Abduction in Legal Reasoning

Seiichiro SAKURAI* Hajime YOSHINO†

Abstract

In order to acquire legal rules from legal texts, implicit legal requirements of the rules must be identified. Since such implicit legal requirements can be found through understanding process, a lawyer's understanding process of legal texts is analyzed. His understanding process can be viewed as an abductive reasoning process, since he can introduce implicit requirements which are not appeared in legal texts. This paper models such a reasoning process when they interpret the legal texts. Based on the analysis, a knowledge acquisition support system is proposed.

1 Introduction

In order to realize a legal expert system, we must translate legal texts into machine executable forms. Machine translation process is very difficult because of implicitly expressed concepts in legal texts. In other words, lawyers frequently use their legal expertise in order to fill the gap between the real meaning of legal texts and literal meaning of them. In order to realize knowledge acquisition from legal texts, such implicit legal knowledge must be identified. Through the understanding process, he can find legal requirements correctly. This paper analyzes a lawyer's understanding process. Based on the analysis, a knowledge acquisition support system is proposed. Our system acquires new knowledge by using given legal abstractions.

2 Knowledge Representation

2.1 Compound Predicate Formula

Legal knowledge can be represented by *compound predicate formula(CPF)*. Compound predicate formula is a rule, which can be interpreted by an deductive reasoning engine like Prolog. A legal rule can be represented as follows:

$$\overbrace{r_1, r_2, \ldots, r_n}^{\text{legal requirements}} \rightarrow \overbrace{e}^{\text{legal effect}}$$

r_i represents a legal requirement and e represents a legal effect. r_i and e are

*Tokyo Institute Technology
†Meiji Gakuin University

called compound predicate terms. A compound predicate term can be represented as follows:

$$predicate(predicateID, CaseList)$$

predicate is a predicate name, and *predicateID* is an identifier of predicate. The identifier is used as a reference of predicate instance. *CaseList* is a list of pairs and each pair represents case role and filler. Each filler may be a compound predicate term. Case and filler are separated by ":". For example, a concept, "a contract is concluded", is represented as follows:

```
be_concluded(id1,
            contract(id2,[agt:_,obj:_,...]))
```

In the above example, "id1" and "id2" are predicate identifiers, and "agt", which stands for *agent* case, and "obj", which stands for *object* case, are case names. "_" is an anonymous variable. Note that the predicate "contract" appears as a subterm in the above example, and the predicate contract should be proved in order to verify whether the concept holds or not.

2.2 Interpreting CPF

A CPF is a formula for representing legal rules and it directly corresponds to a legal sentence. Legal concepts in a CPF are represented as chunks of primitive legal conccepts. However, since the variables in a CPF seem to be higer order variables, computational interpretation of CPFs is not easy. In order to interpret a CPF, a fixed legal dictionary, including taxonomy, are assumed for eliminating the predicate variables. By using the dictionary, all predicate variables are determined. If all predicate variables are determined, a CPF can be converted into a flat CPF automatically like [5]. A flat CPF allows only identifiers, i.e. constants, and other functional structures. In other words, a flat CPF is a Horn Clause which can be directly interpreted by Prolog.

Figure 1 shows variants of CPF. Legal texts are translated into CPFs by hand or natural language processing. The translated CPFs are normalized by using the dictionary. By the normalization, all flat CPFs can be transformed into the original normalized CPFs. Without the normalization, a flat CPF cannot always transformed into the original CPF. The normalized CPFs are flattened and each sub compound predicate, which should be proved as a subgoal, is added into bodies.

3 Human Understanding Process

A lawyer's understanding process is analyzed by using United Nations Convention on Contracts for the International Sale of Goods. Since the convention apllies to contracts of sale of goods, Article 23, which is applied to "conclusion of contract," is concerned. Article 23 is shown in Figure 2.

Since a legal rule consists of legal re-

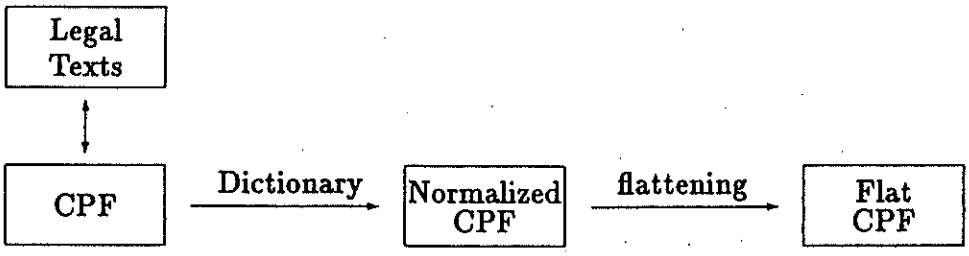


Figure 1: Flat CPF

Article 23

A contract is concluded at the moment when an acceptance of an offer becomes effective in accordance with the provisions of this Convention.

Figure 2: United Nations Convention on Contracts for the International Sale of Goods

quirements and a legal effect, the requirements and the effect must be identified. In Figure 2, even a novice or a beginner of law can identify that the legal effect is "a contract is concluded". Since most of articles describe such legal effects explicitly, the identificatiion of the legal effect is an easy task. However, it is not so easy to identify legal requirements of articles because of implicitly described requirements. Of course, the identification of explicit legal requirements is easy. The article is initially understood as a rule shown in Figure 3. In Figure 3, unused cases are omitted for simplicity. Indeed, some relations between primitive concepts are not sufficiently specified. The real rule must specify such relations.

```
be_concluded(_,[agt:[A,B],
                obj:contract(_,[agt:[A,B]]),
                tim:D]) ←
    become_effective(_,
        [obj:
            acceptance(_,
                [agt:B,
                 obj:offer(_, [agt:A,goa:B]),
                 goa:A]),
        tim:D).
```

Figure 3: Initial Understanding of Article 23

A novice would understand the article as such a rule, since the rule seems to represent the literal meaning of the article. Once such a rule is obtained, even a novice tries to its validity. The rule can be verified by assuming an ideal situation in which a contract is concluded. In such a situation, the acceptance of the offer must be effective. Since the following Article 18-2 describes when the

acceptance becomes effective, the rule of Article 18-2 will be used for the verification of the Article 23.

> Article 18
> 2. An acceptance of an offer becomes effective at the moment the indication of assent reaches the offeror.

A CPF of Article 18-2 is shown in Figure 4.

```
become_effective(_,
          [obj:
           acceptance(IdA,
             [agt:A,
              obj:offer(C,
                    [agt:offeror(B,[obj:C]),
                     goa:A]),
              goa:offeror(B,[obj:C])])]) ←
   reach(_,
    [obj:indication(IdA,
         [agt:A,
          obj:assent(_,
               [agt:A,
                obj:offer(C,
                     [agt:offeror(B,[obj:C]),
                      goa:A]),
                goa:offeror(B,[obj:C])]),
          goa:offeror(B,[obj:C])]),
     goa:offeror(B,[obj:C])]).
```

Figure 4: Article 18-2

If the legal requirement of Article 18-2, "the indication of assent reaches the offeror", is assumed in the ideal situation, "contract is concluded" can be proved by using two rules. In this way, rules translated from legal texts are verified. Because of such successful explanation, novice's understanding process may halt. Are the above rules really correct ones? In spite of such successful explanation, most lawyers can provide its counter-example. For example, by means of "withdrawal of offer," even if the acceptance is reached, the contract cannot be concluded. Therefore, lawyers' understanding process don't halt when only such explanation is made successfully. The lawyers' massive legal expertise seems to enable them to investigate further.

When lawyers understand legal texts, they also confirm whether the legal effect can be derived from the identified legal effects or not. Then they judge the validity of the derivation by using their legal expertise. The verification process can be seen as a backward reasoning process of a legal effect. Figure 5 shows such an explanation.

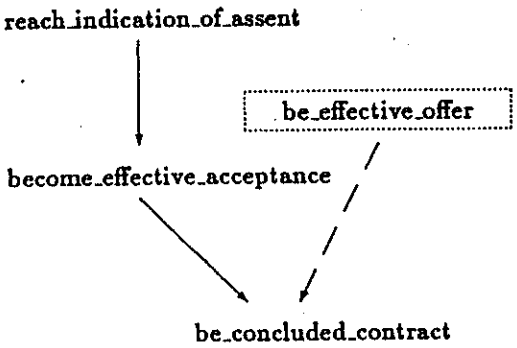


Figure 5: Explanation

In Figure 5, each goal, compound predicate term, is represented by a proposition. The first top goal is the

"be_concluded_contract". In the explanation, although "be_effective_offer" does not appear in the legal texts, it is regarded as a legal requirement by the lawyer. The understanding process can be seen as an abductive reasoning process since the explanation is constructed by adding implicit legal requirements. In the search of such legal requirements, the candidates of the requirements seem to be weakly constrained by legal knowledge. In Figure 5, a general principle about contract, "contract is a match of offer and acceptance" weakly constrains the candidates. Therefore, knowledge acquisition from legal texts can be seen as abduction constrained by legal knowledge.

After the verification, a new requirement, "offer is effective", is added into the body of the rule. For example, the following rule can be made.

be_concluded_contract ←
    become_effective_acceptance,
    be_effective_offer

Of course, if the rule whose effect is "acceptance become effective" is modified so that its requirement part includes "offer is effective", the same result can be derived. For example, the following rule can be made.

become_effective_acceptance ←
    reach_indication_of_assent,
    be_effective_offer

However, the former rule is preferrerd since the former rule matches their legal knowledge.

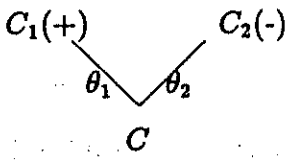
4 Inverse Resolution and Abstract Knowledge

Knowledge acquisition process can be viewed as extraction process of a legal effects and abductive explanation process of the legal effects. As an abductive explanation engine, an inverse resolution [3, 4, 5] is used. The inverse resolution is an inverse operation of resolution. Figure 6 shows typical operations of the inverse resolution. The basic operation is performed by V operator. In the resolution, C is derived from C_1 and C_2. In the inverse resolution, C_1 is derived from C and C_2 or C_2 is derived from C and C_1.

Rouveirol[4] has proposed "Saturation Operator", which is a kind of the inverse resolution operator. The alogorithm of Saturation [4] is shown in Figure 7.

Since the search space of the inverse resolution is very huge, the search space should be constraind by some legal knowledge. Such a knowledge is called a legal abstraction, since it represents abstract legal knowledge. An example of a legal abstraction shown in Figure 8. In order to represent abstract concepts, a legal abstraction is represented by Higher Order Horn Clause[1]. By assuming such legal abstraction, the legal effect can be proved by a theorem

- V Operator



- W Operator

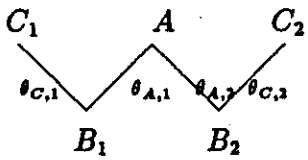


Figure 6: Inverse Resolution as Abduction[3]

Given the clause $C_e : T_e \leftarrow L_{C_e}$ and a domain theory T, proceed to steps 1,2 and 3:

1. Skolemization of C_e into $C_e\theta_s$ (one keeps track of θ_s for skolemization). For each literal $L_{C_{ei}}$ in the body of C_e, a clause C_{sk_i} is created, with $C_{sk_i} : L_{C_{ei}}\theta_s \leftarrow$.
2. Deductive phase: we apply all possible resolutions between the set of clauses $\{C_{sk_i}\}$ and the clauses of T.
3. If atoms have been deduced after step 2, they are transformed into unit clauses and added to $\{C_{sk_i}\}$. The process then iterates on step 2. If no literals have been deduced after step 2, the $\{C_{sk_i}\}$ are deskolemized and added to L_{C_e} to form the body of the saturated clause, C_s.

Figure 7: Algorithm of saturation[4]

If two indications of intention forms a legal norm sentence, the legal norm sentence is concluded when the prior indication of intention is effective and the posterior indication of intention becomes effective.

Figure 8: Example of Legal Abstraction

prover like λprolog.

5 Knowledge Acquisition Support System

KASS is a knowledge acquisition support system based on the analysis of human understanding process. Figure 9 shows the overview of KASS.

KASS has two component for knowledge acquisition. One is a bug detection module and the other is an abductive reasoning engine based on the inverse resolution. The bug detection module is based on Shapiro's algorithmic debugger[6]. A set of facts and rules are given to the module. Then the module interacts the user to identify the bug rule. This bug rule is given to the abductive reasoning engine. The reasoning engine then find an appropriate legal abstraction. Finally, a rule, which satisfy the legal abstraction, can be learned.

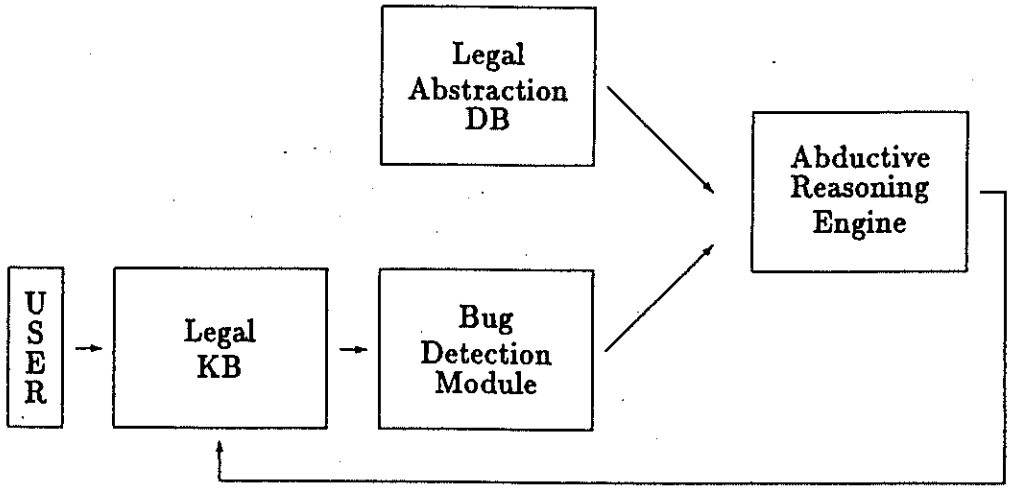


Figure 9: KASS

6 Learning Examples

The legal abstraction in Figure 8 can be represented by Higher Order Horn Clause. Figure 10 shows an example of the legal abstraction represented by a Higher Order Horn Clause.

Based on the abstraction in Figure 10, the new rule can be obtained. Figure 11 shows an example of a learned rule.

7 Conclusions

This paper describes a knowledge acquisition from legal texts based on the analysis of human knowledge acquisition process. If background knowledge is provided, we can approach the automatic knowledge acquisition from legal texts. By replacing the legal abstrac-

be_concluded(_,[agt:_,obj:Xid,tim:C]) :-
    X(Xid,[obj:_]),
    become_effective(_,[obj:Yid,tim:C])
    Y(Yid,[obj:_]),
    be_effective(_,[obj:Zid,tim:C]),
    Z(Zid,[obj:_]).
Contraints:
    X is legal norm sentence.
    Y is a indication of intention.
    Z is a indication of intention.
    Z is prior to Y.

Figure 10: Legal Abstraction Represented by a Higher Order Horn Clause

```
be_concluded(_,[agt:[A,B],
                obj:contract(_,[agt:[A,B]]),
                tim:C]) :-
  become_effective(_,
        [obj:
         acceptance(_,
             [agt:B,
              obj:offer(D,[agt:A,goa:B]),
              goa:A]),
        tim:C]),
  be_effective(_,
        [obj:offer(D,[agt:A,goa:B]),
         tim:C]).
```

Figure 11: Learning Example 1

tion database, a variety of legal knowledge can be obtained.

Since the acquired knowledge does not contain dynamically interpretation knowledge, in order to implement a legal expert sytem, other inference engine such as [7, 2] is required.

References

[1] D. Miller, et al.. Uniform proofs as a foundation for logic programming. In *Annals of Pure and Applied Logic*, 51, pp. 125-157, 1991.

[2] M. Haraguchi. A form of analogy as an abductive inference. In *ALT91*, pp. 266–274, 1991.

[3] S. Muggleton and W. Buntine. Machine invention of first-order predicates by inverting resolution. In *Workshop on Machine Learning*, pp. 339–352, 1988.

[4] C. Rouveirol. Semantic model for induction of first order theories. In *IJCAI91*. Morgan Kaufmann, 1991.

[5] C. Rouveirol and J. F. Puget. Beyond inversion of resolution. In *Workshop on Machine Learning*, pp. 122–130, 1990.

[6] E. Shapiro. *Algorithmic program debugging*. MIT Press, 1983.

[7] H. Yoshino et al. Foundation of Systematization of Legal Analogy (In Japanese). Proc. of the 5th Annual Conference of JSAI, pp. 219–222, 1991.

Legal Knowledge Acquisition Using Case Based Reasoning and Model Inference

Takahira YAMAGUTI Masaki KUREMATSU
Faculty of Enginnering Shizuoka University
3-5-1, Jouhoku, Hamamatsu, Shizuoka, 432, JAPAN e-mail:yamaguti@cs.shizuoka.ac.jp

Abstract

Although Case Based Reasoning comes out in order to solve knowledge acquisition bottleneck, a case structure acquisition bottleneck has emerged, superseding it. Because we cannot decide an appropriate case structure in advance, a framework for CBR should have the capability of improving a case structure dynamically, collecting and analyzing cases. Here is discussed a new framework for knowledge acquisition using CBR and model inference. Model Inference tries to obtain new descriptors with interaction of a domain expert, regarding the slots that compose a case structure as descriptors, with an eye to the function of theoretical term generation. The framework has two features: (1) CBR obtains a more suitable group of slots (a case structure) incrementally through cooperation with model inference, and (2) Model inference with theoretical term capability discovers the rules which deal with a given task better. Furthermore, we evaluate the feasibility of the framework by implementing it to deal with law interpretation and certify two features with the framework.

1 Introduction

In the field of knowledge engineering, research on case based reasoning has been getting active in recent years, reflecting the difficulty in building an expert model in the development of expert systems. In CBR, however, the difficulty has not been resolved so much: it has just been replaced by appropriate case structures and effective retrieval and/or repair strategies based on them. That is, case structure acquisition bottleneck has emerged, superseding the knowledge acquisition bottleneck. In the process of collecting and analyzing cases, a framework for CBR should have the capability of improving case structures dynamically.

On the other hand, in the field of model inference, there has been increasing interest in research on the identification mechanism of the first order language including the theoretical term generation issues. Theoretical term generation means the operation to take out an effective concept and give it a name. It can be regarded as the operation to obtain new descriptors to define a problem.

From the above-mentioned background, a framework for knowledge acquisition, discussed here in this paper, starts as an attempt to obtain new effective descriptors through CIGOL-based model inference by regarding the slots that compose a case structure as descriptors, with an eye to the function of theoretical term generation. The framework has two features : (1) CBR obtains a more suitable group of slots (a case structure) incrementally through cooperation with model inference, and (2) Model inference with theoretical term capability discovers the rules which deal with a given task better. Furthermore, we evaluate the feasibility of the framework by implementing it to deal with law interpretation and certify two features with the framework.

2 A Legal Knowledge Acquisition System

This section gives an outline of our framework and explains the knowledge acuisition system configurations of CBR and model inference with theoretical term capability.

2.1 System overview

Figure 1 shows the framework for cooperation between CBR and model inference with theoretical term generation. The following explanation uses the numbers assigned in the figure.

(1) With the provision of a case base for each individual civil law precedent with a different case structure, CBR stores solved cases incrementally through the retrieval, matching, modifying, evaluating, repairing, organizing and clustering.

(2) CBR gives a cluster to model inference. The cluster is a set of similar cases classified from the viewpoint of retrieval in CBR, and is converted to a set of ground clauses.

(3) With advice from an user, model inference tries to generalize the descriptor of the cluster (case description slots) and thereby invent new descriptors and compose relations among them as new clauses (rules). Consequently, some law interpretation rules can be obtained. If the model inference does not work well (for example, it can not invent a new descriptor), the input is

Legal Expert System Association (LESA)
The 6th International Symposium

LEGAL KNOWLEDGE AND LEGAL REASONING SYSTEMS

Proceedings of the Symposium

October 24-25, 1992
Meiji Gakuin University
Tokyo, Japan

Symposium Host
Legal Expert System Association
Sponsors
Ministry of Education, Science and Culture
(Grant-in-aid for Scientific Research)
Meiji Gakuin University
Institute for New Generation Computer Technology
Laboratory for International Fuzzy Engineering Research

Edited by
Hajime Yoshino