ALKETge : 유비쿼터스 환경을 위한 지식표현 언어

조성원° 이건수 송세헌 김민구
아주대학교 정보통신전문대학원
{ajoujoa°, lks7256, lego, minkoo}@ajou.ac.kr

ALKETge :ALcun Knowledge rEpresenTation language Knowledge Representation Language for Ubiquitous Environment

Sungwon Cho°, Keonsoo Lee, Seheon Song, Minkoo Kim
Graduate School of Information and Communication, Ajou University

요 약

유비쿼터스에 대한 연구에서 지식표현 시스템은 반드시 필요하다. 최근 Description Logic을 많이 사용하고 있는데, Description Logic은 다양한 단계를 표현할 수 있다. 단계가 높아질수록 표현력이 커지는 반면 추론이 어려워진다. 유비쿼터스 환경에서는 서비스를 하기 위해 지식들의 상하위 관계나 동일 관계 등을 추론하는 능력이 필요한데, Description Logic의 단계 중에 ALCUN의 표현력이면 이에 필요한 지식을 모두 표현할 수 있다. 본 논문에서는 ALCUN의 표현력을 지니는 언어를 제안하였다.

1. 서 론

유비쿼터스 환경에서는 지식이 필요하다. 유비쿼터스 환경에서는 사람이 기계의 행동을 지시하는 것이 아니라, 기계가 사람의 요구를 판단하고 자동으로 행동하도록 하는 것이 필수적인 서비스이다. 이러한 서비스는 미리 갖추어진 지식이 없다면 불가능하다. 사용자가 무엇을 좋아하는지, 지금 무엇이 필요한지, 어떤 서비스를 해 줄 수 있는지 이미 알고 있어야 한다.

또한 지식을 갖추고 있을 뿐만 아니라, 그 상황에 맞는 행동을 할 수 있도록 추론을 해야 한다. 활용되지 못하는 지식은 쓸모가 없다. 마치 데이터베이스에서 자료를 불러오듯이, 주어진 환경에서 사용자에게 맞춘 어떤 서비스를 제공할 수 있는지 지식으로부터 추론할 수 있어야 한다.

이러한 이유로 유비쿼터스 환경을 구축하기 위해서는 지식표현과 추론을 할 수 있는 시스템이 반드시 필요하다. 그러나 유비쿼터스 환경에서는 꼭 높은 표현력이 필요한 것은 아니다. 어떠한 서비스를 하려고 할 때, 이 서비스를 하기 위해 이루어져야 하는 하위 서비스를 찾는다던지, 서비스의 속성을 찾는 정도의 능력을 필요로 한다.

제안하는 언어는 Description Logic을 기본으로 한다. Description Logic은 표현력에 따라 여러 단계로 나누어져 있기 때문에, 필요한 정도의 표현력을 가진 언어를 새로 고안할 수 있기 때문이다. 여러 단계가 있지만 ALCUN의 표현력이면 가능하다. 본 논문에서는 유비쿼터스 환경에서 필요한 표현력을 지닌, 즉 ALCUN의 표현력을 지닌 Description Logic 언어를 제안한다.

이후의 내용은 관련연구에서 지식 표현과 Description Logic에 대해 알아본다. 그리고 본 논문에서 소개할 새로운 Description Logic 언어에 대한 소개를 한 뒤 새로운 언어를 가지고 모델링을 하고 ALREX 시스템으로 추론을 해 볼 것이다. 마지막으로 결론으로 마무리한다.

2. 관련연구

2-1. 지식표현 연구

지식표현 분야는 크게 두 방향으로 진행되어 왔다. 탄탄한 수학적 기반을 둔 논리 기반의 표현 방법과 비논리 기반의 표현 방법이 있었다.

First-order logic (FOL)[1]으로 대표되는 논리 기반의 지식표현은 그 표현력이 대단하고, 범위가 특정 도메인에 국한되지 않는다. 수학에 기반을 둔만큼 표현에 있어서의 완전성이나 추론 과정에서의 명확함이 두드러진다. 그러나 논리 기반의 지식표현에는 단점도 있었다. 바로 추론 시 시간 복잡도의 문제였다. 완벽한 논리력에 비해 비결정론적(nondeterministic) 시간 복잡도를 지닌 논리 기반의 지식 표현은 실제 시스템에서 느린 응답속도를 보이는 경우가 많다.

비논리 기반의 지식표현은 시멘틱 네트웍(semantic network)과 프레임(frame) 모델로 대표되어 왔다. 이러한 방법은 지식을 Ad hoc한 자료구조(data structure)로 표현하고, 추론 과정 역시 표현된 자료구조를 다루는 Ad hoc한 과정으로 서술한다. 이러한 지식표현 방법은 시간 복잡도가 향상되어 실제 시스템에서 효율적으로 사용될 수 있는 장점이 있으나, 표현력의 한계가 있다는 단점도 지니고 있다.

2-2. Description Logic

비논리에 기반 지식표현 방법이 좋은 방법이긴 하지만, 더 나은 시스템을 만들기 위해서는 구조적인 지식에 의미(semantic)를 부여하는 과제가 남아있다. 따라서 두 방향을 접목시킨 연구들이 이어졌는데, 그 내용은 논리 기반 지식표현의 의미를 프레임의 구조에 대입하는 형식이었다. 예를 들면 First-order logic의 경우, unary predicate은 구조적인 지식의 'concept'로, binary predicate은 'relationship'으로 변환하는 식이다. 이러한 맥락에서 Description Logic에 대한 연구가 시작되었다.

Description Logic은 발전을 거듭하여 여러 단계의 표현력을 지닌 언어가 생겨났으며, 그에 따라 여러 시스템이 개발되고 있다.

Description Logic은 개념(Concept)과 관계(Role)로 의미를 표현하고, 개체(individual)를 더함으로써 지식을 표현하는 방법이다.[1] Description Logic 시스템은 TBox와 ABox로 구성되어 있다. TBox는 개념들과 그 속성, 그리고 개념들과의 관계로 지식을 표현한다. ABox는 TBox에 정의된 개념들의 개체로 지식을 표현하며, ABox의 개체들은 각 도메인에 밀접한 관련을 가지고 있다.

3. 제안된 언어

3-1. ALKETge

Description Logic은 지식을 표현할 수 있는 정도에 따라 단계가 나뉘게 된다. 가장 기본적인 AL에서부터 표현력이 늘어감에 따라 SHIQ나 그 이상까지 확장을 할 수 있다.[2] 표현능력에 따른 Description Logic 언어의 단계는 [표 1]에 나타나 있다.

표 1. Description Logic Family

DL Family		expressive power	semantics
S	AL	A	atomic concept
		⊤	universal concept
		⊥	bottom concept
		¬A	atomic negation
		C⊓D	intersection
		∀R.C	value restriction
		∃R.⊤	limited existential quantification
	C		negation of arbitrary concepts
	R+		transitive roles
U		C⊔D	union
E			full existential quantification
N		≥nR, ≤nR	number restriction
H			role hierarchy
I		R^-	inverse role
Q		≥nR.C, ≤nR.C	qualified number restriction
O		|I| = 1	nominal

어느 정도의 표현력을 필요로 하는가에 따라 Description Logic의 단계를 결정한다. 먼저 어떤 서비스를 하려고 하는지 알아야 한다. 유비쿼터스 환경에서의 추론은 개념간의 subsumption 관계나, equivalence 여부를 판단하는 정도이다. 예를 들어, 개체를 고려하지 않고도 충분히 추론이 가능하다.

가장 기본적인 기능인 AL에 [표 1]에서 C에 해당하는 'negation of arbitrary concepts', U에 해당하는 'union', N에 해당하는 'number restriction'의 기능을 더한 ALCUN의 표현력을 가진 언어를 설계하였다. 'negation of arbitrary concepts' 기능과 'union' 기능은 추론 알고리즘의 unfolding 과정을 위한 기능이다. 그리고 number restriction 기능은 개념간의 subsumption 관계를 판단하는 기준이 될 수 있다.

표 2. ALCUN의 BNF Syntax

```
<concept> ::= <composite_concept> <restriction>0+
                | <concept_name> <restriction>0+
<composite_concept> ::= <intersection_of_concepts>
                      | <union_of_concepts>
                      | <negation_of_concept>
<restriction> ::= <universal_restriction>
                | <existential_restriction>
                | <at_most_restriction>
                | <at_least_restriction>
<universal_restriction> ::=
        (all <role_name>.<concept>)
<existential_restriction> ::= (some <role_name>)
<at_most_restriction> ::=
        (at_most <digit> <role_name>)
<at_least_restriction> ::=
        (at_least <digit> <role_name>)
<negation_of_concept> ::= (not <concept>)
<intersection_of_concepts> ::=
        (and <concept> <concept>)
<union_of_concepts> ::= (or <concept> <concept>)
<concept_name> ::=
        <uppercase_letter><letter_or_digit>0+
<role_name> ::= <lowercase_letter><letter_or_digit>0+
<Maximal_concept> ::= *TOP*
<letter_or_digit> ::= <letter> | <digit>
<letter> ::= <lowercase_letter>|<uppercase_letter>
<lowercase_letter> ::= a|b|c|…|y|z
<uppercase_letter> ::= A|B|C|…|Y|Z
<digit> ::= 0|1|2|3|…|9
```

[표 2]는 ALKETge의 SYNTAX를 BNF 형태로 정의한 것이다.[5] 개념(Concept)은 이름과 속성(restriction), 혹은 다른 개념들의 결합(composite concept)으로 표현될 수 있다. 그리고 관계(Role)는 이름을 가지고 있으며 개념의 속성으로 개념과 개념과의 관계를 표현한다. 각 속성은 개념과 관계(Role)가 어떻게 연관되어 있는지를 나타낸다.

표현 방식과 함께 지식이 추론 가능한가를 판단해야 한다. satisfiability 문제 같은 경우 주어진 문장이 TBox 내의 개념들의 결합을 통해 표현될 수 있는지 아닌지가 판단의 기준이 된다. 예를 들어 TBox 내에 A라는 Concept이 (and B C), 즉 B와 C의 결합으로 정의되어 있고, D라는 기초 개념이 정의되어 있는 경우, (and B D)라는 문장은 A와 D의 결합문에 포함되므로 satisfiability 여부를 판단할 수 있다.

개념간의 상하위관계(subsumption)는 다음과 같이 판단할 수 있다. 첫 번째로, A라는 개념이 B라는 개념을

포함한 개념들의 결합으로 이루어졌을 경우, A가 B의 상위 클래스이다. 두 번째로, A와 B 두 개념을 모두 기초 개념의 결합으로 변환했을 때, 한 개념이 다른 개념의 모든 요소를 포함한다면 상 하위 관계가 성립한다. 동일 관계는 상 하위 관계가 서로 성립하는지를 추론하면 알 수 있다. 이러한 조건을 찾아내는 알고리즘을 설계함으로써 모든 추론이 가능하다. equivalence 문제와 disjoint 문제는 subsumption 문제를 사용하여 해결가능하기 때문이다.

표현 방법과 추론 능력과 함께, 지식을 생성하고 이용할 수 있는 명령어도 필요하다. [표 3]은 개념과 관계를 정의할 수 있는 명령어, 그리고 서비스를 할 수 있는 쿼리 명령어 들을 정리한 것이다. 이 명령어들을 이용해 지식을 표현하여 TBox를 생성하고 추론을 할 수 있다.

표 3.Commands for defining concept

Commands for defining concept	
DefineConcept	(defineconcept <concept_name> <concept>)
Concept subsumption	(implies <concept_name1> <concept_name2>)
Concept Disjointness	(disjoint <concept_name1> <concept_name2>)
Commands for defining role	
Define Role	(definerole <role_name>.<concept1> <concept2>)
Commands for querying	
Unsatisfiability check	(Isunsatisfiable <concept>)
Equivalence check	(Isequivalent <concept1> <concept2>)
Subsumption check	(Issubsume <concept1> <concept2>)
Disjointness check	(Isdisjoint <concept1> <concept2>)
Superconcepts retrieve	(superconcepts <concept>)
Role lists retrieve	(roles <concept>)

4. 실 험

실험에서는 간단한 가족 관계를 지식으로 표현하고, 추론 시스템에서 직접 추론이 되는 것을 증명할 것이다. 추론 시스템은 ALKETge 언어를 이용해 구현한 ALREX 시스템을 사용하였다.[6] 가족관계 지식을 TBox에 삽입하는 명령어들이 [표 4]에 나타나 있다.

이 명령어들을 통해 구축된 TBox엔 8개의 Concept과 두 개의 role이 정의된다. 이렇게 지식을 구축하고 나서 (Issubsume Woman Father)의 쿼리문으로 추론을 시도하였다. Woman은 TBox에 (and Person Female)로 정의가 되어 있고 Father는 (Man (some hasChild))로 정의가 되어 있다. 그런데 Father의 Man이 (not Female)을 가지고 있다는 것을 Man의 정의를 통해 알 수 있다. 따라서 Woman이 Father와 subsume 관계에 놓일 수 없음을 알 수 있다. 시스템에서는 ALKETge에 특화된 알고리즘을 적용하여 추론을 진행하였고, 올바른 결과를 얻었다.

표 4. Assert Commands

```
(defineconcept Person)
(defineconcept Female)
(defineconcept Woman (and Person Female))
(defineconcept Man (and Person (not Female)))
(definerole hasChild.Person Person)
(defineconcept Mother (Woman (some hasChild)))
(defineconcept Father (Man (some hasChild)))
(defineconcept Parent (or Man Woman))
(definerole hasChildasParent.Person Parent)
(defineconcept GrandMother (and Woman (some
hasChildasParent)
```

5. 결 론

유비쿼터스 환경에 지식표현 시스템은 반드시 필요하다. 지식표현에 최근 많이 사용되는 Description Logic의 다양한 단계들 중 ALCUN의 능력을 지닌 ALKETge라는 언어를 제안하였다. Description Logic의 단계가 높아질수록 표현력이 커지는 반면 추론이 어려워지는 특징에 따라 유비쿼터스 환경에서는 서비스를 하기 위한 특수 언어이다. ALKETge는 간편한 SYNTAX와 꼭 필요한 표현력만을 갖추고 있어, 특화된 알고리즘으로 빠른 추론이 가능하다. ALKETge를 바탕으로 구축된 추론 시스템인 ALREX가 그 결과를 보여준다. 유비쿼터스 환경의 특수성으로 볼 때, ALKETge 언어와 추론 방법은 실제 시스템에도 이용될 수 있으리라 기대한다.

6. 참고문헌

[1] Franz Baader, Deborah L. McGuinness, Daniele Nardi, Peter F. Patel-Schneider. The Description Logic Handbook, Canbridge University Press, 2003

[2] Volker Haarslev, Ralf Moller. Description of the RACER System and its Application, In IJCAR-01, volume 2083 of LNAI. Springer-Verlag, 2001

[3] M. Lenzerini, D. Nardi, A. Schaerf. Reasoning in Description Logics, Principle of Knowledge Representation, Studies in Logic. CSLI Publication, 1996

[4] Franz Baader, Ulrike Satter. An Overview of Tableau Algorithms for Descriptoin Logics. Kluwer Academic Publishers, 2001

[5] I. Horrocks. The FaCT system. In H. de Swart, editor, Automated Reasoning with Analytic Tableaux and Related Methods: International Conference Tableaux'98, number 1397 in Lecture Notes in Artificial Intelligence, pages 307 --312. Springer-Verlag, Berlin, May 1998.

[6] YS. Cho, KS. Lee, SH. Song, MK. Kim. ALREX: ALcun REasoner for tboX. 2005