My Visit to ICOT
February 22 to March 4, 1994

Dr. Thomas F. Gordon

1 Overview

Dr. Nitta kindly invited me to visit ICOT for two weeks, principally in order to exchange results in the area of Artificial Intelligence and Law, and to help him with the design of the new HELIC-II language and system, which is to be used in a system for legal reasoning and argumentation. However, as this was my first visit to ICOT, I was also given the opportunity to speak with persons involved in several other ICOT projects, including Dr. Chikayama (KL1, MGTP), Mr. Tsuda and Mrs. Takahashi (Quixote). In Dr. Nitta's legal reasoning group, I also spoke at length with Mr. Oosaki about the EDR concept dictionary, with Mr. Sakata about his model of legal debate, and with Mr. Tojo about situation theory and temporal reasoning.

On Friday afternoon, Feb. 25, I gave a talk and demonstration about my own Artificial Intelligence model of legal argumentation, called the Pleadings Game, which was the subject of my doctoral dissertation. The game may be viewed as a resource-limited, nonmonotonic dialgue logic, which supports practical decision-making. It can also be considered to be formalization of Toulmin's philosophy of argumentation. The Pleadings Game uses Geffner and Pearl's nonmonotonic logic, called Conditional Entailment. The Game includes a rule language with the expressiveness of a higher-order logic. In addition to rules about the legal domain, such as commercial law, the langauge allows various kinds of explicit preference relationships between rules to be expressed, such as Lex Superior (e.g. federal law has priority over state law), Lex Posterior (newer laws have priority over older laws). These priority rules may themselves be ordered, using the same language. For example, one may state that Lex Superior has priority over Lex Posterior. The Pleadings Game is not a theorem prover for this logic; rather it is a kind of "mediating system" which monitors and regulates a debate between two parties. In theorem proving terms, it is a kind of "proof-checker" for a two-party argumentation game. The rules of the game are derived from Robert Alexy's theory of legal argumentation. Rules such as "a party may not contradict himself", "a party shall be held to believe the necessary consequences of his claims", and "a party may only make arguments which are relevant to the issue being debated" are implemented using a production rule language. The theory of issues and relevance used in this last rule is an extension of previous work of mine, in which issues are defined using abduction. The Pleadings Game is implemented in Standard ML, a strongly typed functional programming language.

Prof. Yoshino, the leading authority on Expert Systems in Law in Japan, also attended this talk, and the following week, on March 2, I was given the opportunity to visit Prof. Yoshino's institute at Meiji Gakuin University. Prof. Yoshino described his new joint project on Expert Systems in Law, which is funded by the Japanese Ministry for Science, Education and Culture. The project started in May, 1993, and will continue until March, 1998. Prof. Yoshino also demonstrated the current version of his LES system for legal reasoning, which is implemented in ESP and runs on a PSI machine.

In the next section of this report, I would like to make some comments and suggestions about some of the work in progress in the legal reasoning group here at ICOT. I have been very impressed with the work of the legal reasoning group. These comments are intended to be constructive and helpful, and I hope that my colleagues at ICOT will find them useful.

2 Comments and Suggestions

In this section, I will make a few observations and recommendations concerning some of the issues facing the legal reasoning project at ICOT. When discussing some item here, such as Quixote, I will not try to make a general review, but will instead restrict my remarks to issues relevant for the legal reasoning project.

All of these points have already been made in discussions with the persons involved here at ICOT. But this report may be helpful as a record summarizing our conversations.

2.1 Quixote

In the first half of the 1980s, I designed two object-oriented logic programming languages, OBLOG and OBLOG2, specifically with legal reasoning applications in mind. So I was very interested in learning more about Quixote, which is a modern and considerably more advanced object-oriented logic programming language. Also, there is something of a tradition in the field of AI and Law to write knowledge bases in this style of logic programming. Besides my own work, prominent examples include Prof. Yoshino's LES system and Prof. L. Thorne McCarty's Langauge for Legal Discourse (LLD). Also, Quixote itself has been used at ICOT to build an experimental legal application, TRIAL, which further increased my interest to become acquainted with the language.

However, Dr. Nitta's legal reasoning group, as you know, has decided not to use Quixote, but to develop their own new language, which does not yet have a name. (Following Dr. Nitta, I will call it NL in this report.) So, the first issue I would like to discuss here is whether or not they should instead just use Quixote. After all, there are still many tasks to accomplish in a very short period of time, and implementations of Quixote already exist.

To evaluate Quixote, I began to try to reimplement some parts of my Uniform Commercial Code model, which was developed for my Pleadings Game. The language I used there is an extension of Prof. McCarty's Clausal Intuitionistic Logic (CIL), which is Horn clause logic extended with embedded implications and intuitionistic negation. That is, CIL has a form of monotonic negation, unlike negation as failure, but the logic is weaker than a logic with classical negation. If you prefer, a theorem prover for CIL may be be viewed as a correct but incomplete prover for classical first-order logic. Unlike Quixote, CIL does not however have a type discipline, and doesn't support well an object-oriented style of knowledge representation. To imitate a object-oriented style, I adopted the common convention of using unary predicates for concepts, and binary predicates for attributes. For example, using a Prolog-like syntax:

```
security_agreement(s1).
secured_party(s1,p1).
```

means that `s1` is a instance of the `security_agreement` concept, and a secured party of `s1` is p1.

To define a concept, I used the ancient method known as "genus and differentia", which is also commonly used in, e.g., legal statutes. For example, in the Uniform Commercial Code, an instrument is defined to be a writing which evidences a right to payment of money and is itself not a security agreement. This may be represented in CIL by the following two rules:

```
writing(P) <- instrument(P).
instrument(P) <-
    writing(P),
    evidence(P,R),
    right_to_payment(R),
    not security_agreement(P).
```

Notice that a depth-first search strategy would cause the query `writing(a)` to loop and fail to find a solution. This is one reason my implementation of CIL uses iterative deepening instead. However, one may also consider using forwards reasoning.

When I tried to represent this definition of an instrument in Quixote, I was surprised to discover that it is not possible. Here is my first attempt:

```
&program;;
&subsumption;;
    writing >= instrument;;
    writing >= security_agreement;;
&rule;;
    instrument[name=P] <=
    writing[name=P]/[evidence=right_to_payment[name=R]],
    ...
&end.
```

There are two problems with this attempt, which are probably obvious to experienced Quixote programmers. The first is that an object named `writing[name=a]` is not the same object as one named `instrument[name=a]`. Although the rule above can be used to derive such an instrument object, it is another object, not the same as the writing evidencing a right to payment. These two objects or terms are not unifiable in Quixote, so for example, if there are two facts:

```
writing[name=p]/[evidence=right_to_payment[name=r]];;
joe/[possesses=writing[name=p]];;
```

the query `?- joe/[possesses=instrument[name=X]]` will unfortunately fail, even if, as I assume here, the other conditions defining instruments are satisfied by `writing[name=p]`.

This is quite a serious limitation of Quixote. Mrs. Takahashi was kind enough to show me a partial solution to this problem, using a `kind` attribute:

```
&program;;
&subsumption;;
    writing >= instrument;;
    writing >= security_agreement;;
&rule;;
    writing[name=P]/[kind=instrument] <=
        writing[name=P]/[evidence=right_to_payment[name=R]],
        ...
&end.
```

Although this solution may be adequate for some domains, it is not a general solution. Presumably, one would also like to be able to define the concept of a writing using the technique of *genus* and *differentia*, as well as all other concepts up to the root of the concept taxonomy, `&top`. Thus, this approach would require *all* objects to be named like `&top[name=a]`, `&top[name=b]` and so on. Such Quixote programs would be rather verbous, inelegant and somewhat more difficult to read.

The second problem encountered is suggest by the ellipsis, "...", in my first attempt to define instruments in Quixote, above. Quixote does not currently support any form of negation. I recognize that this is a difficult issue, so I do not want to suggest any "quick and dirty" solutions. However, the Quixote group may want to take a look at Lambda Prolog and McCarty's Clausal Intuitionistic

Logic for ideas about extending Quixote with a form of (monotonic) negation. Negation as failure would not be sufficient, as it makes the logic nonmonotonic. My position regaring negation as failure is that it is not sufficient for realizing defeasibility. For example, it does not make it possible to represent more than one layer of exceptions, whereas in legal codes it is quite common to have exceptions to exceptions, and so on. On the other hand, *some* form of negation would appear to be required, in order to realize the notion of a counterargument, i.e. an argument which counters another argument. Finally, legal statutes, such as in the example above about instruments, often use the concept of negation. As for Quixote, perhaps inequality constraints would be offer at least some relief, such as:

```
person[name=sam]\[father != person[name=joe]];;
```

which is intended to mean that Sam's father is not Joe.

To conclude this section on Quixote, I must say that I have come to agree with Dr. Nitta's assessment that it is not suitable for legal knowledge representation, for the reasons shown above. The problem with negation is the least serious of the two problems discussed, as it would, presumably, not be too difficult to extend Quixote with intuitionistic negation. But the problem with Quixote's type system appears to be more fundamental. The type system requires the programmer to declare or assert every object at the appropriate location in the subsumption taxonomy. An object cannot be reclassified automatically as more information becomes known about its attributes. This restriction may be acceptable when writing logic programs, as one usually has complete information about the constants used in the program; but it is not tolerable in a knowledge representation language, where one usually has only partial information about the attributes and properties of objects.

2.2 Concept Dictionary (EDR)

Mr. Oosaki discussed the cooperation of the legal reasoning project with the EDR company, which is constructing a large concept dictionary for English and Japanese. EDR appears to be using a varient of John Sowa's conceptual graph theory for defining the concepts of the dictionary. In one of the papers I was given about EDR, they claim to have already represented about 400,000 concepts using this language, which is a very impressive accomplishment by any standard.

Initially, the EDR project reminded me in scope and ambition of Doug Lenat's CYC project, which is being conducted at MCC in the U.S. Although the goals of CYC have been modified with time, as is usual and sensible for long-term projects, its initial goal of developing one very large scale knowledge base for all of common sense was, in my opinion, naive and unachievable. For lawyers, the reason is obvious: natural language concepts are "open-textured"; their meaning continually changes with use. For example, the meaning of the

the terms used in legislation is interpreted and reinterpreted by the courts while deciding cases. To exacerbate this situtation further, reasonable people can and do disagree about the meaning of such concepts. The CYC project itself came to realize this, which is why the CYC system now allows there to be conflicting theories of the meaning of each concept.

Perhaps the EDR project has a more limited aim than CYC had originally. It may be that a concept *dictionary* is not the same thing as a knowledge base for common sense. There may be considerably less disagreement about the subsumption relationship between concepts than about their proper definitions. For example, perhaps everyone can agree that cars are vehicles, without having to precisely define either concept.

Nonetheless, in legal reasoning, the decision of a case often turns exactly on submission questions of this kind. For example, the most significant issue in a secured transactions case may be whether or not a "ship" is a "good" in the sense intended by Article Nine of the Uniform Commercial Code. One aspect of the problem here is that terms like "good" do not have their generic, dictionary meaning when used by lawyers, but are "technical" terms of art. This is true, of course, for all disciplines. Surely a "bug" means something else to computer scientist than to other people.

Thus, I am quite sympathetic to the decision of the legal reasoning group here at ICOT not to use the EDR concept dictionary uncritically. The EDR dictionary can be useful as a model of how to represent some concepts and the EDR language might be able to provide some ideas for the NL language, but when creating a legal argument using NL a legal expert will have to be free to reconsider the appropriateness of the EDR definitions.

2.3 New Helic-II Language

The legal reasoning group here at ICOT is still desiging the new Helic-II language (NL), so I hope that my comments in this section will be constructive and helpful during this phase of the project. It may not be too late to make some changes in the design.

According to the working paper on the language design I was given, five goals have been identified for NL:

1. The language should distinguish types and predicates.

2. The language should have the expressiveness of a higher-order logic.

3. The language should have two kinds of negation, including negation as failure.

4. The language should support defeasible reasoning.

5. The language should provide support for reasoning by analogy.

Let me discuss each of these points in turn, except the last point. I can agree that support for reasoning by analogy is important, but I am afraid I do not have the competence in this area to provide any useful suggestions.

Regarding the first point, I think the way types and predicates are distinquised in Quixote could serve as a model for NL. I don't see why NL should deviate from Quixote in this regard. In Quixote, at least micro-Quixote, extrinsic attributes have the semantics of binary predicates. There is no theoretical need to support more than unary and binary predicates, as even Nilsson's old text book on Artificial Intelligence demonstrates. In Quixote, the role of unary predicates is played by the type taxonomy. It is sometimes initially convenient to have predicates with an arity greater than two, but their use usually makes it considerably more difficult to modify and maintain the knowledge base when one decides that the relation should have an additional parameter.

This does not mean, of course, that NL should adopt Quixote's type system, which as has already been discussed is inappropriate for legal knowledge representation. However, the type system of both Quixote and the current design for NL are based on the same theory of types, Smolka's F-Logic, so this choice should be reconsidered. You should take a look at recent concept logics such AT&T's *Classic* system, which is discussed in a recent AI Magazine article, and Franz Baader's system, which if I recall correctly is named SB-ONE. Perhaps it would be best to scan one of the last AAAI and ICAIL conference proceedings. I believe they both had sessions on concept logics, which are also called term logics.

As for the second point, I agree it would be very useful to have some of the expressiveness of higher-order logic in NL. You already have much experience in this area at ICOT. Both the old Helic-II and Quixote allow one to express what appears to be higher-order relationships without actually leaving first-order logic. However, it may be that this "syntactic trick" is not sufficient for legal knowledge representation, as one often needs to write rules which refer to other rules. Modules in the present draft for NL, which are also presumably borrowed from Quixote, are designed for this purpose; but as I mentioned above in the section on situation theory it is important to be able to define modules using the rule language, at a meta-level. The current design module language does not have this degree of expressiveness.

Regarding the plan to have two kinds of negation, I have already commented about this issue in this report. Although others, such as Prof. McCarty disagree, it is my position that there should not be **both** negation as failure **and** some other system for defeasible reasoning in one knowledge representation language. But I agree strongly that some monotonic form of negation is required. Again, you may want to look at Lambda Prolog and McCarty's Clausal Intuitionistic Logic for fruitful ideas.

As for your goal to support defeasible reasoning, I have long been an advocate of nonmonotonic logics for legal reasoning, as you know. (See for example my Tübingen paper called "The Importance of Nonmonotonicity for Legal Reasoning." The full cite in in my disseration. Also, the main arguments of that paper are repeated in the dissertation. This is a controversial issue in the AI and Law community. Many famous legal logicians, as you know, claim that classical first-order logic is sufficient for all purposes. So I am happy to see that you and I belong to the same camp on this issue.

The real question, of course, is just *how* to realize defeasible reasoning. It is my position that *none* of the current crop of nonmonotonic logics are suitable, either because they do not allow defeasible rules to be expressed in a form which resembles the way statutes are written (c.f. isomorphic modelling), or because they do not allow one to write rules within logic about the priority of other rules. That is, some form of logic for higher-order defeasible reasoning is required. Also, nonmonotonic logics, such as conditional entailment, which resolve conflicts using only the principle of specificity (called Lex Specialis by lawyers), or in which Lex Specialis prevails over all other such principles for order rules, are not satisfactory. As every lawyer knows, for example, Lex Superior prevails over Lex Specialis. (For example, a general federal law takes precedence over a more specific state law.)

I'm afraid I can't be of much more help regarding this issue at the moment. It is a very difficult problem. I can only say that there are, or should be, interactions between whatever language you design for expressing defeasible rules and the discourse norms of your debate system. These are not orthogonal issues, but need to be designed together. The debate system is discussed in the next and final section of this report.

2.4 Legal Debate

I do not want to take the space to describe your current draft design of the debate system for legal argumentation, but will instead assume that the reader is familiar with it. Rather, I will instead focus on the following issues:

1. What should be the function of the debate system?

2. What kinds of issues should be debatable?

3. What is an issue?

4. What is an argument?

5. How shall a decision be made when there are competing arguments?

6. When should a party be committed to an argument or claim?

Regarding the first issue, the current intention is for the debating system to *both* assist the parties in finding and making arguments and to mediate the debate or discussion so as to assure that the parties do not violate any of the procedural rules governing the discussion. In my view, these two functions should be cleanly separated, as your current approach entails a conflict of interests. The mediator in a legal case is an agent of the court and state, and should protect their interests. The system which assists the parties in making arguments, however, has the role of an attorney, it attempts to maximize the interests of its client. This does not mean that these two kinds of systems cannot have components in common; both may be constructed using some of the same library modules. But they should not have access to the same information. The attorney for a party has an obligation to keep information about his client confidential, for example.

This brings us to the second point, about what kinds of issues should be debatable. In the current design, it is assumed that there is no disagreement about the proper interpretation or evaluation of the case books, code books (i.e. statutory law) or evidence regarding the facts of the case. These are rather strong assumptions to be making in a system for supporting debate about the proper resolution of a legal conflict! The only kind of issue currently supported in your design is the proper weighing or ordering of conflicting legal rules. Even your support for case-based reasoning is viewed this way, as a question of which of the "case-rules" representing possible interpretations of a case to prefer. I hope my Pleadings Game may provide some ideas about how to mediate a legal debate in which all of these other kinds of issues can be discussed.

The third question concerns your definition of an "issue". As I understand your design, a proposition p is an issue if and only if there is an argument for p and an argument for `not p`. In my theory of issues, as presented in my dissertation, the concept of an issue has two aspects: 1) the proposition must be *relevant* for resolving the "main issue" of the case, and 2) the parties must disagree about the proposition. It does not appear that either of these factors play a role in your current design. Also, it is important to distinguish between denying the claim of a proposition p and claiming `not p`. In my system, the parties disagree about the proposition if one party claims the proposition and the other party denies this claim. To make an issue out of the claim, it is not necesssary to claim its complement. This distinction is important, so as to prove the distribution of the burden of proof amoung the parties.

Next, what is an argument? You take the view, following Pollock, Loui, Sartor and some others, that an argument is a proof tree for a proposition. In the Pleadings Game, I take the view that an argument is a set of formulas together with an effective procedure for deciding whether or not this set of formulas shall be held to support some other formula. The party wanting to assert the argument can apply the procedure to be sure that his statements are in a form which allows the procedure to succeed in finding a proof of the formula he wishes to support. My view may be a bit more general, but I think that both of these interpretations of the argument concept are satisfactory. The main intuition behind the idea of an argument, I claim, is that anyone should be able to appreciate the force of the argument without having to first solve a difficult problem, such as searching for a proof using an undecidable logic.

The next question is how arguments should be compared so as to decide issues. You adopt, and perhaps adapt, Sartor's approach, whereas I used conditional entailment. As you know, I no longer think conditional entailment is suitable for legal argumentation, because it gives the principle of Lex Specialis priority over all other such principles. However, I do not think Sartor's approach is the final answer to this problem. As Jaap Hage and his collegues have pointed out, we sometimes want to weigh all of the arguments which speak for an against some thesis. In such cases, several weak arguments can be used together to defeat an argument which is stronger than each of them individually. This is not possible using Sartor's approach. For a realistic example of where this kind of weighing is used in legal reasoning, consider the weighing of evidence at trial. This said, perhaps it is too difficult to find a general solution to this problem in the time left for your project. It may be best to continue with your plan to use Sartor's approach, and just recognize that it has its limitations.

The final point I want to discuss here concern's the commitments of a party. In your current design, you permit a party to retract claims and to shift his "view". Is there any limit to the number of times a party may do this? You need to be careful that it is not possible for a party to prevent the proceeding from terminating. (In the U.S. Congress, a party may try to prevent a bill from being voted on by continuing to hold speeches until so many congressmen — and women – have gone home that there is no longer permissable to conduct the voting. This tactic is called a filibuster.) In my system, I did not permit a party from retracting his prior claims, and only one argument or counterargument was allowed for each open statement. The party has the burden of making his best possible argument at each choice point. This approach is admittedly rather draconian; but it is surely possible to find some reasonable comprimise. Your current design is too tolerant.

2.5 Situation Theory

Situation theory is new to me, so my comments here should be evaluated accordingly. From my discussion with Mr. Tojo, it appears to allow a form of higher-order reasoning, where rules of the following form can be expressed:

```
s0:p0 <= s1:p1, s2:p2, ..., sn:pn
```

This may be viewed as a higher-order Horn clause, where each `si:oi` is intended to mean that the proposition `oi` is supported by, or derivable from, the situtation `si`, where a situtation is a set of such rules.

Now, the claim is that situation theory is useful for formalizing legal reasoning, because to construct arguments for a particular case lawyers must reason not with just one theory, but with a combination of theories for different legal codes, such as federal and state statutes, or statues for different legal fields, such as contract law and tort law, as well as theories for the prior cases and theories for different interpretations or views of the law and facts of the current case. Moreover, cases and statutes usually refer to other cases and statutes. For example, Section 9-113 of Article 9 of the Uniform Commercial Code (UCC) says:

> A security interest arising solely under the Article on Sales (Article 2) is subject to the provisions of this Artile except so long as the debtor does not have possession of the goods ...

That is, a section of Article 9 refers explictly to Article 2. Similarly, often once section will refer to other sections of the same article. For example, Section 9-203 states:

> Subject to the provisions of Section 9-113, a security interest is not enforceable against the debtor or third parties ...

The idea, then, would be to represent this structure of the law using situtations. One could have one situation per section of an article, and an article could be defined as the union of its sections, and so on. Moreover, as a rule in a situation may refer to other situations, one rule could refer to other rules, as in the above examples.

Thus, it may well be that situation theory will be useful for formalizing these kinds of rules, and the kind of reasoning that is necessary in order to use them. This research direction is interesting and should be persued further, in my opinion.

However, one caveat is that it is probably insufficient, or at least very inconvenient, to be able to define situations only using set operations, such as set union and set intersection. I see no reason why situation theory would require situations to be defined in only this way, but from discussions with Dr. Nitta and Mr. Sakata it appears that they have been limiting their design of NL to just these kinds of situtation constructors. (In NL, situations are called "units".)

Rather, it would be much better if situtations could be defined implicitly using the same rule language to be used for defining legal concepts of some legal domain, such as contract law. For example, one should be able to define the situation for the set of all of sections of the Uniform Commercial Code enacted in California between 1973 and 1977 with a rule like:

```
situation1(R1) <-
    ucc(R1),
    enactment_year(R1, Y),
    Y > 1973,
    Y < 1977.
```

Here I'm using a notation for Horn clause logic, but this kind of definition of a situation is also be possible in whatever notation is chosen for representing rules in situation theory.

It was proposed that Quixote could be used as a theorem prover for situation theory, as Quixote's modules can be used to represent situations. Modules in Quixote cannot be desribed using Quixote rules, but only using set operations. So, although Quixote may well be viewed in the abstract as a theorem prover for situation theory, it is unfortunately poorly suited for modeling the kinds of situations which are common in legal reasoning.

2.6 Temporal Logic

Mr. Tojo also spent some time with me discussing his work on temporal reasoning using situtation theory. It is indeed true that some way for representing and reasoning about time is often, perhaps even typically required when modelling legal domains. This was also the case in my model of Article Nine of the Uniform Commerical Code, where the priority of a security interest can depend on the order of events, such as which secured party first took steps sufficient to "perfect" his interest in the collateral. In my model, I used a version Kowalski and Sergot's event calculus for this purpose. I am not an expert on temporal logics, but I have not yet found a reason to be dissatisfied with the event calculus.

In our discussion, at least, Mr. Tojo focussed primarily on limitations of Allen's interval-based theory of the relationship between events, compared to Vendler's work in linguistics on the temporal characteristics of verbs. In Vendler's theory, an event not only has a starting and ending point, but also a *culmination point*, at which time the action finishes but its effects continue until the ending point of the interval of the event.

Not being an expert on temporal logic, I don't have much to say about this work, except for the following. I am not sure about the relevance of work in the field of linguistics for modeling legal argumentation or building systems which assist in the construction or evaluation of legal arguments, unless one has the ambition to build a natural language understanding system for extracting arguments and interpretations from such legal documents as statutes and cases. So long as a "knowledge engineer" is responsible for interpreting these documents to construct models using a knowledge representation language, the features of this language for representing time may be kept relatively simple.

Also, and perhaps more to the point, merely distinguishing between the culimination point and end point of an event is not sufficient for solving the *frame problem*. Usually, one does not want to state an explicit end point for the effects of an action. Rather, one would like to presume that the effects persist until something else happens to terminate them. The passage of a fixed period of time is just one very simple kind of terminating event. Much work has been done in the AI field of task planning to solve the frame problem, and there are now numerous approaches to consider. Kowalski and Sergot's event calculus, which I favor when using a first-order logic to represent legal knowledge, is one such approach. They deal explictly with this problem in their work.

To sum up, it is my suspicion that the event calculus would be a more suitable candidate for representing and reasoning about time (and action) than Vendler's theory for legal applications.

3 Concluding Remarks

Let me take this opportunity to thank Dr. Nitta again for inviting me to visit ICOT. ICOT certainly has one of the largest and most competent research groups in the world in the area of artificial intelligence and law, and I am very glad to have had a chance to get to know more about the project and its members. It really is sad that ICOT will be closed in about a year's time. I sincerely hope that the knowledge and expertise you have acquired in the area of AI and Law will continue to be maintained and nurtured by the Japanese government, industry and universities in some substantial way.

I would like to thank Mrs. Kumiko Karawaka and Mr. Kazuo Narita for taking care of my travel and hotel arrangements. This was my first trip to Japan, so I was a little apprehensive about how I would find my way about and be able to manage the language barrier. But with their help my visit here was quite pleasant and I didn't encounter any difficulties. Finally, let me thank Mr. Hiroshi Oosaki for taking his time to escort me around Tokyo on the weekend. I would have been lost without him.

Resume

Thomas F. Gordon
September 14, 1993

1 Education

- University of California, San Diego; B.A., Psychology; June, 1977.
- University of California, Davis; King Hall School of Law; Juris Doctor (J.D.); June, 1982.
- Technischen Hochschule Darmstadt; Dr. rer. nat; Computer Science; June, 1993.

2 Employment History

- German National Research Center for Computer Science (GMD); Research Scientist; April, 1983 to present.

3 Professional Service

- Program Committee; International Conference on AI and Law (ICAIL); Oxford, 1991.
- Program Committee; International Conference on AI and Law (ICAIL); Amsterdam, 1993.
- Editorial Board; Artificial Intelligence and Law; Kluwer Academic Publishers.
- Executive Committee; International Association for Artificial Intelligence and Law (IAAIL).
- Vice Chairman; Special Interest Group 6.1.2;