Hypertext-aided Legal Expert Systems

HyperREASONER: An Example of the Integration of Hypermedia and Expert System Technology

GERALD QUIRCHMAYR, ROLAND TRAUNMÜLLER, WOLFGANG BAUER

1. Introduction

Rule-based expert systems have traditionally been based on the assumption that the user thinks in terms of rules, because law is formulated in the form of rules. The problem with this approach is that the user interfaces resulting from this metaphor have not always been satisfactory. When hypertext systems were introduced it was therefore a natural behaviour to view them as a replacement for expert systems and artificial intelligence technology. When it came to modelling the process of decision making, hypermedia systems have, however, fared poorly if they lacked an integration with expert systems. The lesson learned once more is that one technology alone cannot provide an overall solution for constructing working consultation systems which are to be accepted by practicing lawyers. If we have a closer look at the development of information technology, it does quickly become clear that information systems technology is under continuous development, each generation of systems being built with the goal of overcoming the shortages of the previous one (cf. [Turner 1988]). Evolutionary approaches gradually embarking on newly emerging technologies have been shown to be the best way for widening the spectrum of problems which can be solved, whereas strategies using the new technology in an isolated way have failed dramatically in practice. That is why the main idea of our approach is to make use of hypermedia technology in an integrated way by combining it with a traditional rule-oriented inference mechanism and allowing this mechanism to access databases as sources of information. As the major problem of existing legal expert systems is the maintenance of the knowledge base, one focus of the project has been on designing an easy to use interface for the knowledge engineer.

In an ideal situation the lawyer dealing with the system should be provided with a full-scale natural language interface, which unfortunately is not yet at hand. Therefore we have decided to implement a graphical
interface based on the browser concept (similar to the concept used in Smalltalk). The overall system architecture follows the general concept defined for legal expert systems (see [Fiedler 1984]), which means that the kernel of our reasoning system does consist of the following components:

- The dialogue component providing user mode and knowledge engineering mode;
- The inference engine;
- The active knowledge base consisting of rules and facts;
- An arbitrary number of knowledge bases stored on disk;
- The knowledge acquisition component;
- The explanation component (at the current state of the project rather rudimentary).

2. The System Concept

As pointed out in the introduction, the system concepts follows the general rules for the development of expert system shells. As one major goal was to investigate the possible benefits of hypermedia technology added to expert systems technology, the focus of the implementation has been on the user interface, mainly on the interface for knowledge engineering, because we were aiming at a shell which a lawyer should be able to use without the support of a knowledge engineer. In addition to the use of hypermedia technology for creating a better user interface, we have also used Toolbook's pre-defined interface to dBase to demonstrate how elements of a knowledge base can be stored in a database and therefore be made accessible for maintenance and integration purposes. A further advantage of this add-on is that additional information can be entered in the form of batch input files, which helps to overcome the usual isolation of expert systems. So the problem of reading data into a knowledge-based system has been reduced to format conversion, for which solutions are well known from database technology.

2.1. The user interface of HyperReasoner

As making knowledge engineering as easy as possible was a major goal of the project (the description of a previous prototype leading to HyperReasoner is described in [Kappes Quirchmayr 1990]), the interface had to be merely graphically oriented and mouse driven [Quirchmayr Bauer 1993].
Considering lawyers as users of the system, the formulation of rules had to be kept as simple as possible. A major problem of many expert system shells is that negation can be expressed in very limited and tricky ways only, which are insufficient for the legal domain and do often result in the production of unnatural and unnecessarily complex rules. These do in turn lead to asking the user questions which can easily be misunderstood. Further deficiencies of first generation legal expert systems are described in [Susskind 1987] and [Fiedler Traunmüller 1989]. For reasons of clarity and simplicity the well-known example of contracts with minors is used to illustrate the capabilities of HyperReasoer's knowledge engineering interface.

**FIG. 1. The Knowledge Engineering Interface**

As shown in Fig.1 the definition of rules is based on the concept of graphically composing legal concepts in the form of nodes and connecting them by logical expressions (AND, OR, NOT). NOT, in contrast to many other shells, can be used not only for the negation of facts, but also for the negation of whole nodes (*Inferences* in the screen print of Fig. 1). The type of logical connection is chosen by simply clicking the AND-, OR, or NOT-button. Goals and Arguments are specified by selecting elements from the lists of Answers, Inferences, and Questions. To make review and debugging of defined rules easy, we have implemented a browser concept, which
allows the user to walk through the rule tree in a top-down way or to access rules assigned to a specific node directly.

2.2. The knowledge engineering paradigm of HyperReasner

The knowledge engineering component of HyperReasner was designed with the goal of keeping it as simple as possible for the user in order to make it usable for people without thorough training in knowledge engineering, and with the aim of providing true negation, which makes it less complicated to represent negations which do quite frequently occur in statutes and laws. Toolbook provided us with the environment necessary for constructing an easy to handle knowledge engineering interface. For implementing true negation an old trick originally developed for defining confidence factors had to be adapted: truth values are assigned to each node, depending on its logical structure and on the number of its arguments. Once truth values have been assigned, to the basic nodes (Questions) by the user, these values can be propagated through a classical AND-/OR-tree. Now the trick is to internally assign numeric values, 1 representing true, 0 indicating that a value is unknown, and -1 standing for false. A very positive side effect of this strategy is, that, once strict 3-valued Logic is
given up, an extension of the model to dealing with Fuzzy Logic becomes possible.

The underlying concept of the system therefore is as follows:

Knowledge engineer

the main page

the fact editor

the rule editor

the browser

End-user

accesses

which uses

the main page

the query page

the result page

(accessing rule and fact pages)

the fact page

The fact and rule pages are constructed from the fact and rules base, stored in dBase and ASCII format to allow, as pointed out in 2.1, access to external information. In case the end-user has to interrupt the consultation, the information he has entered is stored and the session can be resumed later at the point where the user has quit.

3. Results of the Prototype Development and First Tests

Basing HyperReasoner on a hypermedia system has provided us with a lot of benefits, especially concerning integration and user interface-design. The price which has to be paid in terms of performance, however, is very high. Tests of the prototype have shown that due to the inference engine being implemented in Toolbook's script language, it is not advisable to run the system on anything below 80486-technology. As discussed in [Traunmüller Quirchmayr 1991], hypermedia systems are the best way for quickly reaching the integration of database, expert, text processing, and traditional office information systems. Therefore it is worth to apply this strategy, perhaps with the modification that time critical parts, such as the kernel of the inference engine, should be reimplemented in a compiler language.

4. Modelling Regulations of Regional Development Law

The first realistic test HyperReasoner had to undergo was its application in Spanish regional development law. Research concerning priority-oriented
models of legal and administrative decision making have for a long time been only theoretical. With the example of formalising strategies used by INEM (the Spanish national institute for employment) and the EEC for allocating subsidies we have shown one possible way for putting formal logic-based models to work in legal environments. To be able to represent contradictions naturally occurring between the interests of different applicants and different institutions granting subsidies, we used Kripke-type frames based on possible worlds.

4.1. The formal model

As described in [Baaz et al. 1993] the construction used for the formal model of regional development programs consists of a static part which is used for describing and a dynamic part which is used for deciding situations of allocation of grants:

a) the static part:

- the situation of allocation of subsidies is represented by a Kripke-type frame
- the representatives consist of totally ordered non empty sets of worlds, where a representative stands for all grants given by an institution or all companies a person is handling the applications for.
- the truth values determining the priorities are elements of $C_1 \times \ldots \times C_r \times \{0,1\}$, where $C_i$, $\ldots$, $C_r$ are the components of an application. $C_i$, $\ldots$, $C_r$ are totally ordered non empty sets representing the criteria according to which an application is evaluated.
- we represent the application of a local corporation to an institution $a$ as follows:
  a) a true statement $APP_{c,a}$ valued at $c$ by the actual situation of the company (i.e. the value $\langle c_1, \ldots, c_r,1 \rangle$ is assigned to $APP_{c,a}c_i \in C_i; i = 1, \ldots, r$).
  b) all elements of $a$ are connected to $c$ by arrows
  c) a false statement $APP_{c,a}$ for all $c' \neq c$ valued at $c$ by the actual situation of the local corporation (i.e. the value $\langle c_1, \ldots, c_r,0 \rangle$ is assigned to $APP_{c,a}c_i \in C_i; i = 1, \ldots, r$). The underlying assumption is that the grant can be given to only one corporation.
- $\neg APP_{c,a}$ is true for some element (is false for all elements) of the institution $a$ if $a$ does (not) grant a subsidy for the representative $\beta$, $c \in \beta$. 
b) the dynamic part:
   - for finite situations derivations are defined by cancelling
     a) applications which have already been successful (= have already received a grant)
     b) applications with the weakest arguments; the weakest argument is determined by the lexicographic order of the ring of \(<c_1, \ldots, c_r, 1>\) which follows the total ordering of the criteria \(C_1, \ldots, C_r\) representing the philosophy on which the decision making process is based.

The central issue being to use the framework for representing different decision philosophies, we only considered propositional logic for reasons of simplicity and clarity.

The truth values \(V\) are elements of \(C_1 \times \ldots \times C_r \times \{0,1\}\). \(C_1, \ldots, C_r\) are totally ordered non empty sets.

\(<c_1, \ldots, c_r, 1>\) are designated to be true. (As a consequence of this definition the properties of usual classical modal propositional logics could be saved, cf. [Rescher 1969]). Generalisations to other systems of multiple-valued (first order) logic are obvious.

Let \(L\) be a propositional language with variables \(\text{Var}\) and operations \(\neg, \land, \lor, \Rightarrow\): a presituation is a frame \(F = <D, \text{PHIL}, R, v>\), where

\(D = \{\alpha, \alpha_2, \ldots\} \neq \{\}; \alpha_i\) are disjoint non empty totally ordered sets of worlds (the representatives); the universe \(U = \bigcup \alpha_i\).

\(\text{PHIL}: D \to P(\{C_1, \ldots, C_r\} \times \{C_1, \ldots, C_r\})\) has the property that \(\text{PHIL}(\alpha)\) is a total ordering of \(C_1, \ldots, C_r\). \(\text{PHIL}(\alpha)\) therefore induces a lexicographic ordering \(<_i\) on \(C_1 \times \ldots \times C_r \times \{1\}\) that can be extended to \(V\) by

\[u <_{\text{PHIL}(\alpha)} w \iff u <_i w \quad \text{and} \quad u \text{ and } w \text{ are designed to be true, or}\]

\(u\) is not designed to be true and \(w\) is designed to be true, or

\[w <_i u \text{ and } u \text{ and } w \text{ are not designed to be true}\]

\(R = U \times U\)

\(s\): \(U \to C_1 \times \ldots \times C_r\) (is used to describe the situation of the corporation)

\(v: U \times L \to V\) is inductively defined by

\[v_c(A) \in \{s(c)\} \times \{0,1\}\] for all \(A \in \text{Var}\).

\[v_c(\neg A) = \langle c_1, \ldots, c_r, 1 - e \rangle\] for \(v_c(A) = \langle c_1, \ldots, c_r, e \rangle\)

\[v_c(A \land B) = \min_{\text{PHIL}(\alpha)} \{v_c(A), v_c(B)\}\] for \(c \in \alpha\)
\[
v_c(A \lor B) = \max_{PHIL(\alpha)} \{v_c(A), v_c(B) \} \text{ for } c \in \alpha
\]
\[
v_c(\neg A) = \min_{PHIL(\alpha)} \{v(A)\}_{c, \beta, \epsilon R} \text{ for } c \in \alpha, \text{ the only condition where different philosophies lead to different results}
\]

A situation of allocation of subsidies is a situation extended by an injective function APP: \(U \times D \rightarrow \text{Var} \) such that \(v_c(\text{APP}_{c, \alpha}) \) is designed to be true and \(v_c(\text{APP}_{d, \alpha}) \) is designed to be false for all \(d \neq c\).

Now that we have defined the formal basis necessary to develop a procedure for achieving a unique decision in a situation where different corporations are applying for different grants given by different institutions, we can go ahead with the definition of a formal evaluation of a given situation.

**Relation-based stability:** If \(d, e) \in R \) excludes \((d', e') \in R\) for \(c \neq c'\) and \((d, e) \in R \) excludes \((d', e') \in R\) for \(d \neq d'; \ d, d' \in a \) for some \(a\).

\[F' = <D, PHIL, R', v', APP> \] is a relation-based derivation of \(F = <D, PHIL, R, v, APP>\) iff \((d, e) \in R \) and \((d', e') \in R\) for different \(d, d' \in \alpha\) ; \((d', c') \notin R\) for all \(c' \neq c\) and \(R' = R - \{(d, e)\}\)
or otherwise \((d, e) \in R\) ; \(d \) is maximal in \(\{e \in \alpha, \ <e, k> \in R, <e, k'> \in R \text{ for some } k \neq k'\}\);
\[
\text{SIT} (c) * <1> = \min_{PHIL(\alpha)} \{\text{SIT} (e * <1>)_{<d, e> \epsilon R} \} \text{ and } R' = R - \{(d, e)\}.
\]

**Valuation-based stability:** at least one of \(\neg A\) and \(\neg \neg A \) is true at every element of \(U\) and if \(\neg \text{APP}_{c, \alpha}\) is true at \(d \in \alpha\) then \(\neg \text{APP}_{c, \alpha}\) is not true for all \(d' \in \alpha, d' \neq d\)

\[F' = <D, PHIL, R', v', APP> \] is a valuation-based derivation of \(F = <D, PHIL, R, v, APP>\) iff \(\neg \text{APP}_{c, \alpha}\) is true for \(d' \in \alpha; <d, e) \in R, d \in \alpha, d \neq d' \) and \(R' = R - \{(d, e)\}\)
or otherwise \((d, e) \in R; d \in \alpha; \neg A \) and \(\neg \neg A \) are not true for some \(A\) at \(d\) where \(d\) is maximal in a with this property; \(\text{SIT} (c) * <1> = \min_{PHIL(\alpha)} \{\text{SIT} (e * <1>)_{<d, e> \epsilon R} \} \text{ and } R' = R - \{(d, e)\}.
\]

A situation is **coherent** iff \((d, e) \in R\) for some \(d \in \alpha\) implies \((d, e) \in R\) for all \(d \in \alpha\).

A situation is **acyclic** iff there is no sequence \(d_1, d_2, \ldots, d_n, d_{n+1} \in R\) for \(d_1, d_n\) in the same \(\alpha\). The **application criteria** are **distinct** iff \(\text{SIT} (c) = \text{SIT} (d)\) implies \(c = d\).

**Theorem:** Assume that the situation of allocation of subsidies \(F\) is finite, **coherent**, **acyclic** and that the application conditions are distinct: all relation- and valuation-based derivation chains are finite and terminate with a relation- and valuation-based stable normal form \(F^*\).
Proof (Sketch): Reductions obviously terminate because of the finiteness of \( F \) and all resulting frames have relation- and valuation-based stability. If different derivation steps are possible at a certain stage, these derivation steps possess the diamond property (i.e. there are derivation chains to a common frame \( F^o \)).

Corollary: In \( F^o \) all grants are uniquely allocated if the number of applicants is greater or equal to the number of grants (formally: an institution allocates a subsidy to a corporation \( c \) if \( \Box \text{APP}_{c,d} \) is true in \( F^o \) at some \( d \in \omega \)).

4.2. The application to regional development programs (taken from [Baaz et al. 1993])

In our example we have three corporations A, B, and C which apply for grants of two institutions, INEM and EEC. The 5 criteria described in chapter 2 are denoted by the sets of elements \( C_i = \{a_i, b_i, c_i\} \), where \( a_i, b_i, c_i \) do express the actual position of A, B, C relative to the corresponding criterion \( i \). (\( \text{sit}(A) = \langle a_1, ..., a_5 \rangle \), \( \text{sit}(B) = \langle b_1, ..., b_5 \rangle \), \( \text{sit}(C) = \langle c_1, ..., c_5 \rangle \)).

Criteria \( C_1, C_2, C_3 \) are ordered by \( b_1 < a_1 < c_1 \), criteria \( C_4, C_5 \) are ordered by \( b_2 < a_1 < c_1 \).

All corporations are applying to both institutions; INEM has two grants to allocate, the EEC has one subsidy to give to one of the applicants. (INEM=\{g_1, g_2\}, say \( g_1 < g_2 \), EEC=\{g_3\}). For the INEM decision makers we assume a utilitarian philosophy, for the EEC we assume an impartial one (PHIL(INEM) = \( C_5 < C_4 < C_3 < C_2 < C_1 \), PHIL (EEC) = \( C_5 < C_2 < C_3 < C_1 < C_4 \)).

Our formal model takes the following form:

\[
F = \langle \text{INEM}, \text{EEC}, A^o, B^o, C^o \rangle, \text{PHIL}, R, v, \text{APP}>, A^o = \{A\}, B^o = \{B\}, C^o = \{C\}.
\]

\( R \) is given by the arrows, \( \text{APP} \) is defined in the obvious way. \( F \) is obviously coherent, acyclic and the application criteria are distinct. The numbers associated to the arrows represent their cancellation in a possible derivation chain. The arrows to which \( \langle f \rangle \) is associated are present in the stable normal form. Consequently \( \Box \text{APP}_{A, \text{INEM}} \) is true at \( g_2 \), \( \Box \text{APP}_{B, \text{INEM}} \) is true at \( g_2 \), \( \Box \text{APP}_{C, \text{EEC}} \) is true at \( g_3 \). Therefore INEM grants a subsidy to A and B and not to C (\( \Box \text{APP}_{C, \text{INEM}} \) is true at \( g_1 \) and \( g_2 \)) and the EEC grants a subsidy only to C (\( \Box \text{APP}_{A, \text{EEC}} \) and \( \Box \text{APP}_{B, \text{EEC}} \) are true at \( g_3 \)).
The result has the following properties:

a) $\Box \text{APP}_c \alpha$ or $\neg \Box \text{APP}_c \alpha$ is true at every $d \in \alpha$.

b) if $\Box \text{APP}_c \alpha$ is true at $d \in \alpha$, then it is not true at any $d' \neq d \in \alpha$, which means that the institution grants the subsidy to a corporation $c$ iff $\Box \text{APP}_c \alpha$ is true at $c$.

The derivations produce a normal form which uniquely determines all decisions of granting subsidies.

The advantage of Kripke-type frames (cf. [Kripke 1963], [Kripke 1965] and [Chellas 1980]) is that nested contradictory situations can be expressed (cf. [Baaz Quirchmayr 1987] and [Baaz Quirchmayr 1988] for modelling contradictory situations in legal procedures). Enlarged sets of truth values are useful for the representation of priorities without influencing the set of logically valid propositions determined in this context by an adequate two-valued modal logic.

5. Constructing the Rule Base

For constructing the rule base, we used HyperReasoner [Quirchmayr Galindo 1993] for our first prototype implementation. However complex the rules were, once their structure had been defined, it was comparatively
easy to represent them in Hyper\textsuperscript{Reasoner}. For analyzing decisions and for representing legal knowledge in a qualitative way, Hyper\textsuperscript{Reasoner} turned out to be the tool we needed, but it soon became clear that besides legal reasoning, our project partners did first of all need support of their administrative work. As the administrative component was written in Double Helix, a Macintosh-based graphically oriented database system, we consider migrating Hyper\textsuperscript{Reasoner} from Toolbook to Hypercard, which, taking into account that Hypertalk and Toolbook's script language are very similar, should not be too time-consuming. The alternative is to network PCs and Macintoshes in order to give Hyper\textsuperscript{Reasoner} access to data stored in the administrative component of the system, which will mean to provide the necessary conversion tools. As these tools already exist in a rudimentary form in Double Helix, file exchange can quickly be established with reasonable effort. Achieving real integration, meaning the direct access from Hyper\textsuperscript{Reasoner} to the administrative component, will of course demand a greater effort. Yet another strategy is to reimplement the administrative component, which does mainly consist of traditional form management, in Toolbook or dBase, an approach which probably is the quickest solution and leads to a really integrated environment.

6. Conclusion

The strategy of basing the theoretical model of the decision making process on Kripke-type frames and implementing a prototype in Hyper\textsuperscript{Reasoner} has shown that a sound theoretical basis as well as the integration of modern user interface-technology are both necessary for indicating new ways for coming up with a generation of expert systems which will hopefully be accepted by users. One key to success will certainly be the integration of office information, database, and expert systems with new generations of technology to overcome the isolation which legal expert systems do still suffer from.

References

[Baaz Quirchmayr 1987] M. Baaz and G. Quirchmayr, The Application of Mul-


