T-MERGE 연산자에 기반한 분산 토픽맵의 자동 통합

(Automatic Merging of Distributed Topic Maps based on T-MERGE Operator)

김 정 민 † 신 효 필 †† 김 형 주 †††
(Jungmin Kim) (Hyopil Shin) (Hyoungjoo Kim)

요 약 온톨로지 통합은 두 소스 온톨로지들을 통합하여 하나의 새로운 온톨로지를 생성하는 과정으로서 시맨틱 웹, 데이타 통합, 지식관리시스템 등 여러 온톨로지 응용 시스템에서 중요하게 다루는 연구주제이다. 그러나 과거의 연구들은 대부분 두 소스 온톨로지들 사이에 의미적으로 대응되는 공통 요소를 효과적으로 찾기 위한 온톨로지 매칭 기법에 집중되어 있으며 매핑 요소들을 통합하는 과정에서 발생하는 문제를 정의하고 해결하는 방법에 대해서는 간과하고 있다. 본 논문에서는 매칭 프로세스에 의해 주어진 매핑 결과에 기반하여 두 소스 온톨로지들을 통합해 나가는 상세한 통합 프로세스를 정의하고 매핑 요소들 사이에 존재하는 통합 충돌의 유형에 대한 분류 체계 및 충돌을 탐지하고 해결하기 위한 기법을 제안한다. 또한 충돌의 탐지 및 해결을 포함하여 통합 과정을 캡슐화하는 T-MERGE 연산자와 통합 과정의 기록과 오류 복구를 위한 MergeLog를 설계 및 구현한다. 제안하는 통합 모듈의 성능을 보이기 위해 동, 서양 철학 온톨로지들과 야후 및 네이버 백과사전의 일부를 온톨로지로 구현하여 실험 데이타로 활용하였으며 그 결과 전문가의 수작업에 의한 온톨로지 통합과 동일한 결과를 적은 시간과 노력으로 얻을 수 있음을 보인다.

키워드 : 온톨로지 매핑, 온톨로지 통합, 토픽맵 통합, 통합 충돌

Abstract Ontology merging describes the process of integrating two ontologies into a new ontology. How this is done best is a subject of ongoing research in the Semantic Web, Data Integration, Knowledge Management System, and other ontology-related application systems. Earlier research on ontology merging, however, has studied for developing effective ontology matching approaches but missed analyzing and solving methods of problems of merging two ontologies given correspondences between them. In this paper, we propose a specific ontology merging process and a generic operator, T-MERGE, for integrating two source ontologies into a new ontology. Also, we define a taxonomy of merging conflicts which is derived from differing representations between input ontologies and a method for detecting and resolving them. Our T-MERGE operator encapsulates the process of detection and resolution of conflicts and merging two entities based on given correspondences between them. We define a data structure, MergeLog, for logging the execution of T-MERGE operator. MergeLog is used to inform detailed results of execution of merging to users or recover errors. For our experiments, we used oriental philosophy ontologies, western philosophy ontologies, Yahoo western philosophy dictionary, and Naver philosophy dictionary as input ontologies. Our experiments show that the automatic merging module compared with manual merging by a expert has advantages in terms of time and effort.

Key words : Ontology mapping, Ontology merging, Topicmap merging, Merging conflicts

† 학생회원 : 서울대학교 컴퓨터공학부
jmkim@idb.snu.ac.kr
†† 정 회 원 : 서울대학교 언어학과 교수
hpshin@snu.ac.kr
††† 종신회원 : 서울대학교 컴퓨터공학부 교수
hjk@idb.snu.ac.kr
논문접수 : 2006년 2월 6일
심사완료 : 2006년 7월 26일

1. 서 론

온톨로지(ontology)가 가지는 특성 중의 한 가지는 개방형 및 분산형이다. 인터넷 상의 웹페이지들이 여러 사이트에 분산되어 저장되어 있듯이 온톨로지도 분산된

개방형 구조를 가진다. 즉, 시맨틱 웹에서 여행, 증권, 쇼핑 등 각 지식 도메인의 온톨로지들은 서로 다른 사이트에 존재하며 비슷한 지식 도메인에 대해 여러 개의 온톨로지들이 서로 독립적으로 생성되어 유지된다. 지식 관리 시스템에서도 인트라넷의 모든 지식을 포함하는 하나의 단일 온톨로지를 생성하는 대신 분야별 세분화된 여러 온톨로지들을 단계적으로 생성하고 이들을 응용프로그램 수준에서 연결하는 구조를 가진다[1]. 이러한 분산 환경에서 여러 소규모 온톨로지들을 모은 다음 대용량의 온톨로지를 생성하거나 의미적으로 대응되는 온톨로지들을 연결하여 통합 질의 결과를 제공하는 등의 응용이 요구된다.

이에 따라 의미적으로 대응되는 두 온톨로지를 통합하여 하나의 새로운 온톨로지를 생성하는 온톨로지 통합(ontology merging)은 온톨로지와 관련된 여러 주요 연구들 중의 하나로 간주되고 있다. 온톨로지 통합은 두 소스 온톨로지의 합집합을 구하는 것과 유사하다. 즉, 두 온톨로지의 공통 요소를 찾은 다음 중복을 제거하면서 두 소스 온톨로지의 모든 요소를 하나로 합치는 것이다. 그러나 관계형 데이타의 조인이나 합집합을 구하는 것처럼 단순한 과정이 아니다. 의미적으로 대응되는 공통 요소를 찾는 것, 공통 요소 사이의 중복을 제거하는 것, 통합 과정에서 발생하는 데이타의 충돌을 파악하고 해결하는 것 및 서로 다른 유형의 요소들을 효과적으로 통합하는 것 등 여러 문제점들을 고려해야 한다[2].

그러나 온톨로지 통합을 다루는 이전의 연구들은 대부분 두 소스 온톨로지들 사이에 의미적으로 대응되는 공통 요소를 효과적으로 찾기 위한 온톨로지 매칭(ontology matching)에 집중되어 있으며 매핑 요소들을 통합하는 과정에서 발생하는 문제를 정의하고 해결하는 방법에 대해서는 간과하고 있다[3,4]. 또한 문자열이 완전히 일치하는 용어들을 통합 후보 리스트로 제공한 다음 사용자와의 대화식에 의해 점진적으로 통합을 처리하기 때문에 대용량 보다는 비교적 소규모의 온톨로지 통합에 적합하다[5,6].

본 논문에서는 온톨로지 매칭에 의해 주어진 매핑 결과에 기반하여 두 소스 온톨로지들을 통합하기 위한 상세한 통합 프로세스를 정의하고 통합 과정에서 발생하는 매핑 요소들 사이의 통합 충돌을 정의한다. 두 온톨로지로부터 의미적으로 대응되는 요소들을 찾는 매칭 기법은 본 연구의 선행연구에서 수행되었다[7].

통합 충돌의 유형은 크게 온톨로지 요소 수준에서의 상이한 값에 의한 충돌인 엘리먼트기반 충돌(element-level conflict), 개념화 수준의 차이에서 오는 구조기반 충돌(structure-level conflict), 그리고 통합 과정에서 일시적으로 발생하는 데이타의 불일치에 의한 임시적 충돌(temporal conflict)로 나누어진다. 본 논문에서는 통합 충돌의 자세한 분류 및 각각의 의미를 정의하며 충돌의 탐지 및 해결 방법에 대해 기술한다.

또한 두 소스 온톨로지 개별 요소들을 통합하는 단위 연산들 및 이들을 캡슐화하는 T-MERGE 연산자를 설계하고 충돌 탐지 및 해결 기법을 적용한 통합 알고리즘을 정의한다. 단위 연산들의 실행 과정은 통합 후처리 및 통합 오류 복구 단계에서 사용되기 위해 MergeLog라고 명명된 하나의 로그(log)에 기록된다. 두 소스 온톨로지 통합 한 건에 대해 하나의 MergeLog가 생성된다. MergeLog에 기록되는 로그 데이타의 형식은 XML 구문을 따르고 있으며 MergeLog DTD에 따라 정당한 문서로 생성된다.

본 논문의 통합 알고리즘은 온톨로지를 표현하는 ISO 표준 데이타 모델인 토픽맵(Topic Maps)[8]을 기반으로 한다. 따라서 토픽맵으로 구현된 철학 온톨로지[9]와 야후 백과사전의 독일 문학 온톨로지를 온톨로지 통합의 실험 데이타로 사용한다. 동양 철학 토픽맵과 서양 철학 토픽맵의 통합, 서양 철학 토픽맵과 야후 철학 백과사전 토픽맵과 통합 등의 실험을 통해 제안된 통합 프로세스 및 충돌 탐지 기법, 통합 알고리즘 등이 효과적임을 보인다.

본 논문의 구성은 2장에서 온톨로지 매핑 및 통합과 관련된 이전 연구들을 살펴보고 3장에서는 토픽맵 통합을 위한 통합 문제 정의 및 통합 프로세스를 정의한다. 4장에서는 통합 충돌의 유형을 분류하고 각 유형의 충돌을 탐지 및 해결하는 방법을 설명한다. 그리고 5장에서는 두 매핑 토픽 사이의 통합을 위한 T-MERGE 연산자 및 알고리즘을 설명하고 MergeLog의 역할과 구조에 대해 설명한다. 6장에서는 실험 데이타와 실험 결과를 보이고 마지막 7장에서 논문의 결론과 향후 연구에 대해 기술한다.

2. 관련연구

온톨로지 매칭 및 통합과 관련된 이전의 연구들 중에서 온톨로지 통합을 직접적으로 다루는 연구들로 PROMPT[5]와 이를 확장한 Anchor-PROMPT[10], 그리고 Chimerae[6], FCA-Merge[11], IF-MAP[12] 등이 있다. PROMPT는 온톨로지 편집기인 Protege-2000에 추가된 온톨로지 매칭 및 통합 도구로서 사용자 대화형식의 점진적인 통합 방식을 제공하는 특징을 가진다. 먼저, 두 소스 온톨로지에서 문자열 기반 비교 기법에 따라 단순히 노드명이 완전히 일치하는 요소들 사이에 매핑을 설정한 다음 이 매핑 요소들과 이들에게 적용 가능한 연산들을 사용자에게 보여준다. 사용자가 연산의 실행을 요구하면 두 매핑 요소에 연산을 적용한 다음

이로 인해 발생하는 충돌이 있을 경우 충돌 리스트를 사용자에게 보여주는 등 대화식 방식으로 하나씩 처리해 나간다. 이러한 대화식 방식은 비교적 작은 규모의 온톨로지들을 통합하는 경우에는 적합하지만 대용량의 온톨로지들을 통합하기에는 부적합하다. Anchor-PROMPT는 PROMPT의 문자열 기반 매핑 기법 외에 매핑 노드 사이의 경로에 기반한 구조적 매핑 기법을 적용하여 매치되는 요소들을 찾도록 확장하였다.

Chimerae는 대용량의 온톨로지를 통합하고 테스트하기 위한 환경을 제공한다. 이 시스템에서 온톨로지 매칭은 독립된 프로세스가 아니라 통합 연산자의 하위 타스크로서 실행된다. 통합할 후보를 탐색하는 과정에서 용어들 사이의 유사성을 계산하고 문자열 기반 매칭 기법에 의해 산출된 유사값에 따라 통합할 대상을 결정한다. PROMPT와 유사하게 사용자에게 탐색한 통합 후보들을 보여준 다음 선택에 따라 대화식으로 통합하는 방식을 가지고 있다.

FCA-Merge는 동일한 인스턴스 집합을 공유하는 두 온톨로지들을 통합하기 위해 FCA(Formal Concept Analysis)기법을 적용한다. 그러나 FCA-Merge는 통합할 두 온톨로지가 반드시 동일한 지식 자원으로부터 생성되어야 한다는 가정을 가진다. 또한 문서 집합으로부터 인스턴스를 추출한 다음 개념 격자(concept lattice)를 생성하고 공통 용어들에 대해 점진적으로 통합 온톨로지를 생성하기 때문에 PROMPT와 같이 대용량의 온톨로지나 지식 자원 없이 온톨로지만 존재하는 경우에는 적용이 어렵다.

Pottinger[13]는 데이타베이스 스키마, UML 모델, 온톨로지 모델 등을 통합할 수 있는 범용적인 알고리즘을 제안하고 있다. 범용 통합 요구조건(generic merge requirements)을 정의하고 있으며 충돌 유형 및 해결 기법들을 설명하고 있다. 그러나 상이한 모델들 사이의 범용적인 통합을 위해 자체적으로 정의한 E-R 모델로 변환함에 따라 토픽맵의 통합에 있어서는 토픽맵 모델의 특성을 반영하지 못하는 문제점을 가진다. 예를 들어, 토픽맵의 두 토픽을 통합할 때 두 토픽의 토픽명과 속성의 유사성에 따라 통합할 수 있지만 기본적으로 두 토픽이 동일한 주제식별자(Subject Identifier)를 가질 경우에는 토픽명의 유사성과 무관하게 두 토픽을 하나의 단일 토픽으로 통합할 수 있다.

XTM 1.0 명세서[14]의 Annex F에서는 XTM 프로세서를 개발할 경우 만족해야 할 최소한의 요구조건들을 나열하고 있는데 여기에 토픽맵의 통합 조건을 기술하고 있다. 그러나 여기에서 다루는 토픽의 통합은 토픽의 주제식별자를 기준으로 하는 통합과 토픽명의 일치 여부를 기준으로 하는 통합만 정의하고 있으며 토픽의 속성들의 유사성 또는 계층구조의 유사성은 고려하지 않고 있다. 또한 연관관계들 사이의 통합도 정의되어 있지 않다. 그리고 통합 대상이 되는 두 토픽을 선정하는 매핑과정에서 두 토픽명의 일치(quality) 여부만을 고려하고 있으며 유사성(similarity) 측정에 따른 토픽 매핑은 정의되어 있지 않다.

3. 문제 정의

3.1 토픽맵 모델 정의

본 논문에서 통합 프로세스의 대상이 되는 온톨로지 모델은 토픽맵으로서 간략하게 통합의 주요 대상이 되는 모델의 구성 요소에 대해 먼저 정의한다. 일반적으로 온톨로지는 개념(Concept), 관계(Relation), 인스턴스(Instance), 법칙(Axiom) 4개의 구성요소를 가진다[15]. 정의 1에서는 이를 보다 세분화하여 토픽맵 모델을 정의하고 있다.

정의 1(토픽맵 모델). 토픽맵 TM ≔ (T_c, T_o, T_a, T_r, T_i, R_h, R_a)은 7-튜플로 정의되며 구성요소는 다음과 같다.

- T_c – 지식 도메인의 개념을 정의하는 토픽 타입들의 집합이다.
- T_o – 하나의 토픽이 가질 수 있는 속성을 정의하는 속성 타입들의 집합이다.
- T_a – 토픽들 사이의 연관성(association)을 정의하는 연관 타입들의 집합이다.
- T_r – 연관 관계를 가지는 토픽의 역할을 정의하는 역할 타입들의 집합이다.
- T_i – 실제 지식 내용을 가지는 각 개념의 인스턴스 토픽들의 집합이다.
- R_h – 개념 토픽들 사이의 상하 계층적 관계의 집합이다(superclass-subclass로 정의된 이진관계).
- R_a – 개념 토픽들 사이의 의미적 연관 관계의 집합이다(Ta의 인스턴스들의 집합).

토픽맵은 두 가지 개체인 토픽(topic)과 연관관계(association)들의 집합이다. 토픽은 구체적으로 개념화하는 대상에 따라 토픽 타입(T_c), 속성 타입(T_o), 연관관계 타입(T_a), 역할 타입(T_r) 및 인스턴스(T_i)로 구분된다. 연관관계는 토픽 타입 사이의 상하관계인 'superclass-subclass' 및 'container-containee' 관계와 의미적 연관성을 정의하는 관계로 구분된다.

3.2 토픽맵 통합 정의

토픽맵 통합의 목적은 두 토픽맵의 개체 집합에 대하여 중복을 제거한 합집합을 구하는 것이다. 정의 2에서

는 토픽맵 통합 연산의 입력 및 출력 값과 토픽맵 개체들의 합집합 연산을 정의하고 있다.

정의 2(토픽맵 통합). 토픽맵 집합 S가 주어졌을 때 집합 S에 대한 통합 연산은 다음과 같이 정의된다.

$$merge : (S \times S) \rightarrow S$$

두 토픽맵 $TM_A, TM_B \in S$의 통합은 다음과 같이 각 토픽맵의 요소들의 합집합으로 정의된다.

$$MERGE(TM_A, TM_B, M_{AB}) \rightarrow TM_C \Rightarrow \{\forall c_1 \in T_c \mid T_c \subseteq TM_A\} \cup \{\forall c_2 \in T_c \mid T_c \subseteq TM_B\} \wedge$$
$$\{\forall o_1 \in T_o \mid T_o \subseteq TM_A\} \cup \{\forall o_2 \in T_o \mid T_o \subseteq TM_B\} \wedge$$
$$\{\forall a_1 \in T_a \mid T_a \subseteq TM_A\} \cup \{\forall a_2 \in T_a \mid T_a \subseteq TM_B\} \wedge$$
$$\{\forall r_1 \in T_r \mid T_r \subseteq TM_A\} \cup \{\forall r_2 \in T_r \mid T_r \subseteq TM_B\} \wedge$$
$$\{\forall i_1 \in T_i \mid T_i \subseteq TM_A\} \cup \{\forall i_2 \in T_i \mid T_i \subseteq TM_B\} \wedge$$
$$\{\forall h_1 \in R_h \mid R_h \subseteq TM_A\} \cup \{\forall h_2 \in R_h \mid R_h \subseteq TM_B\} \wedge$$
$$\{\forall a_1 \in R_a \mid R_a \subseteq TM_A\} \cup \{\forall a_2 \in R_a \mid R_a \subseteq TM_B\}$$

통합 연산의 입력 값은 두 토픽맵 TM_A, TM_B와 그들 사이의 매핑 값인 M_{AB}이다. M_{AB}는 두 토픽맵의 요소들 중에서 의미적으로 1-대-1의 대응관계를 가지는 두 토픽의 쌍들의 집합이다. 통합 연산의 출력 값은 두 토픽맵의 요소들의 합집합으로써 두 토픽맵과 별개의 새로운 토픽맵이다.

매핑 값 M_{AB}는 (a, b, SIM_{name}, SIM_{occ}, SIM_H, SIM_{assoc}, SIM)의 7-튜플로 정의된다[7]. 여기서, a와 b는 각각 두 토픽맵의 단일 토픽이고 SIM 값들은 두 토픽의 매핑 정도를 가리키는 복합 유사값으로 [0..1] 범위의 값이다. S_{name}은 문자열 비교에 따라 산출된 값으로 두 토픽명이 어느 정도 유사성을 가지는지 가리키는 토픽명 기반 유사값, S_o는 두 토픽의 속성들 사이에 속성 타입 및 속성값이 어느 정도 유사성을 가지는지 가리키는 토픽속성 기반 유사값, S_H는 두 토픽의 계층구조에서 자식 토픽들이 어느 정도 유사성을 가지는지 가리키는 계층 구조 기반 유사값, S_a는 연관관계 타입 토픽들 사이에 두 토픽의 멤버들과 역할이 어느 정도 유사성을 가지는지 가리키는 연관관계 기반 유사값이다. 그리고 SIM은 이들 4가지 유형의 유사값들을 조합하여 평균값으로 계산한 단일 유사값이다. 따라서 a와 b 두 토픽이 의미적으로 대응되는지 여부는 단일 유사값 SIM이 특정 기준값(threshold)을 초과하는지에 의해 결정된다.

3.3 통합 프로세스 정의

토픽맵 통합 프로세스는 크게 매핑 단계와 통합 단계로 나누어진다. 매핑 단계에서는 두 토픽맵의 요소들 사이에 의미적으로 유사성이 존재하는지 파악하는 단계이고 통합 단계에서는 이전 단계에서 산출된 매핑 결과를 이용하여 두 토픽맵의 합집합을 생성하는 단계이다. 그림 1은 토픽맵 통합 프로세스의 전반적인 처리 흐름을 보이고 있다. 토픽맵들 사이의 매핑 요소를 탐색하는 기법은 선행 연구[7]에서 수행하였으며 본 논문에서는 매핑 결과를 입력으로 하여 두 토픽맵을 통합하는 과정에 대해 서술한다.

통합 방향의 결정 통합 방향을 결정한다는 것은 두 토픽맵 TM_A, TM_B에서 TM_B를 기준으로 TM_A의 토픽들을 TM_B로 통합할지 아니면 TM_A를 기준으로 TM_B의 토픽들을 TM_A로 통합할지 결정하는 것을 말한다. 통합의 방향은 두 토픽의 통합에서 단일 토픽명이나 속성 값을 결정할 때 어느 토픽을 주 토픽으로 할지 결정하기 위함이다. 또한 통합 과정에서 통합된 토픽맵을 위한 별도의 메모리 공간을 할당할 필요가 없다는 장점을 제공한다.

한 토픽맵을 트리 구조로 볼 때 매핑 경로 길이(Length of Mapping Path)는 매핑 토픽들의 최상위 수준과 최하위 수준 사이의 거리를 말한다. 그림 2에서 보면 토픽맵 TM_1의 경우 매핑 토픽들이 Philosophy, Philosopher, Kant이고 최상위 수준이 1, 최하위 수준이 3이므로

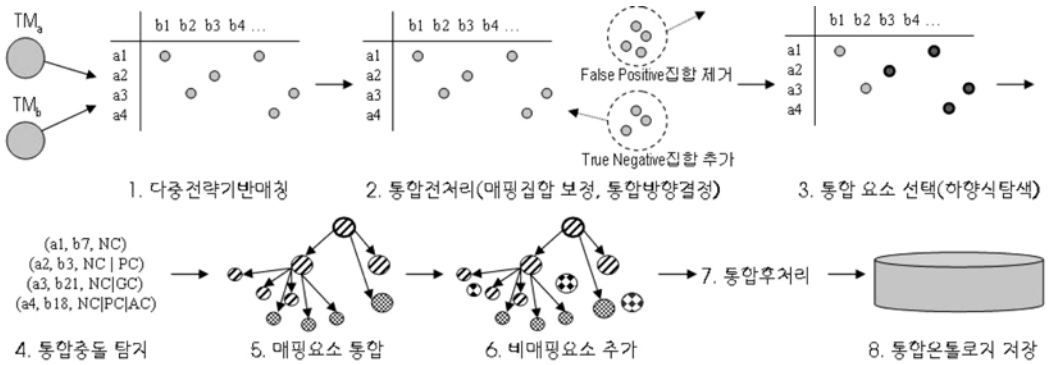

그림 1 토픽맵 통합 프로세스

정의 3(방향성). 두 토픽맵 TM_A, TM_B가 있을 때 통합 방향은 다음의 기준에 의해 결정된다. 기준은 우선순위에 따라 나열된다.

1) 매핑 경로의 길이가 짧은 토픽맵에서 긴 토픽맵으로 방향이 결정된다.
2) 매핑 경로가 같은 경우 토픽의 수가 적은 토픽맵에서 많은 토픽맵으로 결정된다.
3) 전체 토픽의 수가 같을 경우 적은 메모리의 토픽맵이 많은 메모리의 토픽맵으로 결정된다.

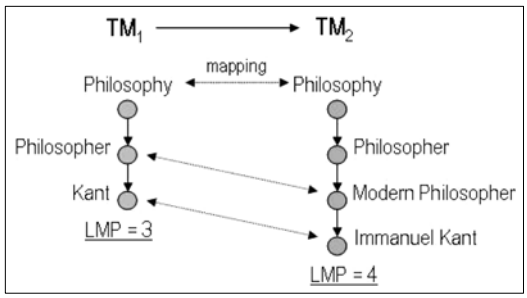

그림 2 토픽맵의 LMP와 통합 방향 결정

LMP는 3이고 TM_2의 최상위 매핑 토픽 Philosopher에서 최하위 매핑 토픽 Immanuel Kant까지 거리로 LMP는 4가 된다. 따라서 통합 방향은 TM_1에서 TM_2로 결정된다.

통합 기법 토픽맵의 요소들 사이에는 상호 의존성이 존재하기 때문에 통합에 앞서 먼저 매핑 토픽들의 통합 순서를 결정해야 한다. 예를 들어, 하나의 토픽은 여러 속성들을 가지고 있으며 하나의 속성은 하나 이상의 속성 타입을 가진다. 또한 하나의 연관관계는 연관관계 타입을 가지며 그 멤버들은 역할 타입을 가진다. 따라서 토픽을 통합하기 전에 속성 타입이나 연관관계 타입 및 역할 타입들을 먼저 통합함으로써 토픽 통합 시에 불필요한 중복이나 충돌을 회피할 수 있다. 통합 순서는 다음과 같이 결정된다.

순위 1과 2의 토픽들은 대부분 다른 토픽과의 연결이 없는 고립된 토픽이므로 매핑 토픽들만 통합함으로써 새로운 통합 토픽을 생성할 수 있다. 그러나 순위 3의 토픽들은 상하 계층관계 및 의미적 연관관계를 가지는 토픽들이므로 매핑 토픽의 통합에 따라 각 토픽과 연결된 다른 토픽들도 새로운 통합 토픽과 다시 연결되어야 한다. 이러한 연쇄적인 통합을 효율적으로 처리하기 위해 하향식(top-down) 접근 기법은 다음과 같이 과정으로 처리된다.

두 토픽맵 TM_A, TM_B와 토픽 매핑 행렬 M이 주어졌을 때,

1) 최상위 수준의 매핑 토픽쌍들을 M으로부터 구한다.
2) 최상위 수준의 매핑 토픽쌍 각각에 대해 3) ~ 7) 작업을 반복해서 처리한다.
3) 매핑 토픽 T_a, T_b를 통합하여 새로운 통합된 토픽 T_c를 생성한다.
4) T_a로의 모든 참조를 T_c로 변경한다.
5) T_b로의 모든 참조를 T_c로 변경한다.
6) 바로 다음 하위 수준의 통합되지 않은 매핑 토픽쌍들을 M으로부터 구한다.
7) 더 이상 하위 수준의 매핑 토픽쌍이 없을 때까지 3) ~ 6) 작업을 반복한다.

4. 통합 충돌 정의

4.1 통합 충돌의 유형

XTM 1.0의 Annex F에서는 토픽맵 통합의 충돌을 직접적으로 정의하지 못하고 XTM 프로세서의 조건으로서 토픽명이나 연관관계의 중복 제거만을 다루고 있다. 그러나 본 논문에서는 토픽맵을 통합할 때 발생할 수 있는 충돌의 유형은 그림 3과 같이 상세히 분류하였다. 통합 충돌은 크게 엘리먼트기반 충돌(element-level conflict), 구조기반 충돌(structure-level conflict), 임시적 충돌(temporal conflict)로 나누어진다. 엘리먼트기반 충돌은 단일 토픽 자체에서 발생하는 충돌로서 토픽명의 차이로 인한 충돌 및 속성의 차이로 인한 충돌로 세분화된다. 속성 충돌은 속성타입 충돌과 속성 값 충돌로

표 1 토픽맵 통합에 있어서 요소들의 통합 우선순위

순위	통합 대상 토픽 유형
1	1) 속성타입 토픽 : 속성을 정의하는 토픽 예 - <Desc, Description>, <Birth, Born date> 2) 연관관계 타입 토픽 : 연관관계를 정의하는 토픽 예 - <author of, written by>, <contribute to, influence to> 3) 역할타입 토픽 : 연관관계에 있어서 멤버의 역할을 정의하는 토픽 예 - <author, writer>, <philosophical texts, books>
2	단일 토픽 : 상위 토픽, 하위 토픽 및 연관관계 등 다른 토픽과의 연결이 없는 토픽
3	토픽타입 및 이와 연결된 인스턴스 토픽 : 그래프 구조로 연결된 모든 토픽들

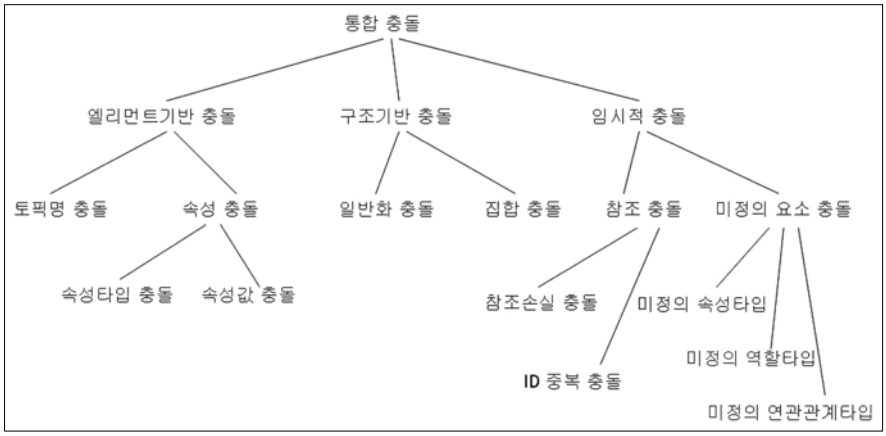

그림 3 통합 충돌 분류 계층도

나누어진다. 속성타입 충돌은 두 토픽의 속성 값은 같지만 속성타입이 다른 경우 발생하며 이와 반대로 속성 값 충돌은 속성타입은 같지만 속성 값이 다른 경우 발생한다.

구조기반 충돌은 두 매핑 토픽의 지식 표현 수준의 차이에서 비롯되는 충돌이다. 예를 들어, 두 매핑 토픽이 동일한 토픽명 Philosopher로 정의되어 있지만 토픽맵 A의 Philosopher는 그 하위에 동서양 및 시대적 구분 없이 Kant, Hegel, Mencius 등 철학자 인스턴스 토픽들로 연결되어 있는 반면 토픽맵 B의 Philosopher는 그 하위에 시대적으로 Ancient Philosopher, Medieval Philosopher, Modern Philosopher 등으로 분류한 다음 다시 각각의 시대 분류 토픽 하위에 철학자 인스턴스 토픽을 연결한 경우 두 Philosopher 토픽은 구조적 충돌을 가지는 것이다.

구조적 충돌은 일반화 충돌(generalization conflict)과 집합 충돌(aggregation conflict)로 세분화되는데 일반화 충돌은 위에서 예를 든 것과 같이 상위 토픽과 하위 토픽의 'IS-A' 관계로 연결된 분류 체계의 차이에서 발생하는 충돌이고 집합 충돌은 상위 토픽과 하위 토픽의 'PART-OF' 관계에서 하위 토픽들의 개수나 내용의 차이에서 발생하는 충돌이다. 예를 들어, 두 토픽이 Philosophy인 경우 토픽맵 A의 Philosophy는 그 하위 토픽으로 Philosopher와 Philosophical Texts만 가지는 반면 토픽맵 B의 Philosophy는 Philosopher, Texts of Philosophy, Terms of Philosophy, Doctrines of Philosophy 등 보다 세분화된 하위 토픽들을 가지는 경우 두 Philosophy 토픽 사이에는 집합 충돌이 존재한다.

임시적 충돌은 토픽맵 통합 과정에서 발생하는 충돌로서 참조 충돌(reference conflict)과 미정의 요소 충돌(undefined element conflict)로 세분화된다. 참조 충돌은 토픽들 사이의 참조 값의 부정확에 의해 발생하는 충돌로서 참조 손실 충돌은 상위 토픽이 다른 토픽맵의 매핑 토픽과 통합됨으로 인하여 하위 토픽이 가지는 상위 토픽으로의 참조가 손실되는 경우이고 ID 중복 충돌은 통합으로 인해 동일한 ID를 가지는 토픽이 다수 발생하는 경우이다. 미정의 요소 충돌은 통합 전 토픽의 속성 타입, 연관관계 타입, 역할 타입이 통합으로 인해 존재하지 않는 타입이 되는 경우이다.

4.2 충돌 탐지 및 해결

4.2.1 토픽명 충돌

토픽명 충돌은 두 매핑 토픽의 토픽명이 일치하지 않은 경우에 발생하므로 토픽맵 매핑 단계에서 산출된 토픽명 비교 연산의 결과인 SIM_{name} 유사값을 검사함으로써 토픽명 충돌 여부를 판단할 수 있다.

정의 4(토픽명 충돌 탐지). 두 매핑 토픽 t_a, t_b와 매핑 행렬 M 이 주어졌을 때,
1) $SIM_{name}(t_a, t_b) = 1$, 토픽명 일치
2) $SIM_{name}(t_a, t_b) < 1$, 토픽명 상이(충돌 발생)
 2-1) 토픽명의 포함관계 존재. t_a.Name $\subset$ t_b.Name 또는 t_a.Name $\supset$ t_b.Name(부분 충돌)
 2-2) 토픽명의 포함관계 없음(완전 충돌).

정의 5를 보면 $SIM_{name} = 1$인 경우, 두 토픽명이 일치하므로 이 경우 충돌이 없으므로 통합 토픽의 토픽명은 통합 방향에 따라 주 토픽맵의 토픽명으로 결정되고 $SIM_{name} < 1$인 경우는 두 토픽명이 상이하여 충돌이 발생한 것으로 판단한다. 토픽명 충돌이 발생할 경우 해결 방법은 두 토픽명 사이에 한 토픽명이 다른 토픽명을 부분 문자열(substring)로 포함하고 있느냐에 따라 달라진다. 포함관계가 있을 경우 통합 토픽의 토픽명은

두 토픽명 중에서 다른 토픽명을 포함하는 토픽명으로 설정된다. 포함관계가 없는 경우는 완전 충돌이 발생한 경우로서 이 경우 시스템에서 자동적으로 우선되는 토픽명을 결정할 수 없으므로 통합 토픽의 토픽명으로 두 토픽명을 모두 지정하고 통합 후처리 단계에서 전문가에 의해 수정되도록 한다.

4.2.2 속성 충돌

두 매핑 토픽 사이에 속성 충돌이 존재하는지 확인하는 기본적인 방법은 매핑 행렬에서 두 토픽의 SIM_{occ} 값을 검색하여 $SIM_{occ} < 1$인지 여부를 확인하는 것이다. SIM_{occ} 값은 두 토픽의 다중 속성들 사이에 의미적으로 얼마나 유사한지를 보여주는 척도이다. SIM_{occ}가 1인 경우 두 토픽은 같은 속성타입과 같은 속성 값을 가지는 것으로 다중 속성들이 완전히 일치함을 가리킨다.

정의 5(속성 충돌 탐지). 두 매핑 토픽 t_a, t_b와 매핑 행렬 M이 주어졌을 때,

1) $SIM_{occ}(t_a, t_b) = 1$, 속성 일치
2) $SIM_{occ}(t_a, t_b) < 1$, 속성 상이(충돌 발생)
   2-1) t_a.OccType <> t_b.OccType and t_a.OccVal = t_b.OccVal(속성타입 충돌)
   2-2) t_a.OccType = t_b.OccType and t_a.OccVal <> t_b.OccVal(속성 값 충돌)

두 매핑 토픽의 $SIM_{occ} < 1$인 경우 각각 토픽의 다중 속성들 사이에 쌍을 지어 속성타입과 속성 값을 비교하여 속성타입 충돌 또는 속성 값 충돌이 발생했는지 탐지한다. 만일 속성타입 충돌이 존재할 경우 이를 해결하기 위해 두 속성타입이 의미적으로 유사한지 여부를 검사한다. 토픽맵에서 속성타입은 독립적으로 정의된 하나의 토픽이므로 두 속성타입의 유사성 여부는 두 속성타입 토픽 사이에 매핑이 존재하는지 매핑 행렬에서 유사값을 확인해 봄으로써 알 수 있다. 만일 두 속성타입 토픽 사이에 매핑이 존재하는 경우, 표 1의 통합 우선순위에 의하여 사전에 두 속성타입 토픽을 통합한 토픽이 생성되어 있으므로 이 통합 속성 토픽을 두 매핑 토픽이 통합될 토픽의 속성타입으로 둔다. 예를 들어, 두 매핑 토픽 T_a와 T_b의 속성타입 books와 texts가 충돌 할 경우, 사전에 매핑 관계에 있는 books와 texts를 통합하여 books_and_texts를 생성해 놓았으므로 T_a와 T_b의 통합 토픽 T_c에서는 books_and_texts 속성타입을 가지며 books와 tetxts의 중복된 속성 값은 하나만 가진다.

두 매핑 토픽 사이에 속성 값 충돌이 존재하는 경우 각 토픽이 동일한 속성타입에 대해 속성 값을 서로 다른 형식으로 기술한 것이므로 시스템에서는 중복이 존재하는지 판단하기 어려우므로 통합 토픽의 속성타입에는 두 토픽의 속성 값 모두를 기술한다. 예를 들어, 두 토픽의 속성타입이 biology로 동일하지만 서로 다른 관점에서 생애해설을 한 경우 속성 값이 서로 상이한 속성 값 충돌을 가진다. 이 경우 통합 토픽에서는 biology 속성타입에 두 토픽의 생애해설 모두를 가져야 하며 어느 생애해설을 선택할지는 전문가가 후처리 단계에서 판단하여야 한다.

4.2.3 일반화 충돌

일반화 충돌은 두 매핑 토픽이 서로 다른 수준의 개념화를 가질 때 발생한다. 따라서 이 충돌을 탐지하기 위해서는 두 매핑 토픽의 하위 토픽들이 동일한 트리 레벨에서 매핑되는지를 판단해야 한다. 개념화가 단순한 토픽의 경우 복잡한 개념화 토픽에 비해 트리 깊이가 낮기 때문에 개념화의 수준이 다를 경우 서로 다른 레벨의 하위 토픽들 사이에 매핑이 존재하기 때문이다. 예를 들어, 그림 4를 보면 TM_2의 Philosopher 토픽이 TM_1의 Philosopher 토픽에 비해 더 높은 수준의 개념화를 보인다.

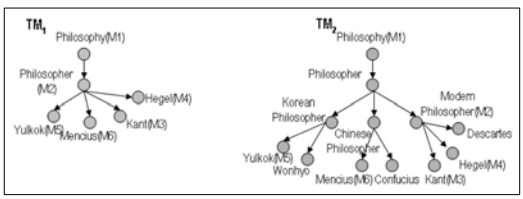

그림 4 일반화 충돌을 가지는 두 토픽맵

일반화 충돌을 탐지하고 해결하는 것은 온톨로지 통합 이전에 매핑 단계에서 선행되어야 한다. 그림 4의 두 토픽맵이 주어졌을 때 토픽명을 비교하는 문자열 매칭 기법에 따라 SIM_{name}(TM1: Philosopher, TM2: Philosopher) = 1이 되어 이들 토픽 사이에 매핑이 존재하는 것으로 결정한다. 그러나 구조적 매칭 기법에서 볼 때 TM_1의 Philosopher 토픽은 TM_2의 Modern Philosopher 토픽과 더 유사성을 가진다. 따라서 두 토픽의 4가지 유사값들을 조합한 단일 유사값으로 판단할 때 TM_1의 Philosopher 토픽과 TM_2의 Modern Philosopher 토픽 사이에 매핑이 존재하고 이 두 토픽이 통합 대상이 된다. 통합 과정에서의 일반화 충돌을 해결하는 방법은 통합 전처리 과정에서 LMP를 계산하고 통합 방향을 결정한 다음 개념화 수준이 낮은 토픽맵의 토픽들을 높은 수준의 토픽맵으로 통합함으로써 통합된 토픽맵의 개념화 수준을 높은 방향으로 맞추는 것이다.

4.2.4 집합 충돌

집합 충돌은 두 매핑 토픽이 서로 다른 범위의 하위

토픽들을 포함하는 경우에 발생한다. 예를 들어 TM_1의 Philosopher 토픽은 하위에 Korean Philosopher, Chinese Philosopher만 가진 반면 TM_2의 Philosopher 토픽은 Korean Philosopher, Chinese Philosopher, Indian Philosopher, Western Ancient Philosopher 등 더 많은 종류의 철학자 분류를 가지는 경우이다. 일반화 충돌과 마찬가지로 집합 충돌도 통합보다는 매핑 단계에서 고려된다. 구조적 매칭 기법에서 하위 토픽들 사이의 매칭 여부를 상위 토픽의 구조적 유사성에 반영하기 때문에 하위 토픽들 사이에 겹치는 부분이 많을수록 상위 토픽의 유사성이 높아진다. 통합에서는 집합 충돌을 통합 방향성을 결정하기 위한 요소로 사용된다. 즉, 두 토픽맵의 LMP가 동일할 경우 하위 토픽들의 수가 더 많은 쪽으로 통합 방향이 결정된다.

4.2.5 임시 충돌

임시 충돌은 두 토픽의 통합 후 통합된 토픽의 속성, 상위 토픽, 연관관계 등에서 참조 무결성이 일시적으로 충족되지 않는 상태를 말한다. 임시 충돌의 탐지는 통합 토픽의 속성, 상위토픽, 연관관계, 역할 타입 등이 통합 토픽맵에 정의되어 있는지 여부를 검사함으로써 가능하다.

정의 6(임시 충돌 탐지). 통합 토픽맵 TM_C와 통합 토픽 t_C가 주어졌을 때,

1) $t_c.OccType_k \notin T_o:\{OccType_i | 1 \le i \le n\}$(속성 타입 미정의 충돌)
2) $t_c.AssocType_k \notin R_a:\{AssocType_i | 1 \le i \le m\}$ (연관관계타입 미정의 충돌)
3) $t_c.RoleType_k \notin T_r:\{RoleType_i | 1 \le i \le l\}$(역할 타입 미정의 충돌)
4) $t_c.TopicType_k \notin T_c:\{TopicType_i | 1 \le i \le p\}$ (참조 충돌)

두 토픽 t_a와 t_b를 통합하여 t_c가 생성될 경우 t_c의 속성들 중에서 속성 타입이 미정의된 것이 있는지 여부는 통합 토픽맵 TM_C의 모든 속성 타입 집합 T_o에 t_c의 특정 속성 타입이 존재하는지 파악함으로써 알 수 있다. 이와 유사하게 t_c의 연관관계 타입의 미정의 여부 또한 연관관계 집합 R_a와 비교로써 알 수 있으며 t_c의 역할 타입도 역할 타입 집합 T_r과의 비교를 통해 알 수 있다. 통합으로 인하여 t_c는 t_a와 t_b의 모든 부모 토픽에 대한 참조를 가지게 되는데 이들 중에서 존재하지 않는 부모로의 참조를 토픽 타입 집합 T_c와의 비교를 통해 탐지할 수 있다.

5. 통합 연산 설계

매핑 토픽 사이의 통합은 두 토픽이 가지는 의미 정보의 합집합을 구하는 것이다. 기본적인 통합 과정은 두 토픽을 통합할 새로운 토픽을 생성한 다음 각 토픽이 가지는 의미 정보 유형에 따라 중복 값을 제거하면서 새로운 토픽으로 의미 정보를 복사한 다음 두 토픽을 제거하는 것으로 완료된다. 본 논문에서는 토픽의 의미 정보를 세분화하고 유형별 의미 정보의 합집합을 구하는 통합 과정의 알고리즘을 정의하고 하나의 통합 연산을 개별 트랜잭션으로 두고 연산의 과정 및 결과를 기록하는 MergeLog의 구조를 설계한다.

5.1 통합 알고리즘

그림 5에서 보듯이 단일 토픽은 내적 속성들과 다른 토픽들과의 외적연결로 구성된다. 토픽 내적 속성은 토픽 자체를 설명하기 위한 의미 정보로서 속성타입과 속성 값으로 이루어진다. 예를 들어, 철학자 토픽은 생애해설, 대표저작, 주요사상 등의 내적 속성을 가지도록 정의할 수 있다. 여기서 생애해설, 대표저작, 주요사상 등은 속성타입으로 철학자 토픽과 별개의 토픽으로 정의되어 있으며 철학자 토픽에서는 속성값을 기술하기 위해 해당 속성타입 토픽으로의 참조를 가진다. 토픽의 외적 연결은 계층적 관계를 표현하는 상위토픽 및 하위토픽과의 연결이 있고 의미적 연관성을 표현하기 위한 연관관계가 있다. 또한 토픽 자체의 정체성(identity)을 부여하기 위한 주제식별참조(SubjectIndicatorRef) 연결이 있다.

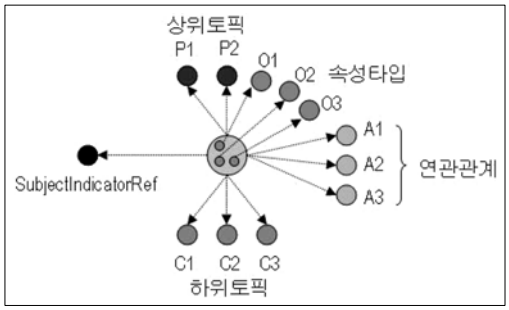

그림 5 토픽의 구조

매핑 토픽의 통합 알고리즘은 이러한 토픽 구조의 각 요소를 차례로 통합함으로써 두 매핑 토픽들을 하나로 합친다. 표 2는 매핑 토픽들의 통합을 처리하는 의사코드(pseudo code)로서 새로운 토픽을 생성한 다음 매핑 토픽들의 토픽명, 속성, 하위 토픽들과의 연결, 연관관계, 주제식별참조 연결 등을 통합해 나감으로써 하나의 통합된 토픽을 생성함을 보이고 있다.

그림 6은 두 매핑 토픽의 통합의 예를 보여준다. 여기서 Kant와 Immanuel Kant 토픽은 하위 토픽을 가지지 않는 인스턴스 토픽이다. 토픽명 Kant와 Immanuel

표 2 매핑 토픽들 사이의 통합을 처리하는 의사코드

5.1.1 토픽 통합 알고리즘

```
procedure T-MERGE(TopicA, TopicB, MAP(TopicA, TopicB))
    //① 통합 방향에 대해 검사하고 통합 로깅을 시작한다.
    direction = getDirection()
    startLogging()
    //② 토픽 ID를 제외한 모든 값이 널인 새로운 토픽을 생성한다.
    TopicC = createNewTopic()
    //③ 매핑 토픽 사이에 토픽명 충돌이 있으면 충돌을 해결하고 통합된 토픽명을 정한다.
    if ( existNameConflict(TopicA, TopicB) ) then
     setBaseName(direction, resolveNameConflict(TopicA, TopicB), TopicC)
    end if
    //④ 속성 충돌이 있는지 확인하고 충돌이 있을 경우 해결한다.
    if ( existPropertyConflict(TopicA, TopicB) ) then
     setProperties(direction, resolvePropertyConflict(TopicA, TopicB), TopicC)
    end if
    //⑤ 매핑 토픽의 모든 하위토픽들이 새로운 통합 토픽으로 참조하도록 변경한다.
    childTopics = unionChildTopics(direction, TopicA, TopicB, TopicC)
    modifyTopicRefs(direction, childTopics)
    //⑥ 매핑 토픽의 SubjectIndicatorRef들을 새로운 통합 토픽으로 모은다.
    unionSubjectIdentifiers(direction, TopicA, TopicB, TopicC)
    //⑦ 매핑 토픽의 모든 연관관계를 모으고 연관관계 타입 토픽과의 참조를 변경한다.
    assocNodes = unionAssociations(direction, TopicA, TopicB)
    modifyAssociationTypeRefs(direction, assocNodes, TopicC)
    //⑧ 임시적 충돌이 있는지 검사하고 충돌을 해결한다. 현재 매핑 토픽의 통합 트랜잭션 // 로깅을
    완료한다.
    if ( existTemporalConflict() ) then
     resolveTemporalConflict()
    end if
    if ( validateConsistency() == ACCEPTABLE ) then
     endLogging()
    else
     aborting()
    end if
end procedure
```

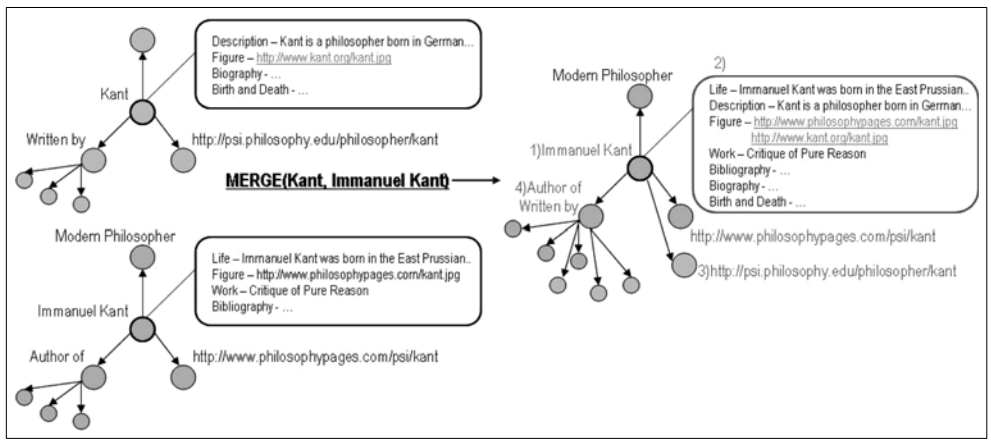

그림 6 두 매핑 토픽의 통합 예시

Kant 사이에 토픽명 충돌이 있으나 두 토픽명 사이에 포함관계({Kant} ⊂ {Immanuel, Kant})가 존재하므로 토픽명 해결 기법에 따라 통합된 토픽명은 Immanuel Kant로 정의된다.

다음으로 Kant와 Immanuel Kant의 속성을 통합하기 위해 두 토픽의 속성들의 합집합을 구한다. 이때 속성타입이 중복되는 경우 통합된 토픽에는 중복을 제거하고 하나의 속성타입을 정의한다. 예를 들어, Kant의 Figure

속성과 Immanuel Kant의 Figure 속성이 중복되므로 통합 토픽에는 Figure 속성 하나에 두 토픽의 속성 값을 모은 다중 속성 값을 가지도록 정의한다. Kant와 Immanuel Kant의 주체식별참조 통합에서는 두 값이 일치하지 않기 때문에 통합 토픽에서는 두 개의 주제식별참조를 가지게 된다. Author of와 Written by 연관관계의 경우 두 연관관계 사이에 매핑이 존재하고 Kant와 Immanuel Kant의 통합 전에 이미 통합되어 있으므로 Kant와 Immanuel Kant의 통합 토픽에서는 통합된 연관관계 타입과 연결을 가지도록 정의된다.

통합 연산 T-MERGE는 내부적으로 여러 단위 연산들로 구성된다. 토픽 통합을 위한 단위 연산들의 실행 과정은 MergeLog로 명명된 하나의 로그에 기록한다. 즉, 통합 연산 하나마다 별도의 트랜잭션으로 만들고 통합의 단위 연산들을 트랜잭션을 구성하는 하위 연산들로 로그에 기록한다. MergeLog는 관계형 데이타베이스의 트랜잭션 로그와 유사하게 T-MERGE 연산의 취소(rollback)에 사용된다. 예를 들어, I/O 오류나 연산 오류 등으로 인해 통합 프로세스가 멈추었을 경우 사용자는 MergeLog의 내용을 검토하여 취소할 구간을 정해주면 해당 구간내의 단위 연산들을 되돌릴 수 있다. 이와 함께 MergeLog는 통합 후 전문가의 후처리 작업을 지원하는 데에도 사용된다. 통합된 토픽맵에는 통합된 결과만 존재하며 어떤 소스 토픽들이 어떠한 연산의 결과로 통합되었는지에 대한 정보는 존재하지 않는다. 따라서 후작업을 하고자 하는 전문가에게 단위 연산 수준에서의 자세한 통합 과정을 보여주기 위해 MergeLog가 사용될 수 있다.

5.2 MergeLog의 구조

MergeLog는 XML 형식으로 로그 데이타를 기록된다. 최상위 엘리먼트인 MergeLog 하위에 하나 이상의 Merge 엘리먼트를 가지고 Merge 엘리먼트는 그 하위에 하나 이상의 Transaction 엘리먼트를 가진다. Transaction 엘리먼트는 하나 이상의 Operation 엘리먼트를 가지며 Operation 엘리먼트는 하나 이상의 Parameter 엘리먼트와 Description 엘리먼트, Exception 엘리먼트를 가진다. Parameter 엘리먼트는 단위 연산에 필요한 매개변수를 정의하는 엘리먼트로 매개변수의 순번을 가리키는 id, 매개변수의 타입을 가리키는 type, 매개변수의 입출력을 가리키는 inout 속성들을 가진다. 여기서 매개변수의 타입은 토픽 ID(topic), 참조(uri), 문자열(string)의 세 가지 값을 가질 수 있다. 매개변수의 inout 속성은 단위 연산에 입력으로 주어지는 매개변수일 경우 in, 단위 연산의 결과를 받는 매개변수인 경우는 out을 가진다. Description과 Exception 엘리먼트는 선택적 엘리먼트로서 단위 연산에 대한 부가 설명이나 예외 사항에 대해 기술한다.

6. 구현 및 실험

6.1 시스템 구조

본 연구의 선행 연구에서는 토픽맵 기반의 온톨로지 관리 시스템 K-Box를 구현하였다[16]. K-Box 시스템은 토픽맵을 생성하고 변경, 저장하며 키워드 검색 및 주제 검색 등을 지원하는 여러 컴포넌트들로 구성된다. 그림 8은 K-Box에 구현된 토픽맵 통합을 지원하는 주요 모듈들과 이들의 관계를 개념적으로 보이고 있다. TMMatcher와 IndexManager는 토픽맵 매핑 관리자의 하위 모듈들로서 각각 매핑 토픽 산출과 색인 관리 기

```
<MergeLog>
    <Merge sourceTM1="philosopher.xtm" sourceTM2="mod_philosopher.xtm"
                    mergedTM="philosopher&mod_philosopher.xtm"
                    direction="mod_philosopher.xtm">
        <Transaction id="T1" sourceTopic1="philosopher"
                    sourceTopic2="mod_philosopher" timestamp="2005-09-21 13:23:46">
            <Operation id="O1" Operator="CREATE_TOPIC"
                                timestamp="2005-09-21 13:23:46">
                <Parameter id="P1" type="topic" inout="out">modern_philosopher
                </Parameter>
            </Operation>
            <Operation id="O2" Operator="CREATE_NAMES"
                                timestamp="2005-09-21 13:23:48">
                <Parameter id="P1" type="topic" inout="in">modern_philosopher
                </Parameter>
                <Parameter id="P2" type="string" inout="in">Modern Philosopher
                </Parameter>
            </Operation>
            ...
        </Transaction>
    </Merge>
</MergeLog>
```

그림 7 MergeLog 데이타의 예시

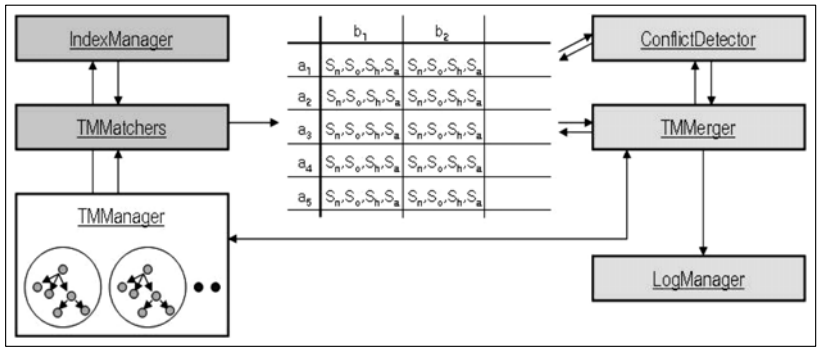

그림 8 토픽맵 통합을 위한 모듈 개념도

능을 가진다. TMMatcher는 두 토픽맵의 모든 토픽쌍들에 대해 유사값을 계산하여 의미적으로 유사한 매핑 집합을 산출하고 이를 토픽쌍들의 매핑 행렬(정의 2에서의 M_{AB})로 저장한다. 매핑 행렬은 통합 모듈들의 입력으로주어진다. ConflictDetector는 매핑 토픽 사이의 충돌이 있는지를 판단하는 모듈로서 TMMerger에게 매핑 토픽 사이의 충돌 유형을 알려준다. TMMerger는 매핑 토픽들 사이의 통합을 처리하는 모듈이며 통합 과정은 LogManager에게 보내어 MergeLog로 기록된다.

6.2 구현 결과

통합 모듈의 성능을 평가하기 위해서는 동일한 지식 도메인에 대해 서로 다른 전문가에 의해 생성된 여러 토픽맵들이 필요하다. 본 연구의 선행 연구[9]에서는 철학 분야의 동양과 서양의 고전 텍스트들을 분석하고 해설한 내용으로부터 개념을 추출하고 지식 구조화한 다음 토픽맵으로 구현하였다. 따라서 본 논문에서는 이 철학 분산 토픽맵들을 기본 실험 데이타로 하여 이들 사이에 대응되는 매핑 토픽들을 발견하고 충돌 탐지 및 해결함으로써 통합 토픽맵을 생성하는 과정을 실험하고자 한다.

6.2.1 실험 데이타 및 결과

실험을 위한 데이타는 세 그룹으로 A 그룹은 철학 분야에 대하여 동일한 전문가 그룹에 의해 생성된 온톨로지들의 그룹으로서 동양철학 전문가와 서양철학 전문가에 의해 생성된 동양철학, 서양근대철학, 서양현대철학 온톨로지들로 구성되어 있다. 여기서 동일한 전문가 그룹은 동일한 프로젝트에 소속된 서로 다른 전문분야의 연구자들을 의미한다. A 그룹의 온톨로지들은 상위 수준의 철학 지식 분류체계와 참조 토픽들을 공유하고 있으며 각각 인스턴스에 적합한 어커런스와 연관관계를 정의하고 있다.

B 그룹은 철학 분야에 대하여 서로 다른 전문가 그룹에 의해 생성된 온톨로지들의 그룹으로서 야후 코리아 포털에서 제공하는 백과사전에서 서양 근대철학과 현대철학 부분을 토픽맵으로 작성한 온톨로지들을 가진다. C 그룹은 철학외의 다른 지식 분야의 온톨로지로서 야후 코리아 백과사전과 네이버 백과사전에서 독일문학 부분을 토픽맵으로 작성한 온톨로지들을 가진다. B 그룹과 C 그룹은 철학 온톨로지외의 데이타들과의 매핑 및 통합을 실험함으로써 제안하는 기법의 일반적인 적용이 가능함을 보이기 위한 데이타이다.

표 3에서 동양철학 온톨로지는 한국, 중국, 인도 철학의 지식을 포함하고 있으며 서양근대철학 온톨로지는 칸트, 헤겔, 데카르트 등 근대시대의 철학 지식을 포함

표 3 A, B, C 그룹의 온톨로지의 구조적 특성

온톨로지	A 그룹			B 그룹		C 그룹	
	동양철학 온톨로지 (T_1)	서양근대철학 온톨로지 (T_2)	서양현대철학 온톨로지 (T_3)	야후 근대철학 (T_4)	야후 현대철학 (T_5)	야후 독일문학 (T_6)	네이버 독일문학 (T_7)
최대 깊이	11	10	9	5	5	4	4
토픽 수	1821	782	1087	95	231	121	210
토픽타입 수	1335	388	598	5	5	8	8
어커런스타입 수	71	59	65	5	5	5	12
연관관계타입 수	49	43	44	2	2	2	2
역할타입 수	22	15	18	2	2	2	2
PSI 수	667	331	352	5	5	8	8

한다. 그리고 서양현대철학 온톨로지는 마크르스, 흄, 러셀 등 현대시대의 철학 지식을 포함한다. 온톨로지에 구성된 철학 지식 내용은 철학자, 철학문헌, 철학이론, 철학용어 및 철학 텍스트의 내용 지식으로 구성되어 있다. B 그룹과 C 그룹의 온톨로지들은 백과사전 내용을 토픽맵으로 재작성한 것이기 때문에 토픽들의 분류 구조가 계층적 구조로 단순하며 토픽의 속성 또한 '해설', '관련항목' 등으로 비교적 간단한 수준이다. 실험 결과는 크게 매핑 성능을 보이는 결과와 통합 성능을 보이는 결과로 나누어 제시되는데 통합 결과의 경우 일대일 매핑 토픽들 사이에 존재하는 유형별 충돌의 수와 통합 후 생성된 결과 토픽맵의 개체수로써 제시되고 있다. 실험 온톨로지들의 매핑 및 통합 결과는 표 4에 주요 항목별로 수치 데이타로 표현하였다.

6.2.2 매핑 성능

실험 온톨로지 쌍들 사이의 매핑 결과에서 전문가매핑집합은 철학 전문가에 의해 수작업으로 생성된 매핑 원소들의 집합이고 시스템매핑집합은 토픽맵 매핑 모듈에 의해 자동적으로 생성된 매핑 원소들의 집합이다. 일치매핑집합은 수동 및 자동 생성 매핑집합의 공통 요소들의 집합으로써 두 매핑 집합에 공통적으로 존재하는 원소들을 가진다. 이 수치를 통해 시스템에 의해 자동 생성된 매핑집합이 약 80% 이상의 재현율(recall)을 보임을 알 수 있다[7]. 또한 매핑 결과에서 (T_2, T_3)의 매핑 원소수가 가장 많이 나오고 (T_2, T_6)의 매핑 원소수가 가장 적게 나옴을 알 수 있는데, 이는 서양근대철학과 서양현대철학 사이의 유사성이 높은 반면 서양근대철학과 독일문학 사이에는 지식의 유사성이 적기 때문이다.

6.2.3 충돌 탐지 및 통합 성능

통합 결과에는 토픽맵의 통합 방향, 충돌 유형별 발생빈도, 통합 토픽맵의 요소 개수 등의 세 가지 유형의 결과 데이타를 보이고 있다.

통합방향의 결정. 통합 방향 결정에 있어서 (T_2, T_3)의 경우 서양현대철학 온톨로지가 서양근대철학 온톨로지보다 크기가 크기 때문에 $T_2 \rightarrow T_3$로의 통합 방향을 가지며 (T_1,T_2)의 경우는 동양철학 온톨로지의 크기가 더 클 뿐만 아니라 LMP 값이 5로서 LMP 값이 4인 서양근대철학 온톨로지보다 더 크기 때문에 $T_2 \rightarrow T_1$의 통합 방향을 가진다.

충돌 탐지. 충돌 유형별 발생빈도에서는 대부분 토픽명 충돌이 많은 수를 차지하고 다음으로 속성 충돌이 많이 발생한다. 이는 토픽맵 요소들의 대부분이 토픽이므로 토픽들 사이의 관계에서 발생하는 구조적 충돌인 일반화 및 집합 충돌에 비해 토픽명 충돌 및 속성 충돌이 더 많은 발생 빈도를 가지기 때문이다. 그림 9에서 보면, 동양철학 온톨로지의 경우 '철학'→'동양 철학'→'철학자'→{'한국 철학자', '중국 철학자', '인도 철학자'}의 계층 구조를 가지는 반면 서양근대철학 온톨로지는 '철학'→'서양근대 철학'→'철학자'→{'임마누엘 칸트', '헤겔', '데카르트', ...} 등 철학자 하위에 칸트, 헤겔 등의 인스턴스를 가지는 계층 구조를 보인다. 마찬가지로 '동양철학' 및 '서양근대 철학' 하위의 철학문헌, 철학이론, 철학학파 등에서도 계층 구조의 차이를 가진다. 그러므로 (T_1, T_2)에서 일반화 충돌은 각각 (T_1:철학, T_2:철학), (T_1:철학자, T_2:철학자), (T_1:철학문헌, T_2:철학문헌), (T_1:철학이론, T_2:철학이론), (T_1:철학학파, T_2:철학학파) 토픽쌍에서 발생한다.

일반화 충돌은 4.2.3절에서 서술한 바와 같이 대응 요소를 찾는 매핑 단계에서 고려해야 한다. 만일 일반화 충돌을 고려하지 않을 경우 그림 10과 같은 통합 결과를 얻게 된다. 그림 9에서는 철학자 계층 구조만을 보이고 있는데 여기서 철학과 철학자 토픽은 통합된 토픽이다. 통합 결과의 오류는 철학자 토픽을 통합한 데서 발생된다.

표 4 실험 데이타 A, B, C 그룹의 매핑 및 통합 결과

온톨로지 쌍			(T_1,T_2)	(T_1,T_3)	(T_2,T_3)	(T_2,T_4)	(T_3,T_5)	(T_2,T_6)	(T_6,T_7)
매핑결과	전문가매핑집합		81	112	214	57	93	3	83
	시스템매핑집합		113	132	276	69	116	7	97
	일치매핑집합		69	95	203	51	85	3	76
통합결과	통합방향		$T_2 \rightarrow T_1$	$T_3 \rightarrow T_1$	$T_2 \rightarrow T_3$	$T_4 \rightarrow T_2$	$T_5 \rightarrow T_3$	$T_6 \rightarrow T_2$	$T_6 \rightarrow T_7$
	충돌빈도	토픽명충돌	21	35	53	17	29	1	13
		속성충돌	29	27	14	43	71	3	65
		일반화충돌	5	3	0	2	2	0	0
		집합충돌	3	3	1	1	1	0	0
	통합토픽맵	통합토픽수	2468	2812	1673	828	1233	900	255
		통합어커런스수	80	87	69	59	65	59	9
		통합연관관계수	61	55	45	45	46	45	2
		통합역할타입수	24	26	18	17	20	17	2

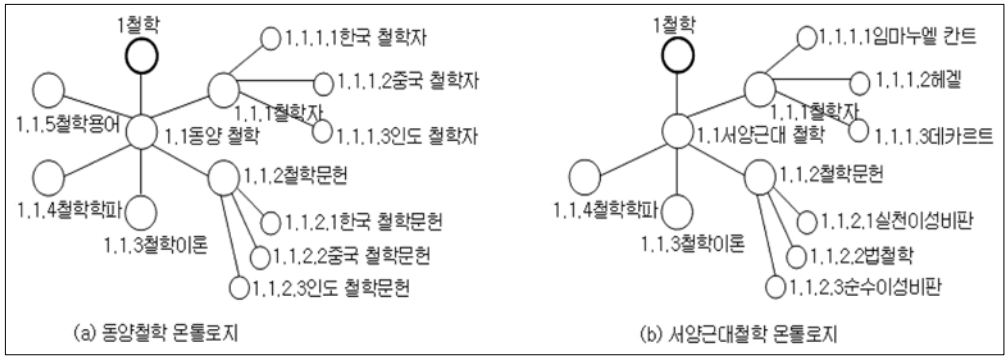

그림 9 동양철학 온톨로지와 서양철학 온톨로지의 부분적 계층 구조도

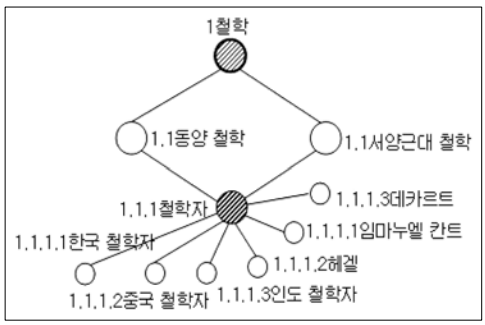

그림 10 일반화 충돌을 고려하지 않은 통합결과

즉, 매핑 단계에서 일반화 충돌을 고려하지 않을 경우 동양철학 온톨로지와 서양철학 온톨로지의 철학자 토픽들은 토픽명의 일치로 인해 최상의 유사값을 가지게 된다. 따라서 두 토픽은 통합 후보가 되고 이에 따라 통합 모듈이 통합할 경우 그림 10와 같은 결과를 생성하게 된다. 문제는 서양근대 철학에서 철학자로 검색해 나갈 때 서양근대 철학자와 관련 없는 한국 철학자, 중국 철학자, 인도 철학자를 발견하게 되고 반대로 동양 철학에서 철학자로 검색해 나갈 때에는 동양 철학자가 아닌 칸트, 헤겔, 데카르트 등을 발견하게 된다. 일반화 충돌을 고려할 경우 매핑 단계에서 두 철학자 토픽의 유사값은 낮아지게 되고 통합 결과에서는 서로 별개의 토픽으로 존재하게 된다.

표 4의 통합 토픽맵의 요소별 개수를 가리키는 수치에서 특징적인 부분은 '통합토픽수' 항목의 수치가 두 소스 토픽맵의 전체 토픽 수에서 일치매핑집합의 토픽 수를 차감한 수보다 더 크다는 데 있는데 이것은 일대다의 매핑이 존재하기 때문이다. 즉, 하나의 토픽이 여러 토픽과 비슷한 유사값을 가질 경우 이 토픽은 다중 매핑을 가지게 되고 통합 또한 매핑된 토픽들에 대해 각각 실행된다.

통합 모듈의 성능을 평가하기 위해 실험 데이타의 각 토픽맵 쌍에 대하여 토픽맵 검색어인 Tolog 질의어를 이용한 통합 전과 후의 검색 결과를 비교하였다. 매핑 수가 극히 적은 (T_2,T_6)를 제외한 나머지 토픽쌍에 대해 각각 50개의 Tolog 질의어를 작성하였다. 예를 들어 (T_1,T_2) 토픽쌍에 대한 질의어는 T_1에서만 검색될 수 있는 질의어, T_2에서만 결과를 검색할 수 있는 질의어, T_1과 T_2의 매핑으로 인해 양쪽에서 결과를 검색할 수 있는 질의어를 포함한다.

성능 평가 척도는 정보검색에서 사용하는 정답율(precision)과 재현율(recall)을 사용한다. 통합 전의 두 소스 토픽맵에서의 검색 결과와 통합 토픽맵에서의 검색 결과를 비교하여 정답율과 재현율을 평가해 봄으로써 통합 토픽맵이 정보의 손실없이 두 소스 토픽맵을 완전히 통합한 것인지 여부를 확인할 수 있다. 아래 정답율과 재현율을 구하는 수식에서 **P**는 두 소스 토픽맵으로 부터 검색된 결과 집합이고 **R**은 통합 토픽맵으로 부터 검색된 결과 집합이다. 그리고 **I**는 **P**와 **R**의 교집합이다. 그림 11은 각 토픽맵쌍의 정답율과 재현율을 보이는 그래프이다. 전체적으로 90% 이상의 정답율과 재현율을 보이고 있으며 통합 토픽맵이 소스 토픽맵을 손실없이 통합함을 보이고 있다.

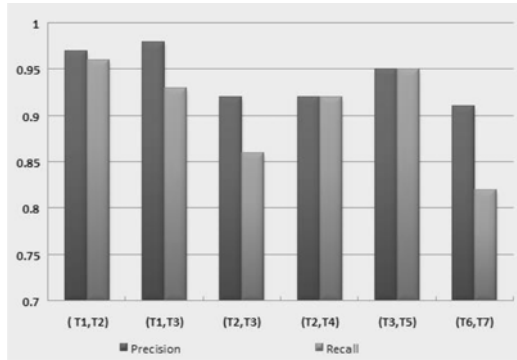

그림 11 통합 성능 평가 그래프

$$precision = \frac{|I|}{|R|} \quad recall = \frac{|I|}{|P|}$$

두 소스 토픽맵 사이에 매핑되는 토픽이 많을 경우 재현율이 낮게 나오고 있다. 그 이유는 두 소스 토픽맵의 매핑 토픽들이 통합 토픽맵에서는 하나의 토픽으로 통합되기 때문에 동일한 질의어에 대해 통합 토픽맵의 질의 결과 수가 더 적게 나오기 때문이다. (T_2,T_3)와 (T_6,T_7)의 경우 전체 토픽 수에 비해 매핑 토픽 수의 비중이 높기 때문에 재현율이 낮게 나왔다.

6.2.4 선행 관련연구와의 비교

본 논문의 매핑 및 통합 모듈(TM-MAP)과 타 관련 연구와의 간접적인 성능 비교를 표 5에서 보이고 있다. 여기에서 패턴 컬럼은 매핑 단계에서 대응되는 토픽을 탐색하기 위해 적용하는 기법을 가리키는 것으로 T(Terminological)는 용어들 사이의 유사성을 계산하기 위한 문자열 비교 기법이고 IS(Internal Structure)는 속성과 같은 내적 구조 사이의 비교 기법이다. ES(External Structure)는 용어들의 계층구조 사이의 비교 기법이고 E(Extensional)는 용어들 사이의 연관관계에 대한 비교 기법이다. 그리고 I(Instance)는 인스턴스 수준에서의 비교 기법이다. 본 논문의 매핑 및 통합 방법은 다른 방법에 비하여 더 많은 비교 기법들을 적용하고 있으며 실제적인 철학 온톨로지를 대상으로 실험한 결과를 제시하고 있다. 또한 토픽맵의 모델 특성에 기반하여 개체 유형별로 매핑 여부를 판단함으로써 모든 개체들을 서로 비교하는 N*M 보다 낮은 복잡도를 가진다.

7. 결론 및 향후연구

본 논문에서는 두 소스 토픽맵으로부터 의미적으로 대응되는 매핑 토픽들을 탐색하고 이들을 중심으로 두 토픽맵을 하나로 통합하기 위한 토픽맵 통합 프로세스 및 T-MERGE 연산자를 정의하였다. 또한 매핑 토픽들을 통합할 때 발생하는 충돌 유형을 엘리먼트기반 충돌, 구조적 충돌, 임시적 충돌 등으로 세분화하여 분류하였으며 각각의 충돌을 탐지 및 해결하는 기법을 정의하였다. 매핑 토픽들을 통합하는 데 있어서 토픽 생성, 속성 추가 등의 구체적인 단위 연산들을 정의하였으며 통합 과정의 자세한 기록을 위해 실행된 단위 연산의 종류와 결과에 대해서 보관하는 MergeLog의 구조와 로그 관리자를 설계, 구현하였다.

토픽맵 관리 시스템인 K-Box의 하위 모듈로 구현된 통합 모듈은 철학 온톨로지와 야후 및 네이버의 백과사전 데이타를 토픽맵으로 변환한 온톨로지들을 대상으로 실험하여 성능을 평가하였다. 다중전략 매칭 기법에 의해 주어진 매핑 토픽 리스트를 입력으로 하여 일대일로 대응되는 매핑 토픽들을 하나의 토픽으로 통합하고 나머지 비매핑 토픽들을 상하 계층 관계 및 연관관계를 보존하면서 추가함으로써 통합된 토픽맵을 생성하였다. 주어진 실험 데이타에 의한 통합 모듈의 성능 평가에서는 철학 전문가가 XML 편집기로 소스 토픽맵을 통합하는데 3일이 소요된 것에 반하여 통합 모듈은 대략 1.5시간 내에 동일한 결과를 보였다.

본 논문의 매핑과 통합 기법은 의미적으로 동등한 대응 관계의 두 토픽만을 고려한다. 하나의 토픽이 다른 토픽의 하위 개념인 SubclassOf 또는 다른 토픽의 구성요소인 PartOf 등의 대응 관계는 고려하지 않고 있다. 온톨로지 통합에서 두 토픽을 하나의 토픽으로 합치는 동등 관계 외에 의미적으로 superclass-subclass 또는 container-containee 관계를 설정하기 위해서는 온톨로지 매핑 단계에서 이러한 의미적 관계를 탐지할 수 있어야 한다. 향후연구에서는 지식 도메인 컨텍스트를 반영한 의미사전 등을 활용하여 이러한 의미관계를 파악하는 기법과 이를 활용한 통합 기법에 대해 연구하고자 한다.

참 고 문 헌

[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," Scientific American, 279, 2001.

[2] R. A. Pottinger and P. A. Bernstein, "Merging Models Based on Given Correspondences," In Proceedings of the 29th VLDB Conference, Berlin, Germany 2003.

[3] F. Giunchiglia and P. Shvaiko, "Semantic matching," In The Knowledge Engineering Review

표 5 관련연구와의 비교

Methods	L	P	D	R	M	C
Anchor-PROMPT	Graph	T/ES	HPKB	Merging	Interactive	N*M
Chimerae	Graph	T/E	Toy	Merging	Interactive	N*M
IF-MAP	Graph	T/I	Toy	Mapping	Batch	N*M
FCA-Merge	Graph	T/I	Toy	Merging	Interactive	N*M
TMRM	Topic Maps	T	-	Merging	Batch	N*M
TM-MAP	Topic Maps	T/IS/ES/E	Real Ontology	Merging	Batch	nlogn

Language(L), Patterns(P), Experimental Data(D), Results(R), Merging Type(M), Complexity(C)

Journal, 18(3), 2004.
[4] P. Shvaiko and J. Euzenat, "A survey of schema-based matching approaches," University of Trento, Technical Report #DIT-04-087, 2004.
[5] N. Noy and M. Musen, "PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment," In Proceedings of the National Conference on Artificial Intelligence(AAAI), 2000.
[6] D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder, "An environment for merging and testing large ontologies," In Principles of Knowledge Representation and Reasoning: Proceedings of the Seventh International Conference(KR2000), 2000.
[7] 김정민, 신효필, 김형주, "분산 토픽맵의 다중전략 매핑 기법", 정보과학회논문지(소프트웨어 및 응용), 게재예정.
[8] Michel Biezunski, Martin Bryan and Steve Newcomb, ISO/IEC 13250 TopicMaps.
[9] 김정민, 최병일, 김형주, "텍스트 내용지식 기반의 철학 온톨로지 구축", 정보과학회논문지(컴퓨팅의 실제), 11(3), June 2005.
[10] N. Noy and M. Musen, "Anchor-PROMPT: Using Non-Local Context for Semantic Matching," In Proceedings of the Workshop on Ontologies and Information Sharing at the International Joint Conference on Artificial Intelligence(IJCAI), 2001.
[11] G. Stumme and A. Madche, "FCA-Merge: Bottom-up Merging of Ontologies," In Proceedings of 17th Internaltional Joint Conference on Artificial Intelligence(IJCAI), 2001.
[12] Y. Kalfoglou and M. Schorlemmer, "Information-Flow-based Ontology Mapping," In Proceedings of the 1st International Conference on Ontologies, Database and Application of Semantics, 2002.
[13] R. Pottinger and P. A., Bernstein, "Merging Models Based on Given Correspondences," In Proceedings of VLDB, 2003.
[14] S. Pepper and G. Moore, "XML Topic Maps(XTM) 1.0," TopicMaps.Org http://www.topicmaps.org/xtm.
[15] M. Ehrig and S. Staab, "QOM: Quick ontology mapping", In Proceedings of ISWC, 2004.
[16] 김정민, 박철만, 정준원, 이한준, 민경섭, 김형주, "K-Box: 토픽맵 기반의 온톨로지 관리 시스템", 정보과학회논문지(컴퓨팅의 실제), 10(1), February 2004.

김 정 민
정보과학회논문지 : 소프트웨어 및 응용
제 33 권 제 1 호 참조

신 효 필
정보과학회논문지 : 소프트웨어 및 응용
제 33 권 제 1 호 참조

김 형 주
정보과학회논문지 : 소프트웨어 및 응용
제 33 권 제 1 호 참조