운 하위 목표가 "(?p A ?y2) (?y2 A ?y1) (?y1 A ?q)"로 된다. 이러한 과정을 반복하면 (1 A 3), (2 A 4), (3 A 5), (1 A 4), (2 A 5), (1 A 5) 등과 같은 추론 결과들을 모두 얻게 된다.

4. SMART-B의 개발

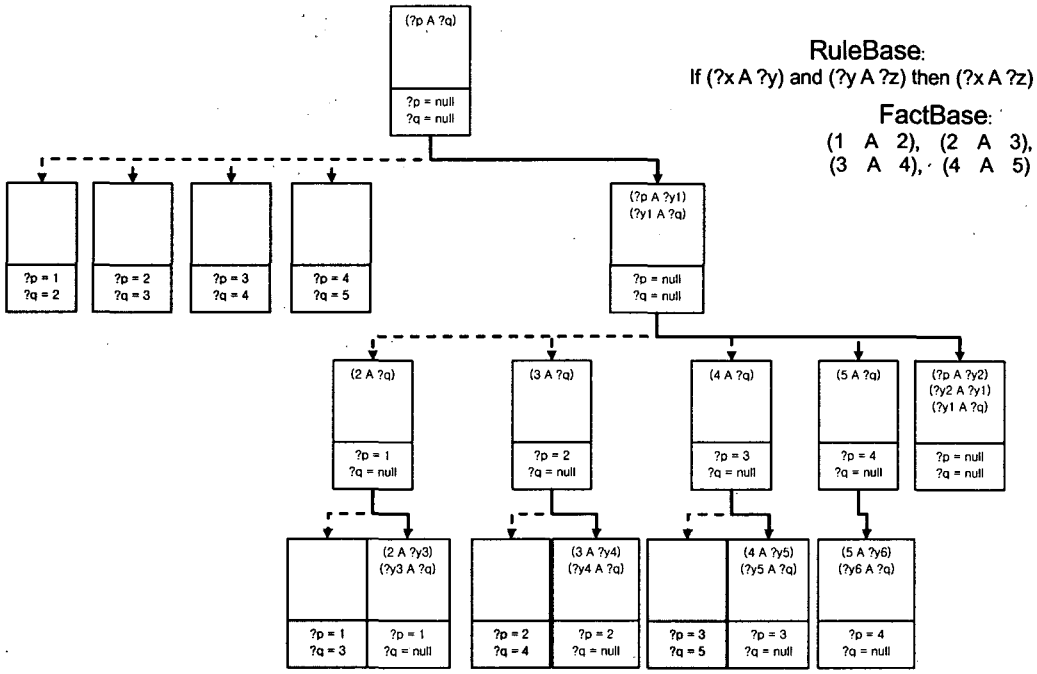
4.1 시맨틱 웹 기반 추론엔진의 요구기능 분석

본 연구에서는 차세대 웹에서 소프트웨어 에이전트의 핵심 두뇌 역할을 수행하게 될 추론엔진을 개발하기 위하여 가장 먼저 추론엔진의 요구 기능을 분석하였다. 에이전트의 하나의 컴포넌트로 사용될 추론엔진의 기능은 다른 컴포넌트와의 인터페이스를 통해 전체 에이전트의 기능을 지원하게 되는데, 이러한 기능을 입출력의 관점에서 필요한 요구 기능을 파악하였다. 첫 번째로 입력의 관점에서는 추론엔진의 모든 입력 기능이 차세대 웹의 표준을 이해할 수 있어야 한다는 점이다. 앞에서도 언급한 바와 같이 차세대 웹을 지향하고 있는 시맨틱 웹에서 제시하는 표준은 추론에 필요한 입력 정보인 사실베이스, 규칙베이스, 추론 목표를 충분히 표현할 수 있다. 따라서 추론엔진은 OWL로 표현된 사실베이스, SWRL로 표현된 규칙베이스, SPARQL로 표현된 추론 목표 명제를 이해할 수 있는 기능이 필요하다.

두 번째로 출력의 관점에서는 추론엔진이 주어진 입력 지식을 이용하여 추론 목표에 대한 결과를 명백하고(Sound) 완전하게(Complete) 도출해 내야 한다는 점이다. 이를 위하여 주어진 사실들과 규칙들을 이용하여 추론 목표에 대한 해석을 통하여 진위값(참/거짓)을 판정하고, 추론 목표에 대한 모든 변수의 제한 값을 탐색하는 기능이 필요하다. 이와 같이 분석된 추론엔진의 요구 기능은 추론엔진의 핵심 구성요소를 파악하고 추론엔진의 시스템 구조를 설계하는 기틀을 제공한다.

4.2 역방향 추론엔진의 구성요소

3절에서 제시된 시맨틱 웹 기반 역방향 추론 알



[그림 7] 역방향 추론에 의한 Backtracking 트리 예제

고리즘과 4.1 절에서 분석된 추론엔진의 요구 기능에 따라 역방향 추론엔진은 2 개의 입출력 관련 구성요소와 3개의 핵심 구성요소를 필요로 한다. 각각의 구성요소에 대한 구체적인 기능은 다음과 같다.

(1) 사실베이스와 규칙베이스의 저장 및 관리

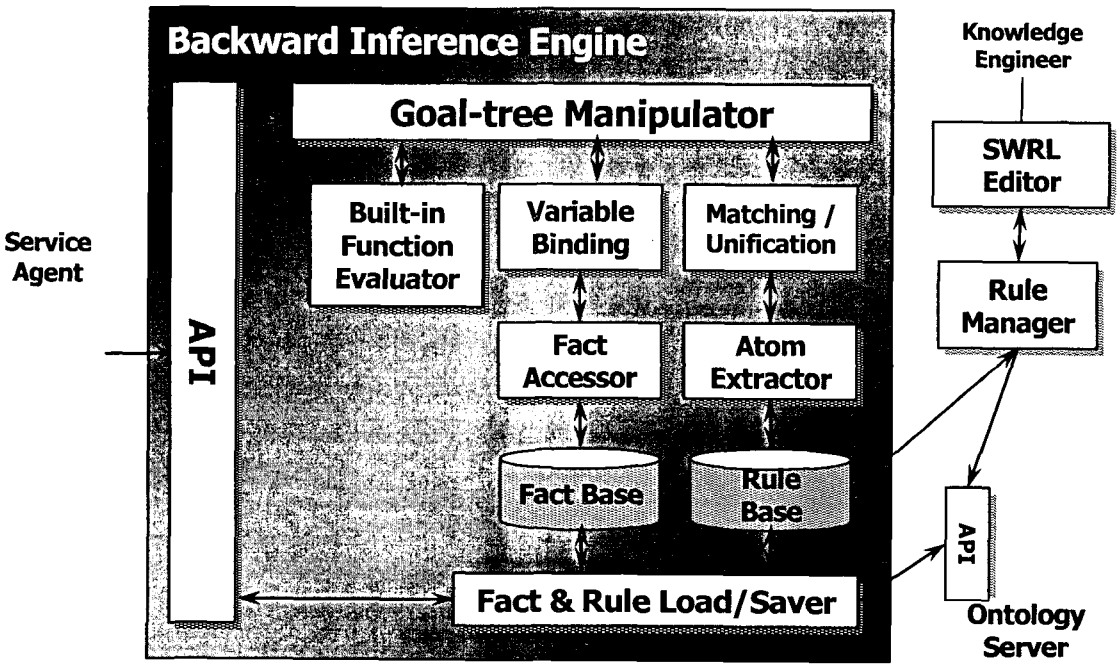
시맨틱 웹의 표준인 OWL로 표현된 사실베이스와 SWRL로 표현된 규칙베이스를 적재하는 경우에 입력된 사실베이스와 규칙베이스를 추론에 이용할 수 있도록 저장하고 관리하는 기능을 수행하는 구성요소이다. 이는 입력 관점의 요구 기능에 따라 OWL 문서와 SWRL 문서를 이해하고 그 정보 내용을 내부적인 기억구조에 따라 저장할 수 있어야 하며, 추론 알고리즘에서 필요한 시점에 요청된 지식을 제공할 수 있도록 관리하고 있어야 한다는 것을 의미한다.

(2) 추론엔진의 관리 및 결과의 반환

추론엔진이 소프트웨어 에이전트의 컴포넌트로 동작하기 위하여 다른 컴포넌트의 요청에 따라 역방향 추론 기능을 수행시키고, 추론이 수행된 결과로 추론 목표에 대한 진위값과 추론 목표에 따라 탐색된 새로운 사실들을 OWL로 표현하여 추론을 요청한 컴포넌트에게 반환하는 기능을 가진 구성요소이다. 즉, 추론엔진의 인터페이스 기능을 담당하게 된다.

(3) 목표 명제 집합의 저장 및 관리

출력 관점의 요구 기능을 충족시켜주는 핵심 구성요소로써 추론 알고리즘을 전체적으로 통제하는 기능을 수행하는 구성요소이다. 역방향 추론의 도출 원리에 따라 추론 과정에서 생성되는 목표 명제 집합에 대한 탐색 트리를 저장하는 기능



[그림 8] 시맨틱 웹 기반 역방향 추론엔진의 시스템 구조도

과 탐색 트리를 저장하는 기능과 탐색 트리상에서의 추론 경로를 추적하여 필요한 경우에 정확한 위치로 Backtracking할 수 있는 기능을 제공한다.

(4) 목표 명제의 사실로부터의 확인

추론 알고리즘에 따라 추론을 위한 도출 과정을 수행하는 두 가지 구성요소 중의 하나이다. 추론 탐색 트리 상에서 새로운 추론 노드를 추가하여 탐색 트리를 확장하기 위한 방법으로, 사실베이스 관리 구성요소를 이용하여 주어진 순서에 따라 선택된 하나의 목표 명제에 대한 진위값을 해석함으로써 새로운 추론 노드를 생성하는 기능을 수행한다.

(5) 목표 명제에 대한 규칙의 적용

앞의 구성요소와 함께 추론을 위한 도출 과정을 수행하는 구성요소이다. 이는추론 탐색 트리를 확장하는 방법을 제공하는데, 선택된 목표 명제에 대한 진위값을 해석하는 것이 아니라 규칙베이스 관리 구성요소를 이용하여 선택된 목표 명제(결론)를 대체하여 새로이 진위값을 해석할 다른 목표 명제(조건)들을 기존의 목표 명제 집합에 추가함으로써 새로운 추론 노드를 생성하게 된다. 이러한 기능은 규칙을 이용한 추론 기능의 가장 중요한 역할을 담당한다.

4.3 시스템 구조

본 연구에서는 추론 알고리즘과 요구 기능에 따라 [그림 8]과 같은 시스템 구조를 갖는 시맨틱 웹 기반 역방향 추론엔진을 설계하였다. 각각의 구성요소에 대한 기능과 역할은 다음과 같다.

(1) Fact Accessor, Atom Extractor와 Rule Loader 및 Saver를 포함하는 API

추론엔진을 기동시키고 사실베이스와 규칙베이스를 적재하는 기능을 컴포넌트 간에 호출 가능한 메소드로 제공한다. 이러한 API는 추론엔진을 위한 에이전트의 컴포넌트 간의 모든 인터페이스를 제공한다.

(2) Goal-tree Manipulator

목표 명제 집합의 저장 및 관리를 담당하는 것으로 API에 의해 수행이 시작되면 추론을 위한 탐색 트리를 저장하면서 필요한 하부 기능들을 추론 알고리즘에 따라 제어한다.

(3) Variable Binding

목표 명제의 사실로부터의 확인을 담당하는 것으로 Goal-tree Manipulator에 의해 호출되며, 목표 명제를 사실베이스와 비교하여 목표 명제에 대한 진위값을 확인함과 동시에 목표 명제에 포함된 변수의 제한 값을 탐색한다.

(4) Matching / Unification

목표 명제에 대한 규칙의 적용을 담당하는 것으로 앞의 것과 마찬가지로 Goal-tree Manipulator에 의해 호출되어 목표 명제와 규칙의 결론 명제가 일치하는 규칙을 찾는 과정에서 일치 여부에 대한 판단과 동시에 목표 명제에 포함된 변수의 대체 관계를 찾아낸다.

(5) Built-in Function Evaluator

이는 함수 관계에 의한 목표 명제에 대한 진위값을 확인하는 것으로 특수한 경우에 대한 사실로부터의 확인 기능의 일부라고 할 수 있다. SWRL로 표현된 규칙 내의 명제에 대하여 SWRL 내장 함수를 모두 처리할 수 있는 기능을 갖는다.

4.4 개발 환경 및 개발 현황

이상과 같이 설계된 차세대 웹을 위한SWRL 기반의 역방향 추론엔진 SMART-B를 JDK Ver.1.5를 기반으로 NetBeans Ver.5.0 환경에서 구현하였다. 또한 SWRL로 표현된 규칙베이스에 대한 저장 및 관리 기능은 독자적으로 개발하였으나, RDF 및 OWL로 표현된 사실베이스에 대한 저장 및 관리 기능은 HP사의 Jena Ver.2.3을 이용하였다. 현재까지 개발된 프로토타입은 시맨틱 웹 기반 역방향 추론엔진의 요구 기능을 성공적으로 충족시켜주고 있다. 그러나, 대용량 지식베이스에 대한 추론 효율 및 속도에 대한 검증이 추가로 필요한 상황이며, 이를 위하여 다양한 테스트 사례에 대한 실험을 진행 중에 있다.

5. 결론

웹 서비스(Web Services) 그리고 시맨틱 웹(Semantic Web)으로 이어지는 최근의 인터넷 기반 컴퓨팅 환경의 변화는 곧 가까운 미래에 컴퓨터간의 상호 운영성의 수준을 한 단계 높은 수준, 즉 컴퓨터간에 서로 이해하고 서로 운용할 수 있는 수준으로 비약시킬 것으로 예상된다. 그러나 이러한 변화는 우리에게 한편으로 새롭고 편리한 환경을, 다른 한편으로는 과제를 던져주고 있다. 컴퓨터간에 서로 이해할 수 있고, 또한 운용할 수 있다 하더라도 이를 바탕으로 다른 컴퓨터나 인터넷 서비스와 협업하여 특정 목적 혹은 주어진 과

제를 수행할 수 있는 방법론이 제공되지 않는 한 컴퓨터간의 상호운영성은 제한될 수 밖에 없는 것이다.

시맨틱 웹 분야에서도 이러한 과제를 인식하고 이를 독립적인 과업인 규칙 혹은 논리 프레임워크로 설정, 상호운영성의 효과를 극대화하고자 하고 있으며, 이에 대해 현재 진행 중인 표준은 RuleML을 기반으로 한 SWRL(Semantic Web Rule Language)로서 웹에서의 표준 규칙 언어이다. 본 연구에서는 이와 같이 추진 중인 SWRL의 표현력과 표준을 완벽하게 활용할 수 있는 추론 엔진 개발 연구의 한 성과로서 역방향 추론엔진인 SMART-B (SeMantic web Agent Reasoning Tools - Backward chaining inference engine)의 구조적 특성과 관련 기술, 표준 및 개발 결과를 소개하였다. 향후 SMART 프로젝트는 정방향 및 온톨로지 추론 기능을 보완 탑재한 완성된 형태의 시맨틱 웹 추론 엔진을 목표로 하고 있다. 또한, 유비쿼터스 환경에서의 각종 플랫폼 간의 독립성과 이식성을 확보하고 기기 간의 성능 차이를 극복할 수 있도록 사실 베이스 및 규칙 베이스의 관리도구 등도 Java 프로그래밍 언어를 이용하여 단위 컴포넌트의 형태로 개발하였으며, 또 개발하고 있다. 본 프로젝트의 결과물은 향후 시맨틱 웹의 상호운영성을 최대화하며, 그 응용가능성을 탐색할 수 있는 테스트베드나 기초 응용 도구로서 활용될 수 있을 것으로 기대한다.

참고문헌

[1] 김홍기, *월드와이드웹에서 시맨틱 웹으로*, 마이크로소프트웨어. 2002.

[2] 송용욱, 김우주, 홍준석, "지식분석도를 이용한 지식기반 웹 사이트 자동 생성 도구의 개발", *경영정보학연구*, 제13권, 제1호 (2003), 213-230.

[3] 송용욱, 이재규, "웹 기반 전문가시스템의 자동생성체계", *한국지능정보시스템학회논문지*, 제6권, 제1호(2000), 1-16.

[4] Berners-Lee, T., "The Semantic Web", *Scientific American*, Vol.501(2001).

[5] Blaze Advisor, http://www.blazesoft.com/

[6] Expertise2go, http://www.expertise2go.com/

[7] EXSYS, http://www.exsys.com/

[8] EXSYS, Inc., Moving an EXSYS Application to the EXSYS Web Runtime Engine (WREN).

[9] Euler, http://mathsrv.ku-eichstaett.de/MGF/homes/grothmann/euler/

[10] F-OWL, http://flora.sourceforge.net/

[11] Giarratano, Joseph and Gary Riley, *Expert Systems: Principles and Programming*, 2nd Edition, PWS Publishing Company, 1994.

[12] Hoolet, http://www.hoolet.org.uk/

[13] ILog, http://www.ilog.com/

[14] Jena, http://jena.sourceforge.net/

[15] Kim, W. J., Y. U. Song, and J. S. Hong, "Web enabled expert systems using hyperlink-based inference", *Expert Systems with Applications*, Vol.28, Issue 1(2005), 79-91.

[16] Nilsson, N. J., *Principles of Artificial Intelligence*, Springer-Verlag, 1982.

[17] Pellet, http://www.mindswap.org/2003/pellet/

[18] RDF Primer, http://www.w3.org/TR/rdf-primer/

[19] Russell, S. and P. Norvig, *Artificial Intell-*

igence - A Modern Approach, Prentice Hall, 1995.

[20] Song, Y. U., Y. M. Chae, S. H. Ho, and K. W. Cho, "Web-enabled Healthcare System for Hypertension: Hyperlink-based Inference Approach," *한국지능정보시스템학회논문지*, 9권 1호(2003), 91-107.

[21] SWRL, http://www.w3.org/Submission/ SWRL/

Abstract

Development of an SWRL-based Backward Chaining Inference Engine SMART-B for the Next Generation Web

Yong Uk Song*, June Seok Hong**, Wooju Kim***, Sung Kyu Lee****, Suk Hee Youn***

While the existing Web focuses on the interface with human users based on HTML, the next generation Web will focus on the interaction among software agents by using XML and XML-based standards and technologies. The inference engine, which will serve as brains of software agents in the next generation Web, should thoroughly understand the Semantic Web, the standard language of the next generation Web. As abasis for the service, the W3C (World Wide Web Consortium) has recommended SWRL (Semantic Web Rule Language) which had been made by compounding OWL (Web Ontology Language) and RuleML (Rule Markup Language). In this research, we develop a backward chaining inference engine SMART-B (SeMantic web Agent Reasoning Tools –Backward chaining inference engine), which uses SWRL and OWL to represent rules and facts respectively. We analyze the requirements for the SWRL-based backward chaining inference and design analgorithm for the backward chaining inference which reflects the traditional backward chaining inference algorithm and the requirements of the next generation Semantic Web. We also implement the backward chaining inference engine and the administrative tools for fact and rule bases into Java components to insure the independence and portability among different platforms under the environment of Ubiquitous Computing.

Key words : Backward Chaining Inference Engine, OWL, RDF, RuleML, Semantic Web, SWRL, XML

* Department of Management Information Systems, Yonsei University Wonju Campus
** Division of Business Administration, Kyonggi University
*** Department of Information and Industrial Engineering, Yonsei University
**** Insurance Operation Team, Dongbu Information Technology