Emerging Computer-Assisted Legal Analysis Systems

Mark Morrise
Emerging Computer-Assisted Legal Analysis Systems

I. INTRODUCTION

The computer is becoming an increasingly useful tool for the legal profession.¹ Law firms currently use computer systems for word processing,² legal research,³ litigation support,⁴ and office accounting and billing.⁵ Of these systems, only computerized legal research has provided the attorney with any assistance in legal analysis.⁶ However, during the 1970's work began on several systems specifically designed for legal analysis.⁷ These new

⁴ Computerized litigation support systems allow an attorney to store a large number of evidentiary documents in the computer, to review them on a video terminal by document type or by certain “key words” in the document, and to select documents to be printed and introduced at trial. See generally Arthur, The Computer and the Practice of Law: Litigation Support, 63 A.B.A.J. 1737 (1977); Sherman & Kinnard, The Development, Discovery, and Use of Computer Support Systems in Achieving Efficiency in Litigation, 79 COLUM. L. REV. 287 (1979).
⁶ Computerized legal research systems aid the attorney in searching out the law but not in applying it to his case.
⁷ This Comment will examine four of these systems: JUDITH, the ABF processor, TAXMAN, and Jeffrey Meldman's system. A description of one other system that assists in legal analysis has been written. See R. McCoy, Improving Legal Service Delivery with Computer Technology (1976) (doctoral thesis at University of Wisconsin-Madison) (Available from University Microfilms International, Ann Arbor, Michigan). There are at least three other systems about which nothing has yet been written. They are being developed at the J. Reuben Clark Law School, Brigham Young University, Provo, Utah; at the Lewis and Clark Law School, Portland, Oregon; and at Legal Management Systems, Ann Arbor, Michigan.
systems were to aid the attorney in drafting documents, finding an appropriate cause of action, and determining whether the elements of a legal doctrine or statute are satisfied. Although these new systems are still experimental, at least one has been tested by law students and attorneys and is ready for field-testing in law firms.

These legal analysis computer systems are variously termed by their developers "automated legal reasoning,"8 "artificial intelligence and legal reasoning,"9 and "computer-aided legal analysis."10 The term computer-assisted legal analysis will be adopted in this Comment to refer to these systems. As the term implies, these systems assist the attorney in applying relevant law to the facts of his case in order to arrive at a legal conclusion.

Simply described, computer-assisted legal analysis involves four steps. First, an image of the law is stored in the computer's long-term memory.11 Second, the attorney either answers questions posed by the computer about his case or types into the computer a special description of his case. Third, the computer calculates a correct legal conclusion by applying the law to the attorney's answers or case description. Fourth, the computer performs a task based on the conclusion, such as including a clause in a document or determining if the elements of a legal doctrine are satisfied.

In calculating legal conclusions, computers cannot reason in a completely human sense; they only reason mechanically, following a predetermined, step-by-step procedure.12 To date, the computer reasoning procedures used in computer-assisted legal analysis have been patterned after two types of human reason-

11. Throughout this Comment "long-term memory" is used in place of the technical computer science term "data base." A computer's data base consists of minute pieces of information permanently stored on a magnetic disc or tape. The computer also has a "short-term memory," or "core," in which it can store information temporarily.
12. For a discussion of the limitations of computer reasoning, see H. DREVFUS, WHAT COMPUTERS CAN'T DO (1972); J. WEIZENBAUM, COMPUTER POWER AND HUMAN REASON (1976).
This Comment focuses upon the use of deduction and analogy in computer-assisted legal analysis. Part II describes their role in legal analysis, and Part III relates them to their computer counterparts, propositional logic and semantic network comparison. Part IV describes two deductive and two analogical computer-assisted legal analysis systems. Part V compares these systems' ease of operation. And Part VI explores the potential applications and workable variations of these systems.

II. THE ROLES OF DEDUCTION AND ANALOGY IN LEGAL ANALYSIS

Deduction is a formal method of reasoning based on syllogistic logic. In deduction a major premise is applied to a minor premise to arrive at a conclusion. The classic syllogism is: All men are mortal; Socrates is a man; therefore, Socrates is mortal. In the context of legal analysis, deduction involves the application of a legal rule (major premise) to a specific fact situation (minor premise) in order to reach a legal conclusion. This syllogistic format is easily recast into the IF-THEN format used in computers.

Analogy, on the other hand, is a method of reasoning based on factual comparison. With analogy two situations are compared for factual similarity, and a characteristic of one situation is deemed true of a similar situation. A well-known proverb is in the form of an analogy: "As cold waters to a thirsty soul, so is good news from a far country." In the context of legal analysis, analogy involves (1) perceiving relevant factual similarities between cases, (2) determining the rule of law inherent in one case, and (3) applying the rule to the second case. The steps in analogical comparison can also be recast into a computer format.

Deduction and analogy play a limited role in the process of legal analysis. In addition to the mechanical application of rules

13. See 2 W. JOHNSON, LOGIC 76 (1922).
15. Proverbs 25:25 (King James).
16. See R. CROSS, PRECEDENT IN ENGLISH LAW 182 (2d ed. 1968); E. LEVI, AN INTRODUCTION TO LEGAL REASONING 1-2 (1946); Hermann, supra note 15, at 1137-39. For an example of legal reasoning by analogy, see Meldman, A Structural Model, supra note 10, at 31-33.
or comparison of cases, legal analysis may involve induction and even intuition. In a broad sense, legal analysis involves the consideration of all factors that might influence a judge in his decision, including historical developments, community standards, and fairness to the parties. Because computer-assisted legal analysis systems are based solely on deduction and analogy, these other approaches and factors will play no part in their calculations. Therefore, an attorney using one of these systems will always need to supplement the computer's analysis with his own analysis and judgment.

III. COMPUTER COUNTERPARTS OF DEDUCTION AND ANALOGY

A. Deduction and Propositional Logic

The computer counterpart of deduction is propositional logic, a special application of Boolean algebra. Like deduction, propositional logic is based on premises. The premises are statements that are either true or false and can be represented by letters and other symbols. Unlike deduction, propositional logic sets up premises in an IF-THEN format. For example, the syllogism stated in Part II is recast into the following IF-THEN format:

IF all men are mortal
AND Socrates is a man
THEN Socrates is mortal.

Since the premise "All men are mortal" is assumed to be true, the propositional statement can be shortened to read:

IF Socrates is a man,
THEN Socrates is mortal.

Propositional logic uses "logical connectors" to represent re-

17. For a discussion of the different forms of reasoning used in legal analysis, see Hermann, supra note 14, at 1135-94. For a discussion of the part intuition may play in legal analysis, see Hutcheson, The Judgment Intuitive: The Function of the "Hunch" in Judicial Decision, 14 Cornell L.Q. 274 (1929).
19. George Boole is considered the father of modern symbolic logic, the various simplified versions of which are collectively called Boolean algebra. See G. Boole, An Investigation of the Laws of Thought, on Which Are Founded the Mathematical Theories of Logic and Probabilities (1853). See generally E. Beth, The Foundations of Mathematics 58-63 (2d rev. ed. 1965). Propositional logic is a special application of Boolean algebra that was suggested by Boole himself. See generally H. Pospiszel, Propositional Logic (1974); M. Gardner, Boolean Algebra, Venn Diagrams and the Propositional Calculus, Sci. Am., Feb. 1969, at 110, 112.
relationships between premises. These connectors and their abbreviations, symbols, and meanings are summarized in Table I:

<table>
<thead>
<tr>
<th>LOGICAL Connector</th>
<th>Common Abbreviation</th>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>not</td>
<td>NOT</td>
<td>~ P</td>
<td>not P</td>
</tr>
<tr>
<td>and</td>
<td>AND</td>
<td>P &amp; Q</td>
<td>P and Q</td>
</tr>
<tr>
<td>inclusive or</td>
<td>OR</td>
<td>P ∨ Q</td>
<td>P or Q, or both</td>
</tr>
<tr>
<td>exclusive or</td>
<td>EXOR</td>
<td>(P ∨ Q) &amp; (\neg (P &amp; Q))</td>
<td>P or Q, but not both</td>
</tr>
<tr>
<td>if, then</td>
<td>IF-THEN</td>
<td>P→Q</td>
<td>If P is true, then Q is true</td>
</tr>
</tbody>
</table>

In this Comment, the capitalized terms AND, OR, EXOR, NOT, IF, and THEN are used in their strict sense as Boolean logical connectors.

Through the use of propositional logic, legal rules and statutes can be drafted to enable a computer to determine when the conditions of the rule or statute are satisfied. Consider, for example, section 9-108 of the Uniform Commercial Code, which reads:

> Where a secured party makes an advance, incurs an obligation, releases a perfected security interest, or otherwise gives new value which is to be secured in whole or in part by after-acquired property his security interest in the after-acquired collateral shall be deemed to be taken for new value and not as security for an antecedent debt if the debtor acquires his rights in such collateral either in the ordinary course of his business or under a contract of purchase made pursuant to the security agreement within a reasonable time after new value is given.\(^{20}\)

The statute can be broken down into the following premises:

- \(A\) = a secured party makes an advance
- \(B\) = a secured party incurs an obligation
- \(C\) = a secured party releases a perfected security interest
- \(D\) = a secured party otherwise gives new value
- \(E\) = the new value is to be secured in whole or in part by after-acquired property
- \(F\) = the security interest of the secured party in the after-acquired collateral shall be deemed to be taken for new value and not as security for an antecedent debt

---

\[ G = \text{the debtor acquired his rights in such collateral in the ordinary course of his business} \]
\[ H = \text{the debtor acquires his rights in such collateral under a contract of purchase made pursuant to the security agreement within a reasonable time after new value is given} \]

The statute can then be written using symbols for the premises and logical connectors to describe the relationships:

\[
\begin{align*}
\text{IF } & A \text{ OR } B \text{ OR } C \text{ OR } D \\
\text{AND} & \text{ IF } E \\
\text{AND} & \text{ IF } G \text{ OR } H \\
\text{THEN } & F \\
\end{align*}
\]

In expanded form the statute would appear as follows:

\[
\begin{align*}
\text{IF } & \text{a secured party makes an advance} \\
& \text{OR incurs an obligation} \\
& \text{OR releases a perfected security interest} \\
& \text{OR otherwise gives new value} \\
\text{AND} & \text{ IF the new value is to be secured in whole or in part by the after-acquired property} \\
\text{AND} & \text{ IF the debtor acquires his rights in such collateral in the ordinary course of his business} \\
& \text{OR under a contract of purchase made pursuant to the security agreement within a reasonable time after new value is given} \\
\text{THEN } & \text{the security interest of the secured party in the after-acquired collateral shall be deemed to be taken for new value and not as security for an antecedent debt.} \\
\end{align*}
\]

In computer-assisted legal analysis systems based on propositional logic, the computer determines if the conditions of the statute are satisfied by transforming the IF premises into questions and posing them to the attorney. Based on the attorney's
answers, the computer calculates whether the conclusion of the statute, the THEN premise, is true or false.

The foremost advocate of writing legal rules and statutes in propositional form is Professor Layman E. Allen. Allen reports that in research with attorneys, statutes that have been re-drafted, or normalized, in propositional logic form are easier to understand than the original draft. Computer methods for normalizing statutes have recently been developed.

B. Analogy and Semantic Network Comparison

The computer counterpart of analogy is referred to in this Comment as semantic network comparison. The concept of a semantic network originated with modern linguists, who theorized that sentences can be represented by an inverted tree or network. This concept proved useful in semantic information processing, a branch of computer science concerned with developing computer procedures for processing natural language.


22. Allen & Engholm, supra note 21, at 396.


24. These theories are collectively known as generative grammar, and they include such well-known theories as transformational and case grammar. See, e.g., N. Chomsky, LANGUAGE AND MIND (enlarged ed. 1972); N. Chomsky, SYNTACTIC STRUCTURES (1957); Fillmore, The Case for Case, in UNIVERSALS IN LINGUISTIC THEORY 1 (1968). In transformational grammar the sentence “The boy hit the girl” is diagrammed as follows:

```
  S
 / \  
NP   VP
 |  /  |
T    N  V
 |  /  |
The boy hit the girl
```

25. Semantic information processing is actually a subfield within the broader field of artificial intelligence. McCarty, supra note 19, at 841 n.16. Artificial intelligence projects include programs that play chess, prove mathematical theorems, and solve mass spectrometry problems in organic chemistry. Id. at 841 n.15. Researchers in semantic information processing have developed computer systems that allow the computer and the computer user to carry on a simple dialogue. See note 89 infra.
Semantic network comparison incorporates basic principles from both linguistics and semantic information processing.  

Semantic network comparison, like analogical reasoning, involves the comparison of fact situations. In semantic network comparison the fact situations are represented by semantic networks that consist of two types of elements, things and relations. These two networks can be linked into a larger network. For example, the two networks below contain the identical node “ball”:

These two networks can be linked into the following single network:

---

26. The two semantic network comparison systems discussed below are TAXMAN and Jeffrey Meldman’s system. TAXMAN relies on the linguistic concept of semantic networks and on an artificial intelligence language known as Micro-PLANNER. Meldman’s system relies on the linguistic theory of case grammar and on an artificial intelligence language known as PSL. McCarty, supra note 9, at 850-53; Meldman, A Structural Model, supra note 10, at 29, 47.

27. These are the terms used by Meldman. See Meldman, A Structural Model, supra note 10, at 54. McCarty uses different terminology. See note 58 infra.
The result is equivalent to combining the sentences “John hit the ball” and “The ball belongs to Mary” into the sentence “John hit the ball that belongs to Mary.” This same linking process allows the attorney to enter his description into the computer in groups of two or three words that the computer will assemble into a semantic network. To enter the above semantic network, the attorney types:

(HIT JOHN BALL)
(BELONGS-TO BALL MARY)

In semantic network comparison the computer compares the semantic network of the case with which the attorney is presently concerned with the semantic network of a prior case or statute to determine if the two networks match. For a match to occur, the networks must have identical structure, and the nodes must be identical or equivalent. The following simple example illustrates the basic process.28

Suppose Aaron Aardvark had kicked Zachary Zetz in the leg. The attorney types:

(KICK AARON-AARDVARK LEG)
(PART-OF LEG ZACHARY-ZETZ)

The computer links these two word groups into the following semantic network:

---

28. This example is adapted from Meldman, A Preliminary Study, supra note 10, at 12-14. The example here has been simplified for the purpose of illustrating semantic network comparison.
Suppose that in a prior case Joe Moe had punched Fred Foe in the nose, and that the court had held Moe to have committed a battery. The resulting semantic network is stored in the computer as follows:

In the comparison process, the computer finds that both networks have an identical structure:

The computer also finds that the nodes are all identical or equivalent. "Part-of" is identical in both networks. Because they describe persons, "Fred Foe," "Joe Moe," "Zachary Zetz," and "Aaron Aardvark" are equivalent.** "Kick" and "Punch" are

29. In Meldman's terminology, Fred Foe and the others are all instances of the category person. This implies that they are unique members of that category. Meldman, A Preliminary Study, supra note 10, at 65.
also equivalent because they are analogous kinds of relations. The computer makes this latter determination according to a kind hierarchy previously stored in its long-term memory. In this hierarchy things and relations are grouped into categories and subcategories. “Kick” and “Punch,” for example, are grouped together as two kinds of the category “strike.” Since the attorney’s case would therefore match the prior case, the computer concludes that Aaron Aardvark committed a battery.

IV. COMPUTER-ASSISTED LEGAL ANALYSIS SYSTEMS

A. Deductive Systems

1. JUDITH: Searching for a cause of action

JUDITH was developed at Stanford University by German attorneys Walter G. Popp and Bernhardt Schlink. JUDITH stores the law as a set of premises that are related by various combinations of logical connectors. Its function is to aid the attorney in finding a cause of action appropriate to the facts of his case.

JUDITH first displays on a video terminal all possible legal and equitable remedies. The attorney responds by selecting a remedy appropriate to his case. JUDITH then displays all possible areas of the law in which the selected remedy is appropriate. After the attorney chooses the appropriate area of law, JUDITH

32. JUDITH consists of two memory banks or files: a premises file, in which individual premises are stored in random order, and a construction file, in which the premises from the premises file are logically related. For example, premises P₁ through Pₙ in the premises file might be related in the construction file as follows:
   IF (P₁ AND P₂) OR (Pₙ AND (Pₙ EXOR Pₙ)) THEN Pₙ.
   Id. at 306-09. This results in a hierarchical organization of the law: the premises at each level are elements of premises at a higher level.

Reactions to a hierarchical representation of the law are often hostile or skeptical. It can be argued that this approach is unsuitable for representing law as a body. Id. at 306. However, three arguments favor this approach. First, as Popp and Schlink contend, if a hierarchical organization is left flexible, it can accommodate even difficult hierarchies. Popp and Schlink claim that JUDITH has this flexibility. Id. Second, hierarchically organized knowledge bases have proven successful elsewhere—in clinical therapeutics, organic chemistry, and consulting. Id. at 306 nn.1-3. Finally, a hierarchical organization of law has already proven useful in legal hornbooks and summaries where an outline representation of law is pervasive.
displays the various causes of action available in that area. The attorney then selects a cause of action.

At this point, JUDITH displays a set of premises that are elements of the cause of action or defenses to it. The attorney then has two options: the “case” option and the “specify” option. With the case option he may indicate by typing “true” or “false” whether a premise is supported by the facts of his case. With the “specify” option he may request that the premise be broken down into its constituent premises. When constituent premises are displayed, he again has the same options for each constituent premise. If, for example, the attorney requested that “breach of duty” be broken down, JUDITH would display the constituent premises “statutory duty,” “duty defined by common law,” and “standards of a reasonable man.”33 The attorney could further request that any of these be broken down.

With JUDITH the attorney is able to descend and ascend levels of specificity. Each time the attorney requests that a lower level be displayed, JUDITH remembers the level above it and returns when a sufficient number of lower-level premises have been answered.34 During the session, the attorney can type in comments explaining why he indicated that a particular premise was true or false. After the session, the attorney may print out a summarized log of the conversation with JUDITH.

2. The ABF processor: Simplifying document production

The ABF processor was developed by James A. Sprowl, a research attorney for the American Bar Foundation.35 The ABF processor is a computer-assisted legal analysis system that allows the attorney to write his own program for the production of documents like wills, trust agreements, and probate forms.36

33. Id. at 308.
34. For example, the attorney could cause a premise, P, to be broken down into its sub-premises, P₁, P₂, and P₃. Suppose that the sub-premises were logically connected as follows: IF P₁ AND (P₂ OR P₃) THEN P. If the attorney replied that P₁ was true, JUDITH would wait for instructions concerning P₂ and P₃. If the attorney replied that P₁ was false, JUDITH would automatically return to P and indicate that it was also false. Id. at 310-11.
35. Sprowl, supra note 8. Sprowl was assisted in this project by Dr. Barbara A. Sangster, a linguist at Rutgers, and by Dr. Gregory A. Suski, a computer scientist with the Livermore Radiation Laboratory. Id. at 13 nn.21 & 22.
36. These are the applications contemplated by Sprowl. Id. at 73-76. Law students and attorneys have actually set up systems on the processor to produce simple wills, Illinois divorce complaints, real estate closing agreements, Illinois intestate probate documents, and form 1040 tax returns. Id. at 16.
With the ABF processor, the basic text of the document to be produced is first typed into the computer. The attorney next displays the document on a computer terminal and types in bracketed "space holders" whenever an item in the document must vary for the individual client. For instance, where the client's name is to appear, the attorney types "[the client's name]." At this point the attorney may employ the ABF processor's propositional logic power by typing in statutes or regulations in a normalized form. These normalized statutes control whether certain clauses are included or excluded.

For example, suppose that a clause is to be included in a will only if the estate may be probated without court appearance. In the text of the will immediately preceding the clause to be included, the attorney types: "[IF the estate may be probated without court appearance INSERT]." The attorney then runs the ABF processor in the document assembly mode. In this mode the computer turns bracketed statements in the text into questions. The attorney may respond either by answering these questions or by entering a normalized statute that determines their answer. In the above example, when the computer asks, "May the estate be probated without court appearance?" the attorney enters the following normalized statute:

\[
\begin{align*}
&\text{IF the decedent's children and the named beneficiaries ARE of sound mind and over 20 years old} \\
&\quad \text{AND} \\
&\quad \text{IF the named executor IS a resident of the state of Illinois} \\
&\quad \text{AND} \\
&\quad \text{IF the decedent HAS waived security on the executor's bond in the will} \\
&\quad \text{AND} \\
&\quad \text{IF the decedent's children and the named beneficiaries HAVE consented to an out of court settlement} \\
&\quad \text{THEN} \\
&\text{the estate may be probated without court appearance.}
\end{align*}
\]

The ABF processor stores this statute in its long-term memory under the title "the estate may be probated without court appearance."
appearance."40

When the attorney is finally ready to assemble a will, he runs the system in document assembly mode once again. When the system reaches the bracketed statement, "IF the estate may be probated without court appearance INSERT," it links to the normalized statute and asks questions based on the premises in the statute.41 From the answers the attorney has given, the computer calculates whether to insert the clause.

The ABF processor allows the attorney to store as many levels of statutes in the computer's long-term memory as he desires.42 The attorney may also store alternative versions of questions43 and portions of documents to be incorporated into the main document by reference.44 Sprowl asserts that this "top down"46 method of storing propositions and assembling the document ensures simplicity of operation.46 Sprowl calls the language that the attorney uses to insert clauses "ABF." In designing ABF Sprowl sought to combine the best features of the standard programming languages and at the same time to devise a language understandable to attorneys.47 The ABF processor

40. The ABF processor only uses the first eight nonblank characters for the name, but there is no inherent reason why a system could not use more. Id. at 18 n.38, 54 n.57.
41. See text accompanying note 82 infra.
42. For example, the quoted normalized probate statute would constitute one level. The attorney could store a second level by entering a normalized residency statute under the title "the named executor is a resident of the state of Illinois." In the document assembly mode, rather than simply asking "Is the named executor a resident of the state of Illinois?" the ABF processor would ask questions based on the premises of the normalized residency statute. Sprowl, supra note 8, at 56-58.
43. When it encounters bracketed space holders in document assembly mode, the ABF processor turns them into questions by prefixing them with the words, "What is." The space holder "[the client's name]" would become "What is the client's name?" If the resulting question were inadequate, the attorney could store an alternative question that would be asked in its place. A desirable alternative question to the one above might be: "Please type the client's full name, including one middle initial." Id. at 54-56.
44. A clause that is common to several documents may be stored and then incorporated into the individual documents by referring to the title of the clause. This feature saves retyping the clause in each document. The reference appears in a bracketed space holder—for example, "[the passage defining the powers and duties of the executor]." Id. at 53-54.
45. The term "top down" comes from computer science and refers to the designing of computer programs from the general conceptual level down to the specific program level. See, e.g., C. McGowan & J. Kelley, Top-Down Structured Programming Techniques (1975). The ABF processor employs a unique method of top down programming since an operative system may be produced with only the top levels functioning long before the lower levels have been created. Sprowl, supra note 8, at 51-53.
46. Id. at 61-63.
47. In designing ABF, Sprowl studied the following programming languages: BASIC,
has been tested by both law students and attorneys, and in one instance, law students were able to set up a form 1040A tax return completion system in twelve hours.49

B. Analogical Systems

1. TAXMAN: Identifying types of corporate reorganizations

TAXMAN was developed at Stanford by L. Thorne McCarty, who is presently continuing research on the system at the State University of New York in Buffalo.50 TAXMAN's function is to identify type B and C corporate reorganizations as defined by the Internal Revenue Code.51 Although TAXMAN presently is capable of performing only a rudimentary form of legal rea-

COBOL, FORTRAN IV, PL/I, ALGOL, PASCAL, APL, and ALGOL 68. Id. at 14. Based on his previous experience in teaching programming to law students, Sprowl designed ABF to avoid many of the features difficult to teach students lacking a strong mathematics background. Id. Yet ABF implements a full set of Boolean logical connecters and has advanced arithmetic and algebraic capabilities. Id. at 20-23.

48. See note 36 supra.

49. Sprowl, supra note 8, at 61. One reason for the short set-up time is that the ABF processor sacrifices control over question order for ease in setting up the system. Questions are asked in the same order that their corresponding bracketed space holders appear in the document, rather than in some more logical order. The set-up time would be greatly increased if a logical order were sought—for example, by having the processor assemble a "dummy" report that requires the information in a certain order. Id. at 61 n. 69.

50. McCarty, supra note 9, at 837. The TAXMAN program was written in 1972-73 and was first discussed in a paper presented at a workshop entitled "Computer Applications to Legal Research and Analysis" at Stanford Law School, April 28-29, 1972. L. McCarty, Interim Reports on the TAXMAN Project: An Experiment in Artificial Intelligence and Legal Reasoning, in Artificial Intelligence Techniques in Legal Problem Solving (June 1, 1973) (mimeograph, Stanford Law School). McCarty has since written a proposal for an extended version of TAXMAN. L. McCarty, B. Sangster & N. Sridharan, The Implementation of TAXMAN II: An Experiment in Artificial Intelligence and Legal Reasoning (Jan. 16, 1979) (mimeograph, Rutgers University).

51. I.R.C. §§ 354-56, 358, 361-62, 368. A type B reorganization is defined as the acquisition by one corporation, in exchange solely for all or a part of its voting stock (or in exchange solely for all or a part of the voting stock of a corporation which is in control of the acquiring corporation), of stock of another corporation if, immediately after the acquisition, the acquiring corporation has control of such other corporation (whether or not such acquiring corporation had control immediately before the acquisition).


A type C reorganization is defined as the acquisition by one corporation, in exchange solely for all or a part of its voting stock (or in exchange solely for all or a part of the voting stock of a corporation which is in control of the acquiring corporation) of substantially all of the properties of another corporation . . . .

I.R.C. § 368(a)(1)(C).
soning, McCarty believes that with certain extensions it could serve practicing tax attorneys.\textsuperscript{52}

Before an attorney uses TAXMAN, semantic networks of type B and C corporate reorganizations are stored in the computer's long-term memory. These two semantic networks are actually composed of smaller semantic networks of component legal concepts such as stock ownership,\textsuperscript{53} corporate control,\textsuperscript{54} corporate acquisition,\textsuperscript{55} and stock transfers.\textsuperscript{56}

In using TAXMAN the attorney must first describe as a series of word groups the corporate reorganization to be analyzed. For example, suppose one of the facts of the corporate reorganization is that "Phellis owns 100 shares of common stock issued by Smithco Corporation."\textsuperscript{57} The attorney first breaks this sentence down into shorter sentences:

- Phellis owns stock\textsubscript{1}.
- Stock\textsubscript{1} is 100 shares.
- Stock\textsubscript{1} is part of stock\textsubscript{2}.
- Stock\textsubscript{2} is common stock\textsubscript{3}.
- Smithco issued stock\textsubscript{2}.
- Smithco is a corporation.

The attorney next translates these shorter sentences into word groups and enters them in the computer in the form "(relation/thing 1/thing 2)."\textsuperscript{58} The above sentences would be entered as follows:

- (OWN PHELLIS S1)
- (NSHARES S1 100)
- (PART-OF S1 S2)
- (COMMON-STOCK S2)
- (ISSUE SMITHCO S2)

\textsuperscript{52} McCarty, supra note 9, at 858, 881-92. For a detailed discussion of the proposed extensions, see L. McCarty, B. Sangster & N. Sridharan, supra note 60.

\textsuperscript{53} McCarty, supra note 9, at 855-56.

\textsuperscript{54} I.R.C. § 368(c) defines control as "the ownership of stock possessing at least 80 percent of the total combined voting power of all classes of stock entitled to vote and at least 80 percent of the total number of shares of all other classes of stock of the corporation." This concept is relatively easy to program into the TAXMAN system. McCarty, supra note 9, at 871.

\textsuperscript{55} McCarty defines acquisition as a series of exchanges, each of which consists of a pair of transfers. Id. at 872.

\textsuperscript{56} Id. at 866-67.

\textsuperscript{57} This example is adapted from McCarty, Id. at 852-54.

\textsuperscript{58} McCarty uses the terms predicate and object in place of the terms thing and relation. Id. at 851. Although predicate and object imply a different meaning, their interrelation in a semantic network is effectively the same as thing and relation.
The computer then links the series of word groups into a semantic network.\footnote{59}

Note that in TAXMAN nouns used to rename or describe other nouns, such as "COMMON-STOCK" or "CORPORATION," are considered to be relations. Note also that TAXMAN allows word groups that include only one thing.

TAXMAN's semantic network comparison operates much like the example given in Part II,\footnote{60} except that TAXMAN has no kind hierarchy stored in the computer. The attorney himself must classify certain nodes as kinds of general categories at the time he invokes the semantic network comparison. For example, suppose the attorney had previously typed in the series of word groups shown above for comparison with the stock ownership

\begin{itemize}
  \item[59.] The semantic network would appear as follows:
  \begin{itemize}
    \item \textbf{Phellis}
    \item Own
    \item S1
    \item Part-of
    \item S2
    \item Issue
    \item Smithco
    \item Corporation
    \item Nshares
    \item Common-stock
    \item 100
  \end{itemize}
\end{itemize}

\begin{itemize}
  \item[60.] McCarty does not describe TAXMAN in terms of semantic network comparison; instead he describes it in terms of commands that perform pattern matching. This is because McCarty's programming language, Micro-PLANNER, is most easily described as a series of commands that search a semantic network for certain patterns. A set of pattern matching commands in Micro-PLANNER has the same effect as storing a semantic network based on the law and then comparing it with the semantic network entered by the attorney. For a detailed discussion of TAXMAN's pattern matching, see McCarty, \textit{supra} note 9, at 855-62.
\end{itemize}
network stored in the computer. In TAXMAN, the command that invokes such a comparison is typed in the format (GOAL (STOCKHOLDER ?O?C) ABSTRACT)—where "?O" is the owner of stock and "?C" is the corporation issuing stock. In this case, the attorney types (GOAL (STOCKHOLDER PHELLIS SMITHCO) ABSTRACT). This classifies "PHELLIS" as a kind of owner of stock and "SMITHCO" as a kind of corporation issuing stock. Using this information, TAXMAN performs the semantic network comparison, finds a match, and affirms to the attorney that Phellis was a stockholder of Smithco.

One of the most important features of TAXMAN is that it allows an attorney to type in a series of semantic descriptions, or states, that can be labeled as existing at different points in time. A transition from one state to another is called an event. A transfer of stock, for instance, is an event defined by two states: in the first, X owns certain stock but Y does not, and in the second, Y owns stock but X does not. Thus, the attorney can describe a series of occurrences, a function necessary for the analysis of type B and C corporate reorganizations.

In order to analyze a corporate reorganization, the attorney

---

61. This example is adapted from McCarty. Id. at 858-59.
62. Id. at 859. The word "STOCKHOLDER" in the command refers to a set of commands that are equivalent to the following semantic network:

```
OWN
    /\  
   / \ 
OWN
 -?O-

PART-OF
    /\  
   / \ 
PART-OF
   -?P- 

ISSUE
    /\  
   / \ 
ISSUE
 -?S-

COMMON-STOCK
    /\  
   / \ 
COMMON-STOCK
 -?C-
```

The values of the nodes with question marks are replaced during the semantic network comparison. Id. at 856.
63. At the time the semantic network comparison is invoked, "PHELLIS" replaces "?O" and "SMITHCO" replaces "?C" in the STOCKHOLDER semantic network. As the comparison progresses, "?P" is replaced by "S1," and "?S" is replaced by "S2" from the original semantic network.
64. Id. at 865-70.
65. Id. at 866-67.
first types in the complete semantic network of his entire organization. He then invokes the semantic network comparison by typing:

\[(\text{GOAL (B-REORGANIZATION ?A ?C ?T) ABSTRACT})\]
\[(\text{GOAL (C-REORGANIZATION ?A ?C ?T) ABSTRACT})\]

In this entry "?A" is the acquiring corporation, "?C" is the corporation acquired, and "?T" is the time the acquisition is completed. McCarty has successfully demonstrated that this procedure can identify type B and C corporate reorganizations.66

2. Jeffrey Meldman's system: Identifying assault and battery

Another analogical system has been proposed in a thesis written by Jeffrey Meldman at the Massachusetts Institute of Technology.67 The system's function is to identify assault or battery in a legal fact situation. To operate the system, the attorney sits at a computer terminal and types in a series of word groups describing his case.68 The system responds by indicating whether the attorney's case has matched one of the system's semantic network representations of assault and battery. Unlike TAXMAN, the system asks questions when additional information is needed and tells the attorney which cases or legal doctrines it relied upon in reaching its conclusion.69

Meldman asserts that when techniques of semantic information processing are sufficiently refined, the attorney and the computer will communicate in complete sentences.70 He gives the following example of how his system might then be used:

66. McCarty tested TAXMAN using the facts from United States v. Phellis, 257 U.S. 156 (1921). TAXMAN correctly found no type B reorganization to be present but did find a type C reorganization. McCarty, supra note 9, at 876-81.
68. Meldman gives no examples in his thesis of how an intricate fact situation would be entered as word groups. He does give the following simple example of how the sentence "Marsha Moe believes that her attorney is Richard Roe" would be entered:
\[(\text{BELIEF MARSHA-MOE (ATTORNEY MARSHA-MOE RICHARD-ROE)})\]
Meldman, A Preliminary Study, supra note 10, at 93. Note how Meldman allows the entire second word group to constitute the "thing 2" of the first word group.
70. At present the user and the computer can communicate in an artificial intelligence language called PSL (Preliminary Study Language). The computer is able to translate PSL statements into data structures and machine procedures. Hopefully the computer will eventually be able to perform similar translations directly from English. Id. at 29.
USER: With the purpose of frightening Gordon Good, Howard Hood visibly points a saturday-night special at him. The saturday-night special is not loaded.

SYSTEM: What is a saturday-night special?

USER: A saturday-night special is a kind of pistol.

SYSTEM: Did Gordon know that the saturday-night special was unloaded?

USER: No.

SYSTEM: Howard Hood appears to be liable to Gordon Good for an assault.

USER: Why is there liability for an assault?

SYSTEM: According to the doctrine adopted in Jones v. Smith, an action in assault is established by two components: apprehension and intent. These components appear to be satisfied by the facts at hand.71

Meldman's system has three levels of semantic network comparison. The lowest level is purely analogical and operates exactly like the semantic network comparison described in Part II.72 The two higher levels combine analogical comparison with deductive reasoning.73 The two higher levels examine the semantic network to see if the facts of the case satisfy the general doctrines of assault or battery or one of several specific types of assault or battery. For example, if the semantic network contained the elements “plaintiff,” “defendant,” “contact,” and “intent,” and lacked the counter-element “consent,” the general doctrine of battery would be satisfied.74 If the semantic network contained the elements “contact to the plaintiff,” “by a movable object,” “that is thrown by the defendant,” the system would identify a specific type of battery involving contact with a projectile. Each time the attorney typed in a description of his case the system would compare the case at all three levels before it concluded that no match existed.

Unfortunately Meldman's proposed system has severe limitations. Its analysis is based on very simplified doctrines of as-

71. Id. at 29-30.
72. Meldman calls this analogical level “instantiating by example.” Meldman, A Preliminary Study, supra note 10, at 160.
73. Meldman calls these two methods of comparison “instantiation by element” and “instantiation by type.” Id. at 160-63.
74. Meldman takes his definition of battery from Prosser's treatise on torts: “One is liable to another for unpermitted, unprivileged contacts with his person, caused by acts intended to result in such contacts, or the apprehension of them, directed at the other or a third person.” W. PROSSER, HANDBOOK OF THE LAW OF TORTS 43 (1941).
sault and battery, and its design relies upon its very small knowledge base. The system also ignores time as a factor and uses artificial categories in its kind hierarchy. The system has not been fully implemented on a computer, although the implementation of a single example from Meldman’s thesis has been achieved at Stanford. Nevertheless, Meldman argues that the system could be implemented and expanded with more sophisticated representations of assault and battery. He also argues that the system could be extended to contract law and other areas of tort law.

V. COMPARATIVE EASE OF OPERATION

The deductive and analogical systems described above differ in ease of operation. Deductive systems typically ask the attorney to answer a series of questions with “yes” or “no,” or to identify a series of premises as “true” or “false.” For example, when programmed with a statute concerning the probate of an estate without court appearance, the ABF processor asks the following questions:

- Are the decedent’s children and the named beneficiaries of sound mind and over 20 years old?
- Is the named executor a resident of the state of Illinois?
- Has the decedent waived security on the executor’s bond in the will?
- Have the decedent’s children and the named beneficiaries consented to an out of court settlement?

75. For example, components like the defense of privilege have been omitted. Meldman, A Structural Model, supra note 10, at 68.
78. For instance, the kind hierarchy contains the category “movable-object” which includes hats, pistols, and bricks. Id. at 63. A less artificial approach would be to classify these objects in separate categories and attach to each of them the feature “movable.” J. King, supra note 76, at 5. For example, a hat might be classified as an article of clothing, a pistol as a weapon, and a brick as building material. Besides attaching the feature “movable,” other appropriate features might be attached. For example, a brick might have the features “heavy” and “usable as a weapon.”
79. J. King, supra note 76. The paper also discusses the strengths and weaknesses of Meldman’s system. Id. at 5-6.
81. Another possibility would be to ask the user to answer a multiple choice question.
82. See text accompanying note 39 supra.
83. Sprowl, supra note 8, at 46-47.
The attorney responds to these questions sequentially with a "yes" or "no."

One advantage of deductive systems is that the attorney has only a limited number of ways in which to respond. The computer rejects any inappropriate responses (responses other than "yes" and "no" or "true" and "false"), then repeats the question.

A disadvantage of deductive systems is that the questions asked may be ambiguous. In the above example, the second question does not explain what "resident" means and the attorney might be unsure whether the named executor is a resident of the state of Illinois. This drawback is partly alleviated by the ability of deductive systems to break the premises upon which questions are based into their constituent premises.65

Analogical systems typically ask the attorney to type in a description based on the relevant facts of his case. As shown in Parts III and IV, the description consists of a series of word groups such as the following:

(OWN PHELLIS S1)
(INSHARES S1 100)
(PART-OF S1 S2)
(COMMON-STOCK S2)
(ISSUE SMITHCO S2)
(CORPORATION SMITHCO)

Analogical systems have two problems which make them difficult to operate. The first is that the user cannot be sure which facts to include in the description. For instance, in the example above, the attorney might be unsure whether or not to specify who Phellis is.66 Failure to include all the pertinent facts causes the computer to arrive at an incorrect conclusion.67 Meldman's system would deal with this problem by asking for addi-

---

84. The ABF processor is limited by the fact that the attorney must respond sequentially. Ideally, the attorney should be able to back up and change answers and to skip forward to subsequent questions.

85. In JUDITH, an unclear premise can be broken down by invoking the specify option. In the ABF processor, a question can be broken down by adding a lower level legal rule or statute.

86. The attorney could specify whether Phellis is a natural person or a corporation by typing either (PERSON PHELLIS) or (CORPORATION PHELLIS).

87. For each omitted fact a corresponding portion of the resulting semantic network will not be constructed by the computer. If the semantic network is incomplete, the computer fails to find a match between both the structure and the nodes of the two networks.
tional facts if insufficient facts were given.

Even if the attorney includes all the relevant facts, a second problem arises in that no simple formula exists to translate facts into word groups. There are many conceivable ways to break down a sentence. For instance, in the above example an attorney might not use the variable \( S2 \) to represent "common stock issued by Smithco." He might instead use the variable \( \text{COMMON-STOCK} \) and enter the description as follows:

\[
\begin{align*}
(\text{OWN PHELLIS } S1) \\
(\text{NSHARES } S1 100) \\
(\text{PART-OF } S1 \text{ COMMON-STOCK}) \\
(\text{ISSUE SMITHCO COMMON-STOCK}) \\
(\text{CORPORATION SMITHCO})
\end{align*}
\]

This semantic description is incorrect and causes the computer to arrive at an incorrect legal conclusion.\(^{88}\)

At least one semantic information processor has been developed in which the user and computer communicate in complete sentences.\(^{89}\) Interaction in complete sentences might eventually help avoid the operational difficulties of analogical systems.

VI. PRACTICAL APPLICATIONS

The four systems discussed in Part III were all developed

---

88. Here the semantic network comparison finds identical or equivalent nodes, but the structure does not match.

89. Terry Winograd has developed a program that can carry on a conversation about a make-believe world of blocks. He gives the following sample dialogue in which the user's comments are typed in lower case and the computer's responses are typed in upper case:

pick up a big red block.

OK.

... find a block which is taller than the one you are holding and put it into the box.

BY "IT," I ASSUME YOU MEAN THE BLOCK WHICH IS TALLER THAN THE ONE I AM HOLDING. OK.

... what does the box contain?

THE BLUE PYRAMID AND THE BLUE BLOCK

what is the pyramid supported by?

THE BOX.

... how many blocks are not in the box?

FOUR OF THEM.

for use by the practicing attorney. Although none of these systems has yet been field-tested, it is possible to consider the ways in which they might eventually be used.

JUDITH would be useful to an attorney in analyzing a case for a cause of action. He could use JUDITH to guide his initial analysis or to check his own independent analysis to make sure he has not overlooked any significant elements or defenses. He could later refer to the summary of his dialogue with JUDITH in writing trial memoranda. JUDITH might also be modified to allow the attorney to store new knowledge about a particular cause of action in the form of new premises.

The ABF processor would be used to generate any type of document in which legal rules determine whether certain clauses should be included. James Sprowl asserts that the system could assemble "a vast range of documents," including those with passages that vary widely with different clients—wills, trust agreements, and probate forms. Sprowl contemplates that the person who programs and interacts with the ABF processor will usually be the attorney using the system. However, a document assembly system could be programmed more effectively by an attorney specializing in a particular field of law. Also, the person who interacts with the system could be a non-lawyer, such as a client at a legal clinic.

90. Popp & Schlink, supra note 31, do not report any actual testing of JUDITH. Sprowl reports testing of the ABF processor by law students and attorneys and plans to have it voluntarily field-tested by law firms within telephone range of a university-owned computer system, where the processor can be set up. He tentatively hopes to test the processor in Chicago and Champaign, Illinois; Lansing, Ann Arbor, and Detroit, Michigan; Minneapolis, Minnesota; and Portland, Oregon. Sprowl, supra note 8, at 16-17. McCarty reports no field-testing of TAXMAN. He believes that TAXMAN is not yet ready to be used by practicing attorneys. McCarty, supra note 9, at 882. Meldman's system is merely a proposed design and had not yet been implemented for testing by attorneys.

91. Before development of computer-assisted legal analysis systems began, artificial intelligence scholars were speculating about the tasks these systems would perform. See, e.g., Buchanan & Headrick, Some Speculation About Artificial Intelligence and Legal Reasoning, 23 Stan. L. Rev. 40 (1970). Buchanan and Headrick isolated four basic tasks: (1) finding conceptual linkages in pursuing goals, (2) recognizing facts, (3) resolving rule conflicts, and (4) finding analogies. Id. at 53.

92. As described by Popp and Schlink, JUDITH's knowledge base cannot be modified by the attorney. However, the program could easily be extended to allow the attorney to insert his own premises into the premises file and insert the matching logical connectors into the construction file. See note 32 supra.

93. Sprowl, supra note 10, at 74-76.

94. For example, Cook County developed a system that generated divorce documents after a computer-conducted interview with a client at a public legal clinic. R. McCoy, supra note 7.
TAXMAN and Meldman’s proposed system would both analyze a description of the attorney’s case to see whether it matched one of the statutes, cases, or legal doctrines stored in the computer’s long-term memory. Both would identify a match, and Meldman’s system would also describe the case or legal doctrine upon which the match was based. Ideally, these systems will serve attorneys in the same way computer-assisted diagnosis serves doctors.85

These systems might also be used for planning purposes. In an extended version of TAXMAN, an attorney advising his client on a proposed corporate reorganization might type in different versions of the reorganization to see which produces the most desirable tax outcome. Alternatively, the attorney could type in a description of a proposed reorganization and the desired tax outcome. He could then ask the computer what changes in the reorganization would be necessary to achieve that outcome.

Eventually, analogical systems might be used to test theories concerning the nature of legal reasoning. The developer of TAXMAN believes that the computer might be the most important tool ever devised for this purpose.86

Both analogical and deductive systems could be used in continuing legal education if an instructional mode were added. In deductive systems the mode could be invoked as each question was asked, and in analogical systems it could be invoked

95. MYCIN is a computer program that aids doctors in deciding whether their patients need antimicrobial therapy. The dialogue between a doctor and MYCIN is comparable to the dialogue between an attorney and JUDITH. Popp & Schlink, supra note 31, at 314 n.4. See generally Shortliffe, Axline, Buchanan, Merigan & Cohen, An Artificial Intelligence Program to Advise Physicians Regarding Antimicrobial Therapy, 6 COMPUTERS & BIOMEDICAL RESEARCH 544 (1973).

96. McCarty states:

[W]hatever its practical applications, the TAXMAN system provides, I claim, an important tool for the development of our theories about legal reasoning.

The TAXMAN system adds a strong dose of precision and rigor to... discussions of linguistic and conceptual [legal] problems. Its critical task is to clarify the concepts of corporate reorganization law in such a way that they can be represented in computer programs. This requires a degree of explicitness about the structure of these concepts that has never previously been attempted. When we describe concepts in this way, we implicitly articulate theories about them; when we run the computer programs that embody these concepts, we test out the implications of our theories. Used in this fashion, the computer is the most powerful tool for expressing formal theories and spinning out their consequences that has ever been devised.

McCarty, supra note 9, at 839-40.
after the semantic network comparison was completed. When the attorney invoked the instructional mode, the computer would display relevant cases, statutes, and legal doctrines. If the systems were maintained by experts in various areas of the law, the practicing attorney could learn about new developments in that area in the context of his day-to-day work.

VII. CONCLUSION

Deductive and analogical computer-assisted legal analysis systems may eventually become important tools for the practicing attorney. Since they are only capable of a mechanical form of analysis, these systems cannot completely replace the attorney's own legal analysis. Once made available to law firms, these systems could be put to a variety of uses, including case analysis, document drafting, and continuing legal education. With the dramatic progress in this area, computer-assisted legal analysis systems will probably come into widespread use in the 1980's.

Mark Morrise