PROLEG: An Implementation of the Presupposed Ultimate Fact Theory of Japanese Civil Code by PROLOG Technology*

Ken Satoh, Kento Asai, Takamune Kogawa, Masahiro Kubota, Megumi Nakamura, Yoshiaki Nishigai, Kei Shirakawa, and Chiaki Takano

National Institute of Informatics and Sokendai
ksatoh@nii.ac.jp

Abstract. In this paper, we propose a legal reasoning system called *PROLEG* (PROlog based LEGal reasoning support system) based on the Japanese "theory of presupposed ultimate facts" (called "Yoken-jijitsu-ron" in Japanese, the JUF theory, in short). The theory is used for decision making by judges under incomplete information. Previously, we proposed a translation of the theory into logic programming. However, it turns out that the knowledge representation in logic programming is difficult for lawyers to understand. So, in this paper, we change knowledge representation of rules in the JUF theory in PROLEG so that we reflect lawyers' reasoning using the idea of "openness" proposed by a judge who is a main investigator of the JUF theory.

1 Introduction

The Japanese Presupposed Ultimate Fact Theory[3](JUF theory we call in this paper and called "*Yoken-jijitsu-ron*" in Japanese) is a decision making tool used in civil litigation. The JUF theory attaches a burden of proof[6] to each conditions in civil code in order for a judge to enable to make a judgement even if truth values of some facts are not known because of a lack of enough evidence.

Previously, we proposed a mathematical semantics of the JUF theory in [7] based on logic programming with "negation as failure". However, from the experience of giving explanation of our formalization in logic programming to lawyers, it turns out that a semantics of logic programming with "negation as failure" is difficult for lawyers to understand. Since the aim of our research is to give a support system of lawyers and a computer-aided learning system of the JUF theory for law school students and scholars who seek their interpretations of civil code, it is necessary for us to propose a system which is understandable to lawyers. Fortunately, in legal domain, Ito, who is one of the main investigator of the JUF theory as a lawyer, explains the JUF theory using *openness* of the ultimate facts[3]. He divides ultimate facts in a condition of a rule into two

* This work has been done partially while Kento Asai, Masahiro Kubota, Megumi Nakamura, Kei Shirakawa and Chiaki Takano were with NII.

categories; one category corresponds with facts which normally lead to the conclusion of a rule and the other category corresponds with facts which represent exceptional situations. Ito argues that a fact in the latter category is regarded as "open" so that the truth value is not decided until the counterpart explicitly alleges and proves the fact. Therefore, it is sufficient for a judge to make a deduction using normal rules in order to draw a conclusion when exceptional facts are not explicitly known. So, we decide to change the knowledge representation of the JUF theory into the one which simulates Ito's explanation of the JUF theory and propose *PROLEG* (PROlog based LEGal reasoning support system) in this paper. Specifically, we separate a positive condition part and a "negation as failure" part in order to familiarize a rule form with a lawyer in PROLEG.

PROLEG consists of two knowledge bases; a *rulebase* and a *factbase*. A rulebase stores rules and exceptions defined in the JUF theory and a factbase stores information about actions in the court performed by both parties and judgement of truth values by judge for ultimate facts in a given legal case (see Appendix A). Then, PROLEG automatically applies a rule or exceptions which matches true ultimate facts and gives an answer whether a conclusion of a case is derived or not together with a trace of derivation (see Appendix B).

In this sense, it is related with "rule-based legal expert systems" many of which have been developed so far[4, 8, 9]. However, we believe that none of the work incorporates burden of proof in their system. On the other hand, PROLEG is based on the JUF theory so that burden of proof is incorporated into the system and correctly handles uncertainty as shown in [7]. Moreover, while previous approaches considered more specific legal domains such as Tax law[4], British Nationality Act[8], and the United Nations Convention on Contracts for the International Sale of Goods (CISG)[9], PROLEG considered civil code in general since PROLEG is based on the JUF theory which has been developed by judges in the Japanese Legal Training and Research Institute for civil litigation in general. In addition to this generality, the JUF theory has been taught to all trainees of lawyers at the Institute for many years and proved its usefulness. Therefore, we believe that generality and usefulness of the JUF theory ensures the generality and the usefulness of PROLEG.

Moreover, PROLEG can represent actions by both parties in the court and judgements of the ultimate facts by a judge in a factbase. These actions and judgements actually corresponds with actions and judgement of civil procedure law so these representation also enhances familiarity of PROLEG to lawyers.

PROLEG outputs a trace of derivation. This trace is represented in the form of an argument between plaintiff and defendant. In this sense, PROLEG relates with argumentation systems[1, 5, 2]. As shown in[7], the JUF theory itself can be translated into a logic program with negation as failure and therefore the semantics of PROLEG may be formalized by Dung's argumentation semantics[1] since Dung's semantics was originally aimed at giving a semantics of a logic program. However, the main function of PROLEG is not to give an argumentation system, but to simulate the judge's decision process and a derivation trace is only a by-product of legal reasoning performed by a judge in the form of ar-

gument. However, there might be a possibility of using PROLEG indirectly as an argumentation system. That is, in stead of using PROLEG by a judge, a plaintiff and a defendant provide ultimate facts according to a burden of proof so an argument between a plaintiff and a defendant is constructed during the court activity. However, rules in the JUF theory is fixed so both parties can provide only ultimate facts to create an argument. Therefore, PROLEG has less freedom of making arguments compared with other argumentation system such as Carneades system [2] which has a function of making rules to produce new arguments.

2 The JUF Theory

We explain the JUF theory using the following working example.

- Suppose that a plaintiff claims that a lease contract for his house between him and the defendant ended by his cancellation of the contract[1] because the defendant let his sister use a room in the house. The plaintiff alleged that the contract was concluded [2], and his house was handed over to the defendant by the lease contract[3], and the sublease contract between the defendant and his sister was concluded [4], and the room in his house was handed over to his sister by the sublease contract[5], and she used the room or make profit by using the room[6], and the plaintiff manifested the intention of cancellation of the lease contract[7].
- In turn, the defendant alleged that
  - the plaintiff approved the sublease[8], and the approval was before the cancellation[9].
  - his subleasing a room to her does not cause any abuse of confidence with the plaintiff because the time of use was very short[10].
- In turn, the plaintiff alleged that neighbors' complaints about noise from piano lessons during subleasing abused the confidence with the plaintiff[11].

To handle the above case, we took into account the Japanese Civil Code Article 612,

[1] In this paper, this fact is represented as `cancellation_due_to_sublease`. Note that in this paper, we only considers propositional case for the sake of simplicity. But our system can handle the first-order case with variables.
[2] This fact is represented as `agreement_of_lease_contract`.
[3] This fact is represented as `handover_to_lessee`.
[4] This fact is represented as `agreement_of_sublease_contract`.
[5] This fact is represented as `handover_to_sublessee`.
[6] This fact is represented as `using_leased_thing`.
[7] This fact is represented as `manifestation_cancellation`.
[8] This fact is represented as `approval_of_sublease`.
[9] This fact is represented as `approval_before_cancellation`.
[10] This fact is represented as `fact_of_nonabuse_of_confidence`.
[11] This fact is represented as `fact_of_abuse_of_confidence`.

– Paragraph 1 states that a lessee may not assign the lessee's rights or sublease a leased thing without obtaining the approval of the lessor, and
– Paragraph 2 states that if the lessee allows any third party to make use of or take profits from a leased thing in violation of the provisions of the preceding paragraph, the lessor may cancel the contract.

However, according to the previous Supreme Court case[12], paragraph 2 is not applicable in exceptional situations where the sublease does not harm the confidence between a lessee and a lessor.

So from these rules, it firstly must be shown that the contract with the defendant was concluded and then that the contract is terminated by cancellation because of the sublease to the defendant's sister based on paragraph 2. If there were no counter-arguments, we can conclude that the lease contract is canceled. However, there are two possible counter-arguments. One counter-argument is that the plaintiff had given approval of the subleasing to the defendant according to paragraph 1, whereas the other counter-argument is that the defendant had not abused the confidence of the plaintiff according to the case rule above. Suppose that we can not conclude that the approval was given or not because of a lack of evidence. In this case, we cannot deductively prove that the counter-argument is true or not. However, the decision has to be made by the court, so we have to find a way to solve the above problem. In order to solve the problem, a judge use the idea of *burden of proof*. The idea of burden of proof is that if a party which has a burden of proof of a fact fails to prove that the fact is true, then the fact is considered to be false. In the above case, since the burden of showing the existence of approval resides in the defendant, if the defendant fails to prove that there was an approval, no approval was considered to be made.

However, in order to use this idea, we must decide to which party the burden of proof of each ultimate fact is assigned. Therefore, there has been a high demand for deciding to assign the burden of proof in each civil law code. In Japan, a group of judges has been developing just such an assignment of the burden of proof and this theory of assignment is called the "Presupposed Ultimate Fact Theory" (called "*Yoken-jijitsu-ron*" in Japanese).

3 Reflexion of Lawyers' Reasoning

3.1 Rule Format

In our previous logic programming translation[7], we explicitly introduce "negation as failure" in order to express exceptions. In the above example, the rule is formalized using "negation as failure" as follows:

```
cancellation_due_to_sublease :-
    agreement_of_lease_contract,
    handover_to_lessee,
    agreement_of_sublease_contract,
```

[12] Supreme Court Case:1966.1.27,20-1 Minsyu 136.

```
    handover_to_sublessee,
    using_leased_thing,
    manifestation_cancellation,
    not (approval_of_sublease,approval_before_cancellation),
    not fact_of_nonabuse_of_confidence.
```

In this representation, a user must understand the semantics of "negation as failure" in logic programming and it would prevent a user from understanding the behavior of the system. Especially, if "negation as failure" is nested, semantics becomes too complicated to understand. Fortunately, in legal domain, Ito, who is one of the main investigator of the JUF theory as a lawyer, explains the JUF theory using *openness* of the ultimate facts[3]. He divides ultimate facts in a condition of a rule into two categories; one category corresponds with facts which normally lead to the conclusion of a rule and the other category corresponds with facts which represent exceptional situations. Ito argues that a fact in the latter category is regarded as "open" so that the truth value is not decided until the counterpart explicitly prove the fact. Therefore, it is sufficient for a judge to make a deduction using normal rules in order to draw a conclusion when exceptional facts are not explicitly known. So, we decided to change the syntax of the above rules into new knowledge representation call *PROLEG* rules to simulate Ito's inference. That is, we separate a positive condition part and a "negation as failure" part in order to familiarize a rule form with a lawyer. The above example is represented in PROLEG as follows:

```
cancellation_due_to_sublease <=
    agreement_of_lease_contract,
    handover_to_lessee,
    agreement_of_sublease_contract,
    handover_to_sublessee,
    using_leased_thing,
    manifestation_cancellation.

exception(cancellation_due_to_sublease,get_approval_of_sublease).
exception(cancellation_due_to_sublease,nonabuse_of_confidence).
get_approval_of_sublease <=
    approval_of_sublease,approval_before_cancellation.
nonabuse_of_confidence<=
    fact_of_nonabuse_of_confidence.
```

We then introduce "`exception`" predicate which takes two arguments, the former of which is the head of default rule [13] and the latter of which is an exception of the rule. This is used for the check of existence of exceptions by the meta-interpreter.

Moreover, we introduce *intermediate concept* which aggregates some ultimate facts or intermediate concepts in order to summarize meaningful sets of these or if

[13] Let $H \Leftarrow B_1, ..., B_n$ be a PROLEG rule. We call H as a head of the rule and $B_1, ..., B_n$ a body of the rule. H, $B_1, ..., B_n$ are atoms which have the same syntax as PROLOG terms so they have the same representation power as PROLOG terms.

some concepts will be used for a higher level concept. Although the intermediate concept is not proposed in the JUF theory, we believe that it also enhances readability of knowledge representation in PROLEG.

As an example of intermediate concept which aggregates some of the conditions, we could introduce `effective_lease_contract` and a rule

```
effective_lease_contract <=
  agreement_of_lease_contract,handover_to_lessee.
```

in order to summarize how to make a lease contract effective and replace `agreement_of_lease_contract` and `handover_to_lessee` in the above rule by `effective_lease_contract`.

As an example of intermediate concept which is used for a higher level concept, "the cancellation of lease contract due to sublease without approval" can be one of causes for ceasing the lease contract[14] and there are other causes, for example, "expiring the term of lease contract"[15]. In this case, `contract_end` is a higher level concept over `cancellation_due_to_sublease` and `expiration_of_the_term_of_the_lease_contract`, so we can use these concept as an intermediate concept for `contract_end` and introduce the following rules:

```
contract_end <= cancellation_due_to_sublease.
contract_end <= expiration_of_the_term_of_the_lease_contract.
```

We show a complete rule set for the above example in the Appendix A.

3.2 Fact Handling

To check the truth of fact in conditions, we introduce four kind of predicates "`allege`", "`provide_evidence`", "`plausible`", "`admission`".

"`allege`", "`provide_evidence`", "`admission`" are actions performed by each party during argument at the court and "`plausible`" represents judgement about the fact performed by judge.

- "`allege(`F`,`P`)`" is an action of allegement of truth value of a fact F by a party P.
- "`provide_evidence(`F`,`P`)`" is an action of providing some evidence to support the truth of a fact F performed by a party P.
- "`admission(`F`,`P`)`" is an action of admission of the truth of a fact F asserted by a party P.
- "`plausible(`F`)`" means that the standard of proof for F is satisfied and F is regarded as true.

In civil litigation, each party must firstly allege a fact to the judge which the party would like to prove. Otherwise, the fact will not be considered by the judge. This is called a burden of production. `allege(`F`,`P`)` predicate is used to express such allegement. We also consider a burden of providing evidence. If either of

[14] Ceasing the lease contract is represented as "`contract_end`" in this paper.

[15] This fact is represented as "`expiration_of_the_term_of_the_lease_contract`" in this paper.

the burden of production and providing evidence is not fulfilled, the fact will not be considered by the judge. `provide_evidence(`F`,`P`)` predicate represents this action. If `allege(`F`,`P`)` and `provide_evidence(`F`,`P`)` are satisfied then the judge will consider the truth value of the fact F and if `plausible` is true then `F` is considered to be true. Moreover, in civil litigation, even if neither `allege(`F`,`P`)` nor `provide_evidence(`F`,`P`)` is performed by a party P, if the opposite party O admits the fact F, F must be considered to be true. `admission(`F`,`O`)` expresses this situation.

We show an example of such fact information for the cancellation example in the Appendix A[16].

4 Execution of PROLEG

In this section, we explain the main function of PROLEG. The function is to check whether the conclusion is proved or not according to rules and facts. However, for the sake of understanding reasoning process, PROLEG will show a trace of reasoning by adding printing function along with reasoning in the form of an argument between a plaintiff and a defendant.

The detail of the meta-interpreter can be found in Fig. 1. Given a goal G which represents a conclusion, the meta-interpreter calls prove($\{G\}$,P). Then, the meta-interpreter checks whether there is a rule whose head is unifiable with G. Then we try to prove the body of the rule. If all the B_i's are proved, we check exceptions. If there is no exception, then the execution succeeds, that is, the plaintiff wins. If there exist exceptions for the rule, that is, there is `exception`(A,E) s.t. A is unifiable with H, then we have to check whether each `exception`(A,E) fails by checking whether prove($\{E\}$,opposite(P)) fails where opposite(P) is a counter party of P. If there exists an `exception`(A,E) which succeeds then the execution fails.

In the step of derivation of G, we replace a goal by literals in the body of the rule which matches a goal until we encounter an ultimate fact. If we encounter an ultimate fact F, the meta-interpreter checks if `plausible(`F`)` or `admission(`F`,opposite(`P`))` is not in the factbase.

Note that before checking truth of a new body of the rule, we check for all the ultimate facts F to satisfy the rules, if (`allege(`F`,`P`)` and `provide_evidence(`F`,`P`)`) or `admission(`F`,opposite(`P`))` is in the factbase.

[16] In the factbase, we assume that the defendant admits all the conditions of cancellation rule due to sublease without approval, `agreement_of_lease_contract`, `handover_to_lessee`, `agreement_of_sublease_contract`, `handover_to_sublessee`, `using_leased_thing`, `manifestation_cancellation`. In turn, the defendant makes a counter-arguments of `get_approval_of_sublease` and `nonabuse_of_confidence` but fails to persuade the judge about `get_approval_of_sublease` and `approval_before_cancellation`(expressing by the non-existence of `plausible(approval_of_sublease)` and `plausible(approval_before_cancellation)` in the factbase), and in turn a plaintiff make a counter-counter-argument of `fact_of_abuse_of_confidence`.

Otherwise, the meta-interpreter does not check the satisfaction of the body. Along with this checking, applicable rules will be instantiated so the instantiated rules will be shown in a trace of reasoning and it makes the reasoning process more intuitive.

A trace of execution of cancellation of leasing contract defined in the Appendix A can be found at the Appendix B. Note that although there is a rule about `expiration_of_the_term_of_the_lease_contract` is in Appendix A, the rule is not applied since there is no allegement of the fact in the condition of the rule. Therefore, no trace about applying the rule is shown in the Appendix B.

5 Conclusion

We presented PROLEG which simulates a process of reasoning about civil code with a burden of proof based on the JUF theory. The characteristic of PROLEG are as follows.

- PROLEG reflects lawyers' reasoning about law where exceptions are ignored unless it is explicitly stated.
- PROLEG handles legal actions at the court facts such as allegement, evidence production, plausibility and admission.

We have already wrote several examples in civil code in terms of the JUF theory such as contract law and property rights and checked the correctness of behavior of PROLEG using derivation traces. As future research, we plan the following.

- We will evaluate our claim that PROLEG is more familiar with lawyers than other legal representation language.
- We will develop a legal knowledge representation which has a syntax closer to natural language to enhance readability.
- We will develop a unified method of naming predicates and deciding predicate arguments.
- We will develop a diagrammatic representation of reasoning in the JUF theory.
- We will show that the JUF theory is applicable not only for the Japanese civil code, but any other laws such as foreign civil code or criminal laws.

References

1. Dung, P. M., "On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games", Artif. Intell., Vol. 77, No. 2, pp. 321 – 358 (1995).
2. Gordon, T. F., Prakken, H., Walton, D., "The Carneades Model of Argument and Burden of Proof", Artif. Intell., Vol. 171, Nos. 10-15, pp. 875 – 896 (2007).
3. Ito, S., "Lecture Series on Ultimate Facts", Shojihomu (2008) (in Japanese).
4. McCarty, L. T., Sridharan, N. S., and Sangster, B. C., "The Implementation of TAXMAN II: An Experiment in Artificial Intelligence and Legal Reasoning," Report LRP-TR-2, Rutgers University (1979).

```
prove(S,P) goal set S; party P;
begin
  if S == ∅ return(true);
  select an atom A ∈ S;
  if A is an ultimate fact s.t. plausible(F) or admission(F,opposite(P))
    is in a factbase and F is unifiable with A then
  begin
    S := S − {A}; return(prove(S,P))
  end
  else if A is an intermediate concept then
  begin
    select a rule H ⇐ B1, ..., Bn
      whose head matches A with H by most general unifier θ;
    if such a rule does not exist then return(false);
    select δ s.t. alleged_and_having_evidence((B1, ..., Bn)θ,P) returns δ;
    S := (S − {A} ∪ {B1, ..., Bn})θδ;
    if prove(S,P)==true then
    begin
      for every exception(G,E) s.t. Aθδ is unified with G by most general unifier η
        if prove({Eθδη},opposite(P))==true then return(false)
      return(true)
    end
  end
  else return(false);
end

alleged_and_having_evidence(S,P) goal set S; party P;
begin
  if S == ∅ return(ε); /* ε is an empty substitution. */
  select an atom A ∈ S;
  if A is an ultimate fact s.t.
      (allege(F,P) and provide_evidence(F,P)) or admission(F,opposite(P))
      is in a factbase and F is unifiable with A by most general unifier θ then
  begin
    S := S − {A};
    δ = alleged_and_having_evidence(S,P);
    return(θδ)
  end
  else if A is an intermediate concept then
  begin
    select a rule H ⇐ B1, ..., Bn
      whose head matches A with H by most general unifier θ;
    if such a rule does not exist then return(false);
    S := (S − {A} ∪ {B1, ..., Bn})θ;
    δ = alleged_and_having_evidence(S,P);
    return(θδ)
  end
end
```

Fig. 1. Algorithm of PROLEG Meta-Interpreter

5. Prakken, H., "Logical Tools for Modelling Legal Argument: A Study of Defeasible Reasoning in Law", Law and Philosophy Library, Vol. 32, Kluwer Academic (1997).
6. Prakken, H., Sartor, G., "Formalising Arguments about the Burden of Persuasion", Proc. of ICAIL 2007, pp. 97 – 106 (2007).
7. Satoh, K., Kubota, M., Nishigai, Y., Takano, C., "Translating the Japanese Presupposed Ultimate Fact Theory into Logic Programming", Proc. of JURIX 2009, pp.162-171 (2009).
8. Sergot, M.J., Sadri, F., Kowalski, R.A., Kriwaczek, F., Hammond, P., and Cory, H.T., "The British Nationality Act as a Logic Program", CACM, 29(5):370–386 (1986).
9. Yoshino, H., "On the Logical Foundations of Compound Predicate Formulae for Legal Knowledge Representation", Vol. 5, No.1-2, pp. 77 – 96 (1997).

Appendix A: Example of PROLEG program

```
%============================ PROLEG rulebase ============================
contract_end <= cancellation_due_to_sublease.
contract_end <= expiration_of_the_term_of_the_lease_contract.

cancellation_due_to_sublease<=
      agreement_of_lease_contract, handover_to_lessee,
      agreement_of_sublease_contract, handover_to_sublessee,
      using_leased_thing,
      manifestation_cancellation.

exception(cancellation_due_to_sublease,get_approval_of_sublease).
exception(cancellation_due_to_sublease,nonabuse_of_confidence).

get_approval_of_sublease <=
      approval_of_sublease,
      approval_before_cancellation.

nonabuse_of_confidence <= fact_of_nonabuse_of_confidence.

exception(nonabuse_of_confidence,abuse_of_confidence).
abuse_of_confidence <= fact_of_abuse_of_confidence.

expiration_of_the_term_of_the_lease_contract<=
          end_of_the_term_of_the_lease_contract,
          notice_of_renewal_refusal_between_12month_and_6month,
          justifiable_reason.
% ========================= PROLEG factbase ==========================
admission(agreement_of_lease_contract,defendant).
admission(handover_to_lessee,defendant).
admission(agreement_of_sublease_contract,defendant).
admission(handover_to_sublessee,defendant).
admission(using_leased_thing,defendant).
```

```
admission(manifestation_cancellation,defendant).

allege(approval_of_sublease,defendant).
provide_evidence(approval_of_sublease,defendant).

allege(approval_before_cancellation,defendant).
provide_evidence(approval_before_cancellation,defendant).

allege(fact_of_nonabuse_of_confidence,defendant).
provide_evidence(fact_of_nonabuse_of_confidence,defendant).
plausible(fact_of_nonabuse_of_confidence).

allege(fact_of_abuse_of_confidence,plaintiff).
provide_evidence(fact_of_abuse_of_confidence,plaintiff).
plausible(fact_of_abuse_of_confidence).
```

Appendix B: Example of PROLEG Execution Trace

```
 1  plaintiff tried to prove "contract_end".
 2  To prove "contract_end",
 3    we need to prove the following requisites:
 4
 5      requisite1: cancellation_due_to_sublease
 6
 7    plaintiff tried to prove "cancellation_due_to_sublease".
 8    To prove "cancellation_due_to_sublease",
 9      we need to prove the following requisites:
10
11        requisite1: agreement_of_lease_contract
12        requisite2: handover_to_lessee
13        requisite3: agreement_of_sublease_contract
14        requisite4: handover_to_sublessee
15        requisite5: using_leased_thing
16        requisite6: manifestation_cancellation
17
18      defendant admitted "agreement_of_lease_contract".
19      defendant admitted "handover_to_lessee".
20      defendant admitted "agreement_of_sublease_contract".
21      defendant admitted "handover_to_sublessee".
22      defendant admitted "using_leased_thing".
23      defendant admitted "manifestation_cancellation".
24      defendant alleges "get_approval_of_sublease"
25        as a defense against "cancellation_due_to_sublease".
26        defendant tried to prove "get_approval_of_sublease".
27        To prove "get_approval_of_sublease",
```

```
28          we need to prove the following requisites:
29
30            requisite1: approval_of_sublease
31            requisite2: approval_before_cancellation
32
33          defendant tried to prove "approval_of_sublease".
34          defendant failed to prove "approval_of_sublease".
35        defendant failed to prove "get_approval_of_sublease".
36      defendant failed to prove
37        "get_approval_of_sublease" as a defense
38          against "cancellation_due_to_sublease".
39      defendant alleges "nonabuse_of_confidence"
40        as a defense against "cancellation_due_to_sublease".
41        defendant tried to prove "nonabuse_of_confidence".
42        To prove "nonabuse_of_confidence",
43          we need to prove the following requisites:
44
45            requisite1: fact_of_nonabuse_of_confidence
46
47          defendant tried to prove "fact_of_nonabuse_of_confidence".
48            "fact_of_nonabuse_of_confidence" is determined to be plausible.
49          defendant successfully proved "fact_of_nonabuse_of_confidence".
50          plaintiff alleges "abuse_of_confidence"
51            as a defense against "nonabuse_of_confidence".
52            plaintiff tried to prove "abuse_of_confidence".
53            To prove "abuse_of_confidence",
54              we need to prove the following requisites:
55
56                requisite1: fact_of_abuse_of_confidence
57
58              plaintiff tried to prove "fact_of_abuse_of_confidence".
59                "fact_of_abuse_of_confidence" is determined to be plausible.
60              plaintiff successfully proved "fact_of_abuse_of_confidence".
61            plaintiff successfully proved "abuse_of_confidence".
62          plaintiff successfully proved "abuse_of_confidence"
63          as a defense against "nonabuse_of_confidence".
64        defendant failed to prove "nonabuse_of_confidence".
65      defendant failed to prove
66        "nonabuse_of_confidence" as a defense
67          against "cancellation_due_to_sublease".
68    plaintiff successfully proved "cancellation_due_to_sublease".
69  plaintiff successfully proved "contract_end".
```