지식베이스의 재사용을 위한 향상된 지식 표현

현우석
한국성서대학교 정보과학부
wshyun@bible.ac.kr

Enhanced Knowledge Representation for Reusing Knowledge Bases

Woo-Seok Hyun
Department of Information Science, Korean Bible University

요 약

본 논문에서는 합법적인 지식 표현이 그것을 사용하는 응용 시스템과 완전히 독립적이 되어야 한다는 점에 대해서 논의한다. 이것은 동형 표현을 제공해 주고 지식베이스 유지보수를 더 쉽게 하며 오류를 줄여줄 뿐만 아니라 몇몇 지식 기반 시스템에서 합법적인 지식을 공유하고 재사용하기 위한 필요조건이다. 본 논문에서는 지식 베이스의 재사용을 위한 향상된 지식 표현을 위해 하이브리드 지식 표현 접근방법을 제안한다. 이것은 합법적인 표현 컴포넌트를 공통 용어로 사용하여 컴포넌트에 연결함에 의해서 응용시스템의 표현 컴포넌트와 통합한다. 때때로 이와 같은 통합이 응용시스템의 제어 흐름 요구사항에 따르지 않는 경우도 있다. 이 문제를 해결하기 위해서 본 논문에서는 일방적인 의존성을 도입해서 재사용되어지는 지식 베이스가 변경되지 않도록 하는 해결책을 제안한다. 또한 지식 기반 의사결정 지원 시스템에서 합법적인 지식베이스를 통합하기 위한 응용을 논의하여 제안하는 접근방법을 설명한다.

1. 서 론

많은 회사에서, 지식기반 시스템은 질과 효율성을 향상시키고자 회사 업무를 지원해주는 시스템을 구축하기 위한 적절한 방법으로 여겨지고 있다. 대부분의 회사 업무는 조직의 규칙과 법칙에 기초가 되기 때문에 만약 회사 업무를 지원하는 시스템이 이러한 규칙에 대해서 알고 그 규칙을 따르며 사용자에게 그 규칙을 설명한다면 회사 업무를 지원하는 시스템의 가치는 급격히 증가될 것이다[1]. 합법적인 지식을 표현하는 기술이 많이 발전되었다고 할지라도[2], 합법적인 지식을 이미 존재하거나 개발된 응용 시스템의 지식베이스에 실제적으로 통합하는 문제는 아직까지 해결되지 않고 있다. 다음과 같이 중요한 두 가지 문제점을 들 수 있다. 첫째, 우리는 적절한 효율성을 지니는 시스템이 필요하다는 것이다. 둘째, 우리는 실제 시스템의 어떤 속성과도 완전히 합법적인 지식 표현을 가지고 싶어 한다는 것이다. 그 이유는 다음과 같다.

• 법칙(law)과 조직의 규칙(rule)에 관련된 지식은 실제적으로 하나 이상의 응용 시스템에서 사용될 수 있고 그러므로 재사용되어야만 한다. 그래서 지식을 한번만 표현하고 필요할 때마다 가져다 쓰는 것이 필요하다. 특별히 유지보수 목적을 위해서도 재사용성은 아주 바람직하다. 지식 공유의 요구사항은 합법적인 지식에만 한정된 것이 아니고 지식 기반 시스템 영역에서 일반적인 관심에도 적용되는 것이다[3].

• 법칙과 지식은 지식베이스에서 문서 출처를 가지고서 동형으로 표현되어야만 한다[4]. 이것은 사용자에게 적절한 설명을 한다든지 지식 베이스를 유지보수하기 위해서 필요조건이다.

본 논문에서는 위에서 언급한 요구 조건을 수행하는 문제를 통합하는 접근방법을 설명한다. 또한 최근에 만들어진 의사결정 지원 시스템에 실제적으로 적용되는 방법을 제안한다. 이것을 위해서 2절에서는 어떤 지식 유형이 의사결정 지원 시스템에 의해 사용되어질 수 있으며 어떤 표현 형식이 선택되어지는 지를 설명한다. 3절은 본 논문에서 가장 중요한 부분이며 합법적인 지식을 통합하기 위한 몇 가지 대안에 대해서 논의한다. 그리고 통합 방법을 제시해 주고 2절에서 도입된 의사결정 지원 시스템을 위해서 어떤 역할을 하는지를 보여준다. 마지막으로 4절에서는 결론과 향후 연구과제에 대해 언급한다.

2. 회사 업무를 위한 지식기반 의사결정 지원 시스템

보험회사와 같은 다양한 회사에서 고객을 지원하기 위한 회사 직원은 더 이상 특정 회사 업무 영역을 다루는 전문가들이 아니라 다양한 업무들은 다루는 사람이 되어야 한다. 이러한 목적을 위해서 지식기반 의사결정 지원 시스템이 개발되고 있다. 직원이 어떤 새로운 사실을 입력하기를 요청받거나 어떤 전이가 왜 합법적인지 비합법적인지를 알고자 할 때마다 다음과 같은 설명을 요청할 수 있다.

- 많은 정보가 왜 case로 다루어져야 하는지
- 시스템이 하나의 상태 전이가 (비)합법적이라는 결정을 어떻게 하는지

이러한 기능을 성취하기 위해서 의사결정 지원 시스템은 각각의 회사 업무가 다음과 같이 표현되도록 요구한다.

- 상태(state)
- 행동(action) 혹은 하부 태스크(sub-task)
- 객체(object)
- 전제조건(precondition)

보다 자세한 분석은 회사 업무의 표현은 완전하게 서로 다른 두 가지 종류의 지식과 관련되어 있다는 것을 보여준다.

- 다루어질 객체와 객체가 속한 개념 클래스는 용어 표현 컴포넌트를 요구한다.
- 행동은 상태 전이를 발생시키고 행동의 전제조건은 변화의 지식을 다룰 수 있는 이벤트 표현 컴포넌트를 요구한다.

결론적으로 본 논문에서는 개념 클래스와 객체를 표현하는 용어 논리[5]와 상황 계산에 근거를 둔 일차 언어(first-order language)[6]를 통합하는 하이브리드 지식 표현 시스템을 구축하고자 한다. 각각의 회사 업무는 하나의 시작 상태와 한 가지 이상의 종료 상태를 지닌다. 2.1과 2.2절에서는 두 가지 표현 구성요소를 나타내고, 2.3절에서는 활동과 변화를 표현하기 위해서 사건 컴포넌트에 의해 제공되는 추론 서비스를 설명한다.

2.1 용어 컴포넌트(Terminology Component)

용어 컴포넌트는 용어 논리에 의해서 실제적으로 주어지며 제안하는 하이브리드 표현 시스템의 변화를 표현하는 컴포넌트이다.

본 연구에서는 용어 구성요소에서 개념 클래스의 표현과 개념 클래스의 실례인 객체를 구분하였다. 생명보험 응용 업무를 위한 용어는 그림 1과 같으며 개념 클래스는 다음과 같이 술어로 표현된다.

- *class(c)*는 클래스 *c*를 존재하게 한다.
- *has-relation(c, r, rc)* 는 클래스 *c*의 모든 예는 클래스 *rc*의 적어도 하나의 객체에 대해서 관계 *r*을 가진다.
- *has-property(c, p, d)*는 클래스 *c*의 모든 예는 전문영역 *d*로부터 하나의 값을 가지는 속성 *p*를 가진다.
- *is-a(c_1, c_2)*는 c_1은 c_2의 하부 클래스임을 나타낸다.

```
class (office-task)
has-relation (office-task, has-part, thing)

class (application-for-a-life-insurance)
is-a (application-for-a-life-insurance, office-task)

class (contract)
has-relation (contract, has-parties, person)
has-relation (contract, is-signed-by, person)

class (application-form)
is-a (application-form, form)
is-a (application-form, contract)
has-relation (application-form, has-parties, person)
has-relation (application-form, is-signed-by, person)
has-property (application-form, sum-insured, [0, 100'000'000])
has-property (application-form, result-of-risk-analysis, {accept-application, reject-application})

class (person)
has-property (person, social-security-nbr, ******-*******)
has-property (person, name, AnyString)

class (insurance-policy)
has-property (insurance-policy, policy-nby, ****.****.***)
has-relation (insurance-policy, policyholder, person)
has-relation (insurance-policy, result-from, application-form)
```

그림 1 생명보험 응용 업무를 위한 용어

2.2 활동과 변화를 표현하는 사건 컴포넌트

회사 업무의 정적인 면이 용어 컴포넌트로 표현되어지는 반면에, 그것의 동적인 부분은 다른 표현 언어를 요구하며 분리된 컴포넌트로 표현된다. 각각의 회사 업무를 위해서 정의된 활동의 가능한 결과와 각각의 활동을 위한 전제조건과 변화가 이루어진다. 각 활동 a가 회사 업무 ti의 부분으로 수행된다는 것은 action(a, t_i, s_1, s_2)의 형식으로 표현할 수 있는데, 여기서 s_1은 초기 상태이고 s_2는 결과 상태를 나타낸다.

그림 2는 생명 보험 응용 업무를 위한 회사 태스크의 활동들과 전제조건들의 예를 보여준다.

Actions with necessary preconditions:

$\forall ti$:(action(issue-insurance-policy, t_i, S_1, S_2)=> application-accepted(t_2,S_1)) (AP1)

$\forall ti$:(action(supervision-by-division-head, t_i, S_1, S_3)=> application-accepted(t_2,S_1)) (AP2)

Actions with necessary and sufficient preconditions:

$\forall ti$:(action(risk-analysis, t_i, S_0, S_1)<=> exists-application-form(t_i,S_0)) (AP3)

$\forall ti$:(action(reject-application, t_i, S_1, S_4)<=> instance-of(t_i, application-for-a-life-insurance, <t_i,S_1>)∧

related-to(a, result-of-risk-analysis, reject-application, <t_i,S_1>)∧
related-to(t_i,, has-part, a, <t_i,S_1>))) (AP4)
⋮
Action definitions:
∀ti,S_1,S_2: (action(risk-analysis, ti, S_1, S_2)=> (AD2)
∃ip: (instance-of(ip, insurance-policy,<t_i,S_2>∧
∃n: property-of(ip, policy-of(ip, policy-nbr, n,<t_i,S_2>) ∧ ⋯) ∧
∃a: (instance-of(a, application-form,<t_i,S_1>∧
related-to(t_i, has-part, a,<t_i,S_1>) ∧
related-to(ip, result-from, a,<t_i,S_2>)) ∧
∀i,c: (instance-of(i, c, <t_i,S_1>)=>
instance-of(i, c, <t_i,S_2>))∧
OTHER FRAME-AXIOMS)

그림 2 생명보험 응용 업무를 위한 회사 태스크의 행동과 전제들의 예

2.3 시스템에서 추론

사용자가 활동을 시작하게 되면 적절한 활동 원자(atom) action(...)이 지식 베이스에 추가된다. 활동 정의에 기반을 둔 시스템은 이미 존재하는 전 상태의 객체를 가지고 새로운 상태 표현을 추론하며 가능한 여러 가지의 상태 표현을 추론한다. 그림 2에서 활동(action(risk-analysis, t_i, S_0, S_1)은 위험 분석의 결과를 응용 형식에 추가하는 효과를 지닌다. 만약 위험 분석의 결과가 응용을 거절한다면, 사용자는 "reject-application"이라는 활동을 계속하게 된다. 그렇지 않으면, 그림 2에서 "issue-insurance-policy" 혹은 "supervision-by-division-head" 등 실행할 수 있는 활동들 중 하나를 수행할 수 있다. 어떤 활동이 회사 규칙에서 정의된 환경 하에서 선택될 수 있는 지는 3절에서 언급한다.

3. 합법적인 추론 컴포넌트를 통합하기

앞 절에서 언급한 의사결정 지원 시스템은 법칙과 회사 규칙이 고려되었을 때 유용하게 사용된다. 이것을 얻는 직접적인 방법은 법칙과 규칙을 1차 논리로 표현하고 그것들을 활동의 전제조건으로 통합하는 것이다. 하지만 이것은 몇 가지 단점을 지니고 있어서 항상 좋은 생각은 아니다.

• 법칙과 규칙의 지식은 내재적으로 회사 태스크의 지식과 얽혀 있어서 재사용되기 힘들다. 이것은 합법적인 지식이 다른 지식기반 응용에서도 필요하기 때문에 상당한 약점이 될 수 있다.

• 법칙과 규칙의 지식 표현은 더 이상 동형이 아닐 수도 있다[4]. 그리하여 유지보수가 더 어려워지고 적절한 설명을 생성하는 것이 불가능해질 수도 있다.

• 우리가 하나의 큰 동형의 지식 베이스를 가진 것처럼 지식베이스의 구조와 유지보수는 매우 어렵게 된다.

• 다른 표현 형식주의와 합법적인 지식과 사건 지식을 위해 다른 사유를 제공하는 것은 불가능하다. 합법적인 지식과 사건 지식이 일반 목적 사유로 통합되는 것보다 더 효과적이기 때문에 다르고 특화된 사유들은 아주 바람직하다.

이러한 단점을 피하기 위하여 하이브리드 지식 표현 시스템[7]에 근거를 둔 아이디어를 사용하기를 제안한다[8]. 합법적인 지식의 다양한 종류를 다른 하부 컴포넌트로 조직화하는 이론적 논의는 Oskamp가 언급했던 것[9]을 참조로 하였다.

그림 3은 법칙과 규칙 표현의 실례를 보여준다.

Stating that the regulations below must be complied with:
∀co, t_i, s : law(VVG, signing-of-contracts, co, t_i, s)
∀co, t_i, s : law(ZGB296/1, signing-of-contracts, co, t_i, s)
∀ip, t_i, s : s-life-regulation(underwriting-regulation, valid-insurance-policy, ip, t_i, s)
∀ip, t_i, s : s-life-regulation(underwriting-regulation, supervision-by-division-head, a, t_i, s)

Definitions of Regulations:
law(VVG,signing-of-contracts,co,t_i,s)=> (R1)
(instance-of(co, contract,<t_i, s>) =>
∀pa: (related-to(co, has-parties, pa, <t_i, s>)=>
related-to(co, is-signed-by, pa, <t_i, s>)))
law(ZGB296/1,signing-of-contracts,co,t_i,,s)=> (R2)
(instance-of(co, contract,<t_i, s>) =>
∀pa: (related-to(co, has-parties, pa, <t_i, s>)∧
under-age(pa, <t_i, s>) =>
∃g: related-to(co, has-guardian, g, <t_i, s>) ∧
related-to(co, is-signed-by, g, <t_i, s>))))
s-life-regulation(underwritting-regulation, valid-insurance-policy, ip, t_i, s)=> (R3)
(instance-of(ip, insurance-policy, c<t_i, s>) =>
∃a: (instance-of(a, application-form, <t_i, s>)∧
related-to(ip, result-from, a, <t_i, s> ∧
valid-application-form(a, t_i, s)))
⋮

그림 3 법칙과 규칙 표현의 실례

3.1 하이브리드 통합

사건(event) 컴포넌트와 법칙 컴포넌트는 연결 서술자

(connection predicate)를 사용하여 연결한다. 그림 2, 3에서 규칙 (R3)는 활동 정의(action definition) (AD2)의 부분이 될 수 있다. (R3)는 활동 "issue-insurance-policy"가 수행된 다음에 체크된다. 그래서 활동 정의에 부분이 되어야만 하고 전제조건의 부분이 되면 안 된다. 결론은 연결 서술자를 가지는 "s-life-regulation"과 같다.

$\forall t,S_1,S_2$: action(issue-insurance-policy,t_i, S_1, S_2)=>
  $\exists$ip: instance-of(ip, insurance-policy, <t_i,S_2>)∧
    s-life-regulation(
      underwriting-regulation,
      valid-insurance-policy, ip, <t_i,S_2>∧
  $\exists$n: property-of(ip, policy-nbr, n, <t_i,S_2>)∧
    ⋮

법칙 컴포넌트를 의사결정 지원 컴포넌트와 통합하기 위한 구조는 그림 4와 같으며, 연결 규칙을 사용한 최종 구조는 그림 5와 같다.

통합된 의사결정 지원 시스템의 프로토타입은 Windows 환경 하에서 Prolog를 사용하였으며, 메모리가 128M인 Pentium IV Personal Computer상에서 시뮬레이션 시스템을 구현하였다.

3.2 응용 시스템에서 제어 흐름을 조정하기

본 논문에서 논의되고 있는 의사결정 지원 시스템은 모든 회사 업무가 표현된 법칙과 규칙을 확증한다는 것을 확실하게 하고 있다. 하지만 의사결정 지원 응용에서 해결하지 못하고 있는 다음과 같은 문제점이 있다.

첫째, 법칙과 규칙의 몇몇 위반은 응용의 견해에서 너무 늦게 발견된다. 법칙을 재형성하는 것은 특정 하부 태스크가 완성된 후 어떤 사실들이 알려지기 전에 위반이 발견되기 어렵기 때문에 도움이 되지 않는다.

둘째, 가끔 법칙이나 규칙의 위반이 발생했을 때, 하부 태스크만 초기화하는 것이 필요하다.

첫 번째 문제는 해당 법칙에 내재적이기 때문에, 회사 업무 처리에서 초기에 적절한 확인을 강제적으로 함에 의해서 해결할 수 있다. 두 번째 문제는 어떤 규칙이 실패했을 때 수행되는 하부 태스크는 규칙에 대한 부정 참조를 가지는 전제조건을 가지게 함으로 해결할 수 있다.

4. 결론 및 향후 연구 과제

본 논문에서는 합법적 지식 표현이 그것을 사용하는 응용 시스템과 완전히 독립적이어야 한다는 것을 논의했는데, 다음과 같은 장점을 지닌다.

- 동형 표현이 가능하다.
- 합법적 지식은 그것을 필요로 하는 모든 시스템에서 공유되고 재사용될 수 있다.
- 지식베이스의 구조화와 유지보수는 쉽고 에러가 덜 발생하게 된다.
- 응용 시스템에서 법칙 컴포넌트와 표현 컴포넌트를 위해서 특별한 논리를 도입하였기 때문에, 전체적인 추론 관점에서 효율성을 높이게 되었다.

응용 시스템에 의해서 재사용 가능한 합법적인 지식베이스를 만들기 위하여 개념 클래스와 객체를 표현하는 용어 논리[5]와 상황 계산에 근거를 둔 일차 언어(first-order language)[6]를 통합하는 하이브리드 지식 표현 시스템을 제안하였다. 때때로 통합이 응용 시스템의 제어 흐름 요구사항을 따르지 않기 때문에 응용 지식

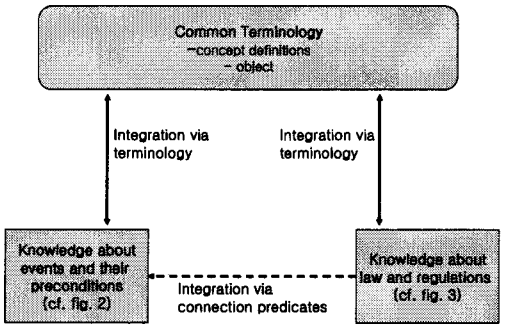

그림 4 법칙 컴포넌트와 의사결정 지원 컴포넌트를 통합하기 위한 구조

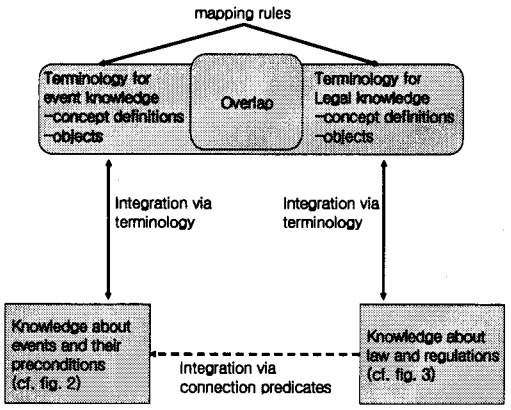

그림 5 법칙 컴포넌트와 의사결정 지원 컴포넌트를 통합하기 위한 최종 구조

베이스에서 연결 서술자를 사용하였다.

향후 연구과제로는 응용시스템에 의해서 필요한 용어와 병합된 법칙 컴포넌트가 겹치거나 같지 않기 때문에 용어 병합을 위한 도구와 방법을 제공하는 것에 대한 향후 연구가 남아 있다.

참고 문헌

[1] Stamper, R., "LEGOL: Modelling Legal Rules by Computer," Computer Science and Law, Cambridge University Press, pp.45-71, 1980.

[2] Sergot, M. J., "The Representation of Law in Computer Programs," Knowledge-Based Systems and Legal Applications, Academic Press, pp.3-67, 1991.

[3] Neches, B., Fikes, R., Finin, T., Gruber, T., Patil, R., Senator, T., Swartout, W.R., "Enabling Technology for Knowledge Sharing", AI Magazine, vol. 12, no. 3, pp.36-56, 1991.

[4] Bench-Capon, T. J. M., Coenen, F., "Exploiting Isomorphism: Development of a KBS to Support British Coal Insurance Claims," Proceedings of the 3rd International Conference on AI and Law, pp.62-29, 1991.

[5] Woods, W. A., Schmolze, J. G., "The KL-ONE Family," Computers and Mathematics with Applications, vol. 23, no. 2-5, pp. 133-177, 1992.

[6] McCarthy J., Jayes, P. J., "Some Philosophical Problems from the Standpoint of Artificial Intelligence," Machine Intelligence 4, pp.463-502, 1969.

[7] Ulrich R., Andreas M., "A Hybrid Knowledge Representation Approach to Reusability of Legal Knowledge Bases," Proceedings of th 5th International Conference on Artificial Intelligence and Law, pp.246-255, 1995.

[8] Nebel, B., "What is Hybrid in Hybrid Representation and Reasoning System?," Computational Intelligence, North Holland, pp.217-228, 1990.

[9] Oskamp, A., "Model for Knowledge and Legal Expert Systems," Artificial Intelligence and Law, vol. 1, no. 4, pp.245-274, 1993.