

온톨로지 추론 개요와 연구동향

숭실대학교 박영택
한양대학교 최증민

1. 서론

온톨로지는 시맨틱웹을 구축하는 데 필수적으로 이용되고 있다. 시맨틱웹을 처음 접하는 많은 사람들은 온톨로지와 일반적인 지식 표현의 차이가 무엇이고, 온톨로지 추론과 인공지능의 일반적인 규칙 추론의 차이가 무엇인지에 대해서 많은 의문을 가지고 있다. 온톨로지가 사람이 가지고 있는 지식을 표현한다는 사실은 이제 모든 사람이 인지하고 있으며, 온톨로지의 중요한 기능 중의 하나는 사람이 보유한 지식을 사람과 컴퓨터, 컴퓨터와 컴퓨터 사이에 정보를 공유할 수 있도록 표현되었다는 점에 대해서도 익히 잘 알고 있다. 온톨로지는 사람이 가지고 있는 지식을 컴퓨터가 처리하고 나아가서는 이해할 수 있는 형식으로 표현된 지식이다. 이러한 경우에는 항상 그렇듯이 지식표현(KR, Knowledge Representation)과 추론(Reasoning)의 두 축에서 적절한 합의점을 찾는 문제에 도달하게 된다. 물론, 표현력이 아주 좋으면서 추론 성능도 우수한 방식을 우리가 찾을 수 있다면 최상이겠지만, 세상은 항상 그렇지 못하므로 적절한 합의점을 모색하게 된다. 즉, 표현력이 아주 좋지는 않지만 사람이 가지는 지식을 불편하더라도 표현할 수 있는 지식 표현 체계가 있고, 그 지식 표현 방식으로 지식을 표현하였을 때 실제 응용에서 Sound하면서 Complete한 추론[1]이 가능하다면 우리는 그러한 지식 체계에 만족하게 된다.

본 논문에서는 근래에 많은 사람들이 관심을 가지는 온톨로지 추론에 대한 개요를 설명하고자한다. 온톨로지는 사람이 가지고 있는 지식을 서술논리(Description Logic)에 기반을 두어 표현되었으므로 서술논리에 대한 이해를 필요로 한다. 본 특집에서는 서술 논리에 대해서 상세하게 설명하는 논문이 별도로 준비되어있으므로 본 논문에서는 서술논리에 대한 의미 및 개요를 2장에서 다루고, 온톨로지 추론이 가지는 특성을 3장에서 설명하며, 4장과 5장에서는 온톨로지 추론 엔진들에 대한 간략한 현황 및 활용 방안에 대한 설명을 하기로 한다.

2. 서술 논리 개요

온톨로지를 구축하는 데 OWL(Web Ontology Language)[2] 온톨로지 표현 언어를 많이 활용하는 추세이다. OWL은 여러 가지 형태가 존재하지만 대부분이 서술논리에 근거하여 표현되고 있다. 따라서 온톨로지 추론에 대해서 이해를 하기 위해서는 서술논리의 개념을 확실히 파악하는 것이 매우 중요하다. 본 장에서는 온톨로지 추론을 이해하는 데 필요한 서술 논리에 대한 개략적인 개념을 설명하고자한다. 인공지능을 연구하는 사람들은 사람이 가지고 있는 지식을 표현하는 데 논리를 활용하고자했다. 특히, FOPC(First Order Predicate Calculus)[1]를 이용하여 지식을 표현하는 연구에 관심을 가지게 되었다. FOL(First Order Logic)[1]은 사람의 지식을 표현하는 데 최상의 지식 표현 방식은 아니지만, 적절한 변환과정을 거치면 컴퓨터가 추론할 수 있는 형식으로 변환이 되기 때문에 인공지능 분야에서 오랫동안 관심을 가지고 연구를 하게 되었다[1]. 사람이 가지고 있는 지식을 FOL로 표현하게 되면 이를 추론할 수 있는 형태로 변환이 어느 정도 가능하였기 때문에, FOL에 대한 연구가 인공지능 분야에서 의미를 가질 수 있었다. 그러나 FOL은 Decidable[1]하지 못하기 때문에 컴퓨터가 추론을 하기에는 어려운 점이 많이 있어서, 연구자들은 FOL에 속하면서 Decidable한 논리를 알아내고 연구를 하게 되었는데 그 중의 하나가 서술논리(Description Logic)이다[3]. 서술 논리는 1980년대부터 연구가 진행되어 왔고 많은 발전을 이루게 된다. 그런데, 서술 논리가 사람들의 관심을 더 받게 된 이유는 W3C에서 표준으로 제안된 온톨로지 표현 언어인 OWL의 구조가 서술 논리의 구조와 일치한다는 점이다. 따라서 OWL로 표현된 온톨로지를 추론하기 위해서는 서술논리로 표현된 지식을 추론하는 기술을 이해하는 것이 필요하게 되었다. 이런 필요성에 따라서 많은 연구자들이 서술논리에 대한 이해와 추론 방식에 대해서 관심을 가지고 있다. 그림1은 FOL, 서술논리,

Horn 논리, DLP, Logic 프로그램의 상관관계를 보여 주고 있다[4]. 그림에서 보여 주고 있듯이 서술 논리는 FOL에 속하게 되고, FOL에 속하는 Horn 논리도 서술 논리와 같이 인공지능 연구자들의 오랜 관심을 끌고 있는 지식 표현 방식이다. 서술논리와 Horn 논리의 교차되는 부분을 DLP라고 하는 데, 이 부분의 연구도 최근에 많은 관심을 끌고 있다. Horn 논리를 포함하는 Logic 프로그램은 FOL과는 달리 Positive Disjunction을 표현 못하는 반면에 Negation as Failure나 Procedural Attachment와 같은 기능을 표현할 수 있는 차이점이 있다[4].

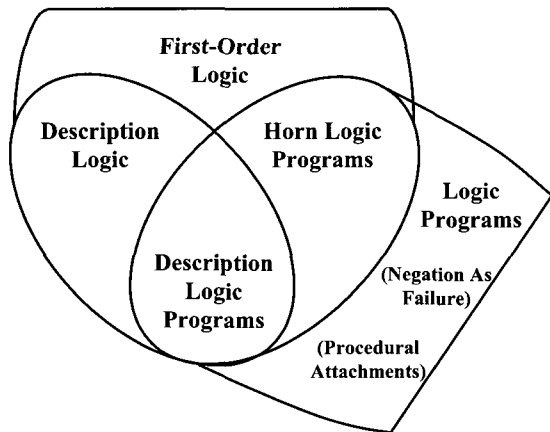


그림 1 FOL, 서술논리, DLP, Horn 논리 및 Logic 프로그램의 표현력[4]

인공지능 분야의 지식 표현 분야에서 연구가 진행되어 온 시맨틱네트워크[1]와 Frame[1]은 서술 논리 연구에 많은 영향을 주었다. 지식을 네트워크 형식으로 표현할 때 기본적인 요소는 노드와 링크이다. 서술 논리는 기본적으로 Concepts, Roles, Individuals로 사람이 가지고 있는 지식을 표현하게 된다[3]. 일반적으로 우리가 서술 논리로 사람이 가지고 있는 지식을 표현하는 경우는 다음의 그림이 보여주는 바와 같이 TBox(terminological box)와 ABox(assertional box)의 구조를 활용하여 지식을 표현한다. 서술논리라는 이름은 1980년대에 정식으로 명명되었고 그 전에는 Terminological Systems, Concept Language로 불리었는데 여기서 Terminology는 지식을 네트워크로 표현하였을 때의 구조를 의미한다.

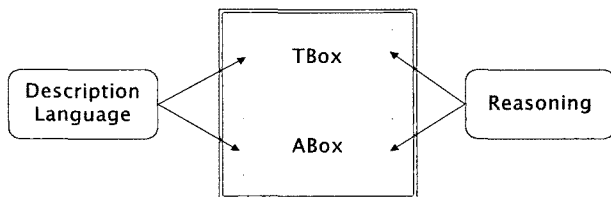


그림 2 서술 논리 표현

일반적으로 지식의 스키마 형식을 TBox에서 표현한다고도 말할 수 있다. TBox에는 지식의 Terminological 부분이 선언되고 ABox에는 지식의 Assertional 부분이 선언된다. 예를 들면 TBox와 ABox에는 다음과 같은 내용을 포함하게 된다. 다음에 있는 내용에서 알 수 있듯이 표현하고자하는 지식의 스키마에 해당하는 Terminology들이 TBox에 표현되고, ABox에는 TBox 형식에 따른 각 Individual에 대한 지식이 표현된다.

TBox:

Mother \equiv Woman \sqcap \exists hasChild.Person
 Parent \equiv Mother \sqcup Father
 Grandmother \equiv Mother \sqcap \exists hasChild.Mother

ABox:

Father(Peter)
 Grandmother(Mary)
 hasChild(Mary,Peter)
 hasChild(Mary,Paul)
 hasChild(Peter,Harry)

서술논리에는 많은 종류가 연구되고 있다[3]. 다양한 서술 논리들은 사람이 가지고 있는 지식을 표현하는 방식에 따라서 그 종류가 다르다. 가장 기본적인 서술 논리를 **ALC** 서술 논리라고 하는데, **ALC** 서술 논리는 \sqcap , \sqcup , \neg , \exists , \forall 만으로 Concept를 표현하게 된다. 서술 논리에는 Concept과 Role로 지식을 표현하는 데 Transitive Role을 **ALC** 서술 논리에 추가한 경우에 S 서술논리라고 표현한다. 이 외에도 서술 논리에서는 다음과 같은 약어를 추가적으로 사용하는 데, 일반적으로 많이 정의된 **SHIQ** 서술 논리는 ALC + R₊ + role hierarchy + inverse roles + QNR 형식의 서술 논리를 의미한다.

- H** for role inclusion axioms (role hierarchy)
- O** for nominals (singleton Classes, written {x})
- I** for inverse roles
- N** for number restrictions (of form $\leq nR$, $\geq nR$)
- Q** for qualified number restrictions (of form $\leq nR.C$, $\geq nR.C$)

3. 온톨로지 추론

현재 많이 활용되고 있는 OWL-DL[5]은 **SHIQ** 서술 논리에 기반을 두고 있는데, OWL-DL은 시맨틱스가 잘 정의되어있고 Complexity와 Decidability와 같은 특성들과 Decidable한 추론 방식에 대한 연구가 일정 수준 이상 진행되어서 매우 최적화된 방식으로 추론

엔진을 구축할 수 있는 장점을 가지고 있기 때문이다 [1]. 일반적으로 추론 엔진을 안정적으로 활용하기 위해서는 Sound하면서 Complete한 추론 알고리즘을 구현하는 것을 전제로 한다. 이와 같은 전제가 만족되지 않는다면 아무리 우리가 지식을 해당하는 언어로 표현 하더라도 활용을 할 수 없기 때문이다. 전산학에서 Turing Machine이 가지는 의미와 같이 온톨로지 표현 언어의 Decidability와 해당 언어를 추론할 수 있는 Sound, Complete한 알고리즘의 존재 여부는 매우 중요하다. 온톨로지 추론뿐만 아니라, 일반적으로 논리로 표현된 지식을 추론하는 데는 앞에서 언급한 바와 같이 Sound하면서 Complete하고 Tractable한 추론 알고리즘이 존재해야 한다. 온톨로지 추론 측면에서 본다면 Sound하다는 의미는 Valid한 Subsumption만이 추론되어지는 것을 의미하며, Complete한 추론은 모든 Valid한 Subsumption Relation들이 추론되어져야 하고, Tractable하다는 의미는 모든 Valid한 Subsumption Relation들이 Polynomial Time 내에 추론이 가능한 것을 의미한다.

앞에서 암시적으로 언급하였지만 서술 논리의 추론은 기본적으로 두 가지 기능에 해당한다. 첫 번째는 Subsumption Relation을 추론하는 것을 의미하는데 이 과정에서 Class/SubClass의 관계를 추론하게 된다. 여기서 Subsumption 추론이라는 것은 하나의 Concept이 다른 Concept을 포함하는 것을 의미하는데, 예를 들면, "윈도우 OS를 사용하는 컴퓨터를 보유한 회사"는 "XP를 장착한 PC를 보유한 은행"을 Subsume한다. 이와 같은 Subsumption 추론은 서술논리가 제공할 수 있는 가장 일반적인 추론 기능에 해당한다. 두 번째는, Instantiation 관계를 추론하는 기능으로 이 경우는 하나의 Instance가 해당하는 Concept에 포함되는지를 확인해 준다. 이와 같은 두 가지 추론이 일반적으로 온톨로지 추론의 주요 기능으로 간주되고 있다. 따라서 OWL-DL 온톨로지 추론은 사람이 가지고 있는 지식을 SHIQ 서술 논리로 표현하였을 때, Sound하면서 Complete한 온톨로지 추론 알고리즘을 적용하여 적절한 시간 내에 Subsumption과 Instantiation을 추론할 수 있는 추론 기술을 의미한다. 이러한 추론 알고리즘이 가능하기 때문에 온톨로지 구축하는 사람들은 마음 놓고 OWL-DL 수준의 온톨로지를 구축할 수 있게 된다. 만약에 SHIQ 서술 논리를 Sound하면서 Complete한 추론을 Tractable하게 수행할 수 있는 알고리즘을 고안하지 못했다면 OWL-DL 수준으로 표현한 온톨로지는 단지 지식일 뿐이고 컴퓨터공학자들이 활용할 수 없는 지식으로 남게 되었을 것이다. 따라서 온톨로지를 구축

하는 사람들은 자신이 구축하는 온톨로지가 Sound, Complete한 추론 알고리즘에 의해서 Tractable하게 활용이 가능한 형태인지를 항상 유념하면서 온톨로지 표현 언어를 선정할 필요가 있다.

일반적으로 Subsumption 관계는 $C \subset D$ 와 같은 형식으로 표현되는 데, 이를 증명하기 위해서는 $\neg C \cup D$ 가 satisfiable[1]하다는 것을 보여주면 된다. 그러므로 서술 논리에서 $C \subset D$ 를 보여 주기 위해서는 $\neg C \cup D$ 가 satisfiable하다는 것을 증명하면 되고, 이를 위해서 $C \cap D$ 가 Satisfiable하지 않다는 것을 증명하면 된다. Tableaux[6] 알고리즘은 이와 같은 satisfiability 테스트를 실험적으로 Tractable하게 보여 줄 수 있음을 증명하였고, 기존의 많은 온톨로지 추론 기술은 이 알고리즘에 기반을 두고 있다. Tableaux 알고리즘의 기본 아이디어는 증명하고자하는 내용의 Negation에 대해서 다양한 변환 규칙을 적용하여 Satisfiable하지 않다는 것을 보여주는 방식을 취하고 있다. 다음의 그림3이 보여주는 바와 같이 모든 탐색공간을 검색하여 증명하고자하는 것의 Negation이 위에서 설명한 바와 같이 Satisfiable하지 않다는 것을 보여 줌으로써, Subsumption을 추론하게 된다.

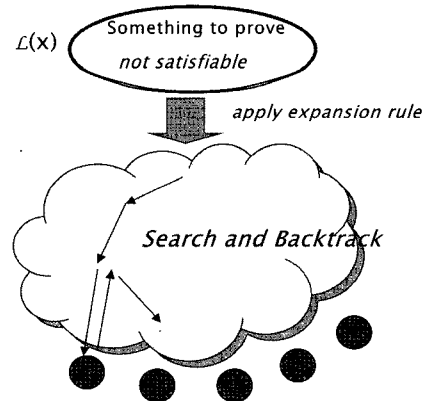


그림 3 Tableaux 알고리즘의 아이디어

앞에서 언급한 바와 같이 서술 논리는 Π , \cup , \neg , \exists , \forall 등으로 표현되기 때문에 Tableaux 알고리즘에서는 이에 대한 Expansion 규칙을 반복적으로 적용하면서 탐색공간을 확장하게 된다. 그리고 모든 탐색 공간의 Terminal 노드가 모순(Contradiction)이라는 것을 보여 주므로써 증명하고자하는 Subsumption Relation의 Negation이 Unsatisfiable이라는 것을 증명하게 된다. 다음은 Tableaux 알고리즘에서 활용하는 Expansion 규칙의 일부를 보여 준다.

$$\begin{aligned} \Pi\text{-rule} \quad & \text{if 1. } (C_1 \cap C_2) \in \mathcal{L}(x) \\ & \quad 2. \{C_1, C_2\} \notin \mathcal{L}(x) \\ & \text{then } \mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{C_1, C_2\} \end{aligned}$$

- U-rule if 1. $(C_1 \cup C_2) \in \mathcal{L}(x)$
 2. $\{C_1, C_2\} \cap \mathcal{L}(x) = \emptyset$
 then a. save **T**
 b. try $\mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{C_1\}$
 If that leads to a clash then restore **T** and
 c. try $\mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{C_2\}$
- ∃-rule if 1. $\exists R.C \in \mathcal{L}(x)$
 2. there is no y s.t. $\mathcal{L}(\langle x, y \rangle) = R$ and $C \in \mathcal{L}(y)$
 then create a new node y and edge $\langle x, y \rangle$
 with $\mathcal{L}(y) = \{C\}$ and $\mathcal{L}(\langle x, y \rangle) = R$
- ∀-rule if 1. $\forall R.C \in \mathcal{L}(x)$
 2. there is some y s.t. $\mathcal{L}(\langle x, y \rangle) = R$ and $C \notin \mathcal{L}(y)$
 then $\mathcal{L}(y) \rightarrow \mathcal{L}(y) \cup \{C\}$

앞에서 언급한 바와 같이 Tableau 알고리즘은 탐색 공간을 검색하여 Unsatisfiability를 확인하는 과정이므로, 많은 Backtracking[6]을 수행하게 되므로, 인공지능 분야에서 오랫동안 연구해온 최적화 방식이 매우 심도 있게 활용되고 있다. OWL 수준의 온톨로지 추론을 실험적으로 Tractable하게 하기 위해서는 이와 같은 최적화 방식을 얼마나 효율적으로 적용하느냐에 따라서 추론 엔진의 성능이 좌우된다. 주로 Backtracking에서 발생할 수 있는 비효율적인 문제를 보다 현명하게 처리하기 위해서 Dependency-Directed Backtracking, Semantic Branching, Partial Backjumping, Satisfiability Status Caching, Optimized Blocking과 같은 다양한 기법이 활용되고 있다[6].

4. 온톨로지 추론 엔진

온톨로지 추론 엔진을 구축하는 연구는 OWL로 표현된 온톨로지를 Sound하면서 Complete한 추론을 Tractable하게 수행할 수 있는 알고리즘 개발에 집중되고 있다. 물론, Sound나 Complete보다는 표현력이 높은 온톨로지 표현 언어에 대해서 온톨로지 추론 엔진을 연구하는 패러다임도 있지만, 이와 같은 연구는 추론 엔진의 Sound, Complete, Tractability등을 보장하지 못하게 되므로 활용성이 한정될 수밖에 없다.

온톨로지를 표현하는 기반으로 서술 논리 기반의 표현 방식에 대해서 주로 다루고 있는 데, 또 다른 연구 분야에서는 F-논리[7]기반으로 온톨로지를 표현하는 연구가 매우 활발하게 진행되고 있다. F-논리로 온톨로지를 표현하는 경우는 시맨틱웹에서 중요하게 고려하고 있는 Property의 계층적 구조에 대한 표현 및 추론에 제한이 있다는 문제점이 있지만, 이러한 제약성이 허용되는 온톨로지 구축 분야에서는 매우 성능이 우수한 온톨로지 추론 엔진[7]이 개발되고 있다. 본 논문에서는 F-논리에 대한 추론에 대해서는 지면의 한계로 기술하지 않기로 한다. 그러나 대용량 데이터를 실시간으로

활용하고자하는 기업용 응용 분야에서는 여러 분야에서 성공적으로 활용된 경험이 있으므로, 이런 분야에 온톨로지 추론을 적용하고자하는 경우는 고려해볼 만한 가치가 있다.

현재 많이 활용되고 있는 OWL기반의 온톨로지 추론 엔진은 다음과 같은 데, Ian Horrocks에 의해서 만들어진 FaCT(Fast Classification of terminology) [6]를 시작으로 앞에서 언급한 Tableau 알고리즘을 효율적으로 구현하고 있다.

온톨로지 추론 엔진	개발 학교/회사	구현 언어 및 상업용
FaCT, FaCT++	Ian Horrocks, U. of Manchester	LISP, C++기반의 open source
Pellet	U. of Maryland, MIND Lab.	Java 기반의 open source
KAON2	U. of Karlsruhe	Java 기반의 open source
CEL(Classifier for Description Logic EL+)	Dresden Univ.	LISP 기반의 open source
MSPASS	U. of Manchester	C 기반의 open source
RacerPro	Racer/Franz Inc.	LISP 기반의 상업용 추론 엔진
Cerebra Engine	Network Inference	C++ 기반의 상업용 추론 엔진

위에 언급한 온톨로지 추론 엔진은 같은 목적을 가지고 개발되고 있으며, 현재는 대용량 온톨로지 추론을 실시간으로 처리하는 문제를 극복하기 위한 다양한 연구가 활발히 진행되고 있다. 대학에서 개발되어온 FaCT, FaCT++는 매우 많이 이용되고 있으며, 최근에는 Pellet과 KAON2의 활용도 점차 증가되고 있는 추세이다. 다음은 Pellet 추론 엔진의 주요 구성 요소를 보여 준다.

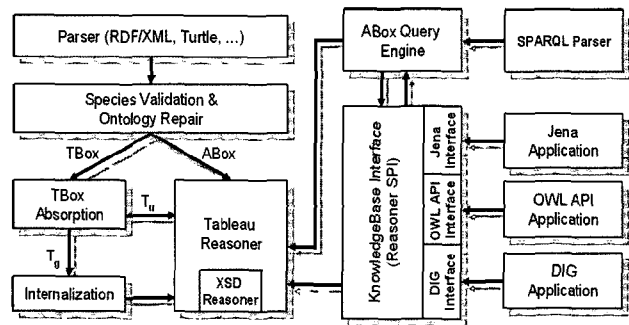


그림 4 Pellet 추론 엔진의 구성 요소

온톨로지 추론은 Open World Assumption (OWA)[7]에 기반을 두고 있다. 데이터베이스와 기타 인공지능 시스템에서는 Closed World Assumption (CWA)[1]을 많이 활용하고 있는 반면에 OWL 온톨로지 추론에서는 OWA에 기반을 두고 있다. 예를 들어, "2시와 3시에 대전으로 출발하는 KTX가 있다"는 사실이 시스템에 있다고 가정을 하고, "5시에 대전으로 출발하는 KTX가 있는가?"하는 쿼리가 들어왔을 때, CWA에서는 데이터베이스나 지식베이스에 "5시에 대전으로 출발하는 KTX가 있다"라는 사실이 존재하지 않으므로 이에 근거하여 No라는 답을 주게 된다. 그러나 OWA에서는 데이터베이스나 지식베이스에 원하는 쿼리의 답이 없다하더라도 다른 지식원에 그 답이 있을 수 있다고 가정하므로 Unknown이라는 답을 주게 된다. 즉, CWA는 주어진 데이터베이스나 지식베이스만을 추론하여 그곳에 원하는 정보가 없다면 False라고 판단을 하는 반면, OWA는 False라고 증명되지 않으면 모든 사실은 True가 될 수 있을 지도 모른다고 가정을 하게 된다. 이러한 특성 때문에 온톨로지를 구축하는 경우는 Closure Axiom을 이용하여 OWA하에서 우리가 원하는 정보를 표현하는 필요가 종종 발생한다.

온톨로지 추론은 OWL로 표현된 온톨로지의 서술논리 측면에서의 Subsumption 관계, Instantiation 관계를 추론하는 목적으로 이용된다. 따라서 일반적인 인공지능 추론의 하부에서 보다 핵심적인 역할을 수행한다고 볼 수 있다. 따라서 기존의 인공지능의 추론이 경험적 지식이나 모델 지식을 활용하여 다양한 제어 Strategy로 실행되었다면 온톨로지 추론은 사람의 지식을 온톨로지 표현하고 기존의 추론 엔진 보다 낮은 수준에서 온톨로지 추론을 담당하는 구조를 가지게 된다. 따라서 온톨로지 추론은 온톨로지 표현된 지식을 추론하는 용도뿐만 아니라, 온톨로지를 구축하는 과정에도 활용된다. 즉, 온톨로지는 한순간에 만들어 지는 것이 아니라, 여러 사람이 모여서 오랫동안 구축하기 때문에 온톨로지의 무결성 유지가 매우 중요하다. 이와 같은 온톨로지 무결성 유지를 사람이나 Visualization 도구를 이용해서 수행할 수도 있으나 일반적으로는 온톨로지 추론엔진의 도움을 받아서 수행한다.

온톨로지 추론 엔진과 온톨로지 저작 도구는 일반적으로 DIG(Description Logic Interface Group) 인터페이스로 연결된다. 가장 보편적인 온톨로지 저작 도구인 Protege[8]는 DIG 인터페이스를 지원하고 있고 대부분의 온톨로지 추론 엔진들도 DIG 인터페이스를 통해서 온톨로지 저작 도구들과 연결되어진다. 본 논문에서는 Protege[8]와 RacerPro[9] 추론 엔진을 연결

하여 온톨로지 추론을 이용하는 방식을 설명한다.

온톨로지 추론 엔진은 다음과 같은 서비스를 제공할 수 있다. 첫 번째는 온톨로지를 표현하고 있는 Class Descriptor의 Consistency를 확인해 줄 수 있다. 이와 같은 기능은 구축된 온톨로지의 무결성을 확인하는데 유효하게 이용될 수 있다. 두 번째는, Class 사이의 Subsumption 관계를 자동으로 확인할 수 있다. 예를 들면 다음은 Person, Woman, Mother Class를 표현한 온톨로지의 일부이다. 그림 5의 위 부분은 사용자가 Protege를 이용하여 온톨로지를 구축한 내용이고, 아래 부분은 RacerPro가 Subsumption 온톨로지 추론을 수행한 결과를 보여주고 있다.

Concept: Person

Parent \equiv (Person $\sqcap \exists$ hasChild.Person)

Woman \equiv (Person \sqcap Female)

Mother \equiv (Parent \sqcap Female)

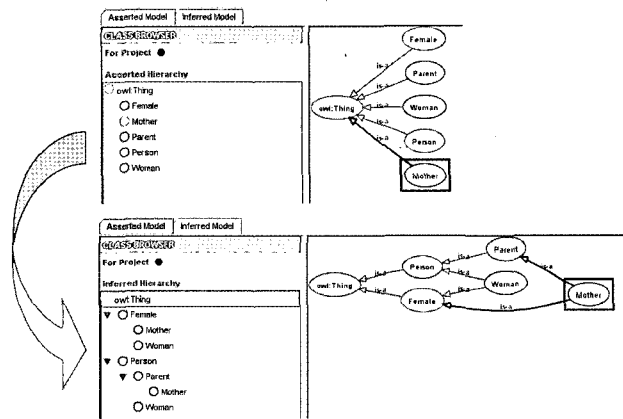


그림 5 Subsumption 관계의 추론

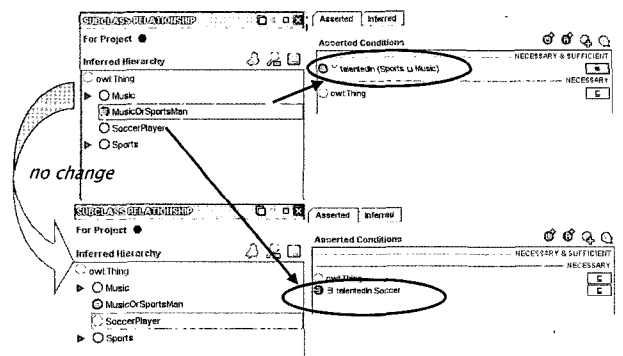


그림 6 Closure Axiom이 필요한 경우

OWL 온톨로지는 앞서도 언급한바와 같이 OWA에 기반을 두고 있다. 따라서 그림 6에 있는 바와 같이 MusicOrSportsMan과 SoccerPlayer를 정의하면 CWA에서는 SoccerPlayer가 MusicOrSportsMan에 Subsume되지만 OWA를 가정하는 OWL 온톨로지 추

론에서는 그림 6에서 보는 바와 같이 SoccerPlayer가 MusicOrSportsMan에 Subsume 안되는 것으로 추론을 하게 된다. 화면의 위 부분은 사용자가 입력한 내용이고 아래 부분은 RacerPro가 추론한 내용인데, MusicOrSportsMan과 SoccerPlayer사이의 Subsumption 관계 변화가 없는 것을 확인할 수 있다. 그 이유는 SoccerPlayer가 Soccer이외에 다른 부분에도 talent가 있다는 것을 가정할 수 있기 때문이다.

그러나 그림 7에 있는 바와 같이 SoccerPlayer를 표현하는 데 Closure Axiom을 사용하여 \forall talentedSoccer, \exists talentedSoccer로 정의하면 아래 부분에서 보여 주는 바와 같이 RacerPro는 SoccerPlayer가 MusicOrSportsMan에 Subsume되는 것으로 분류하게 된다.

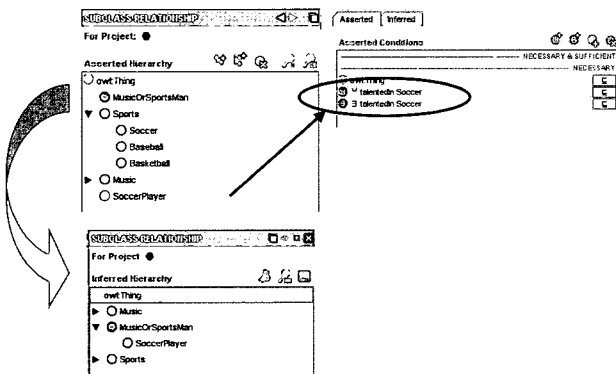


그림 7 Closure Axiom 활용

5. 국내외 연구 동향 및 향후 발전 방향

온톨로지에 대한 관심이 증대되는 추세에 따라서 OWL 기반의 온톨로지 추론 엔진 기술에 대한 연구도 매우 활발히 진행되고 있으며, 상용 엔진도 출시되어 대용량 데이터 실시간 추론에 활용되는 추세이다. 현재, 온톨로지 추론은 서술 논리 중심의 Tableaux 알고리즘을 이용한 Sound, Complete, Tractable한 온톨로지 추론 엔진 기술에 대한 연구와 아울러 F-논리 기반의 추론 기술이 활발히 연구되고 있다. 두 분야에서 각각 상용화 엔진이 출시되어 온톨로지 추론에 활용되고 있어서 대용량 온톨로지를 실용화하는 데 기여하고 있다. 이와 같은 온톨로지 추론 기술은 Subsumption 추론, Instantiation 추론의 두 가지 기능을 제공하는 역할을 담당하고 있다. 이러한 서술논리 및 F-논리를 활용하는 엔진 이외에도 보다 FOL에 가까운 방식을 이용하는 JTP(Java Theorem Prover)나 Hoolet과 같은 추론 엔진 기술도 개발되고 있는 추세이다. 또한, 기반 온톨로지는 표준화되는 경향이 있으므로 DAML-

Time과 같은 시간 추론에 적합한 온톨로지를 처리할 수 있는 Axiom을 처리하는 기능도 온톨로지 추론 엔진에 추가되는 추세이다.

W3C에는 OWL 온톨로지와 아울러 규칙(Rule)에 대한 필요성도 증대되고 있어서 온톨로지 규칙을 표현하는 표준이 활발히 진행되고 있다. RuleML[10], SWRL(Semantic Web Rule Language)[11]에 대한 관심이 고조되면서 W3C에서는 그림 8에 있는 바와 같이 초기에 나온 시맨틱웹 구조를 다소 변화하여 규칙 부분을 강조하는 새로운 시맨틱웹 구조를 제안하면서 SWRL 추론 엔진에 대한 연구도 활발히 진행되고 있다.

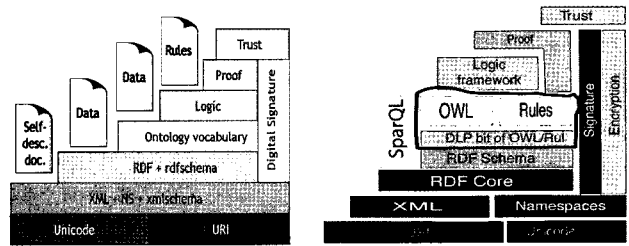


그림 8 초기의 시맨틱웹 구조와 규칙이 강조된 시맨틱웹 구조

참고문헌

- [1] Genesereth, M., Nilsson, N. Logical Foundations of Artificial Intelligence, Morgan-Kaufman, 1987.
- [2] Grigoris, A., Frank van Harmelen, A. Semantic Web Primer, The MIT Press, 2004.
- [3] Baader F., et al, The Description Logic Handbook, Cambridge, 2003.
- [4] Benjamin Groszof, Ian Horrocks, Raphael Volz, Stefan Decker, "Description Logic Programs: Combining Logic Programs with Description Logic," In Prof. of WWW2003, Budapest, Hungary, May, 2003, pp.48-57.
- [5] Michael K. Smith, Chris Welty, Deborah L. McGuinness, "OWL Web Ontology Guide," <http://www.w3.org/TR/owl-guide/>
- [6] Ian Horrocks, Optimising Tableaux Decision Procedures for Description Logics. PhD thesis, University of Manchester, 1997.
- [7] Steffen Staab, Rudi Studer (Eds.): Handbook on Ontologies: International Handbooks on Information Systems, Springer 2004.

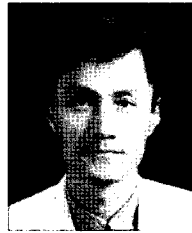
- [8] Protege Homepage,
<http://protege.stanford.edu/>
- [9] RacerPro Homepage,
<http://www.racer-systems.com/>
- [10] RuleML Homepage,
<http://www.ruleml.org/>
- [11] Ian Horrocks, Peter F. Patel-Schneider,
 Harold Boley, Said Tabet, Benjamin
 Grosz, Mike Dean, "SWRL: A Semantic
 Web Rule Language Combining OWL and
 RuleML,"
<http://www.w3.org/Submission/SWRL/>

박 영 택



1978 서울대학교 전자공학과(학사)
 1980 한국과학기술원 전산학과(석사)
 1992 University of Illinois at Urbana-
 Champaign, 전산학과(박사)
 1981~현재 숭실대학교 컴퓨터학부 교수
 관심분야: 인공지능, 시맨틱웹, 온톨로지,
 에이전트
 E-mail : park@comp.ssu.ac.kr

최 중 민



1984 서울대학교 컴퓨터공학과(학사)
 1986 서울대학교 컴퓨터공학과(석사)
 1993 State University of New York
 at Buffalo, Computer Science
 (박사)
 1993~1995 한국전자통신연구원(ETRI)
 인공지능연구실 선임연구원
 1995~현재 한양대학교 컴퓨터공학과
 부교수

관심분야: 지능형 에이전트, 시맨틱웹, 인공지능, 웹정보추출
 E-mail : jmchoi@cse.hanyang.ac.kr
