Artificial Intelligence and Law **2:** 293–314, 1994

Coupling Hypertext and Knowledge Based Systems: Two Applications in the Legal Domain

PAUL SOPER* AND TREVOR BENCH-CAPON**
*Department of Electronics and Computer Science, University of Southampton, Southampton, S09 5NH, England
**Department of Computer Science, The University of Liverpool, Liverpool, L69 3BX, England

(Received 3 February 1994; accepted 17 August 1994)

Abstract. Hypertext and knowledge based systems can be viewed as complementary technologies, which if combined into a composite system may be able to yield a whole which is greater than the sum of the parts. To gain the maximum benefits, however, we need to think about how to harness this potential synergy. This will mean devising new styles of system, rather than merely seeking to enhance the old models.

In this paper we describe our model for coupling hypertext and a knowledge based system, and then go on to describe two prototype systems which attempt to exploit this composite framework. The first application concerns animated hypertext which accords the text a central role whilst giving access to all the advantages of a knowledge based system. The second suggests how we can augment the hypertext by providing links which reflect the conceptual model of a knowledge based system in the domain, so as to provide a more structured traversal of the text.

1. Introduction

Hypertext is a relatively new technology which has swiftly become popular. The field of AI and Law has proved no exception to this trend, and many applications in this area now boast a link to hypertext. Typically this involves having hypertext as a back-end to a relatively conventional consultative expert system, as in [Vossos *et al.* 1993]. Hypertext, however, deserves a little better than this, since the possibilities it opens up are too rich for it to be confined to this limited role. The possibilities are hinted at in [Greenleaf *et al.* 1991] where the authors distinguish between different kinds of integration of a knowledge based system (kbs) and hypertext, and hint at some of the distinctive applications that the different kinds of integration will serve. Thus it matters whether the hypertext is a means of accessing the kbs, or whether the kbs provides access to the hypertext, since these different integration styles will meet different needs. It needs to be recognised that what is offered is not simply a means of one technology enriching another, as when the explanation facilities of a kbs are augmented by hypertext, but a set of new possibilities where the synergy between the two technologies can lead to new kinds of interaction, and hence can give rise to entirely new styles of application.

To reap the advantages of such a composite system, however, it is necessary to identify and think through these new possibilities. In this paper we describe two responses to this opportunity. One application, the animated hypertext of an advice leaflet described in Section 3, enables us to address a problem in which the text must be taken seriously and

given a central role. Our prototype shows how this can be done while maintaining all the advantages of a knowledge based advice system. The second application, described in Section 4, uses an interplay between a knowledge based system and a hypertext to augment the links of the hypertext and so structure the browsing of case reports according to the conceptual model expressed in the knowledge based system. This augmented link structure supports directed traversal of the hypertext and systematic retrieval of case reports.

We will first describe, in Section 2, our architecture for coupling the two technologies, and then describe each of the applications in turn. Our coupling is intended to impose as few restrictions as possible so that the model can be realised using a variety of existing shells and hypertext systems.

2. Knowledge Based Hypermedia Framework

In this section we describe a general framework in which a knowledge based system (kbs) and hypermedia system are coupled into a composite system. Our architecture differs from that of Greenleaf *et al.* [1991], who have built the various possibilities for integration into their shell. We wish to provide a looser framework, which will allow the kbs to be built in a logic programming style and which will not dictate any particular hypermedia system. The general motivation for developing this framework is to improve the performance of both hypermedia systems and kbs by loosely coupling them into a composite system. On the hypermedia side the presence of the kbs can both enhance the hypermedia and address some of the fundamental navigational problems of current hypermedia systems. Conversely the presence of the hypermedia can address some of the fundamental problems of user-system interaction associated with current kbs's. Further, the composite system may enable the development of novel intrinsically integrated application styles. The example applications in this paper will be concerned only with the special case of hypertext, rather than with hypermedia in general.

The architecture of the knowledge based hypermedia framework (kbh) is shown in Figure 1. An important point is that a loosely coupled architecture is assumed in which the hypermedia manager and kbs shell communicate by message passing. Note that the relation between the document base (the contents of the hypermedia) and the kb (the contents of the kbs) is not part of the system. It is this informal relation, however, that will underlie the purpose of the particular application being built. Our work on the kbh framework so far has concentrated on developing the features required of the kbs component in the context of such a composite system [Soper and Bench-Capon 1992], [Soper *et al.* 1993]. Because of the open architecture the resulting kbs component, which is being developed in a logic programming style, will be composable with suitable present or future hypermedia systems.

Briefly, a hypermedia system is a collection of documents, each a piece of text, a photograph, or an item in some other medium. These documents are supplied with a rich collection of inter- and intra-document links. Users are then free to explore the document base by following links under the guidance of the hypermedia manager. Much current research in hypermedia (see [Nielsen 1990] for a recent account) is concerned with introducing

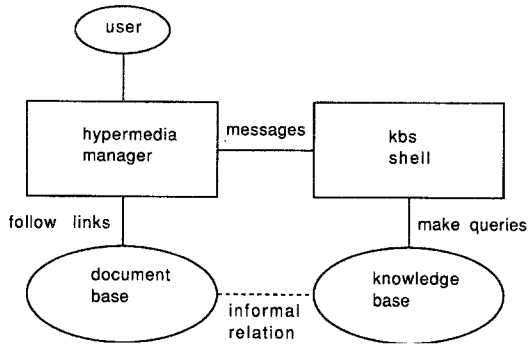

Fig.1. Architecture of knowledge based hypermedia.

additional structure and functionality to this basic system, for example: tours, navigational tools, active documents. The architecture shown in Figure 1 provides an alternative, and very flexible, way of introducing additional structure to hypermedia. The structure, which may concern a whole range of features from semantic support to text formatting, is modelled in the kb. The role of the kbs shell is to manage interaction, via selections (or buttons) in the hypermedia, so that the kb itself contains only declarative knowledge.

The basic idea of our approach is that a relatively domain independent kbs shell can support a wide range of applications with this architecture. The style of kbs we propose here is based on experience with logic databases [Gallaire *et al.* 1984]. The general motivation behind logic databases is that extended Horn clause logic should be investigated as a promising formalism for developing knowledge based systems, since logic provides expressive knowledge representation, a rigorous framework and the implemented tools of logic programming. In this style of developing kbs's not only is a clear distinction between the domain knowledge, the kb, and the kbs shell maintained, but also the shell itself is specified in declarative style. The prototypical shell of this type is query-the-user [Sergot 1983] which treats the user in a symmetric way to other knowledge sources. Many advances have been made using this approach, including incorporating explanations [Hammond and Sergot 1984], hypothetical reasoning [Eshghi and Kowalski 1989], conditional answers [Wolstenholme 1987], and temporal reasoning [Kowalski and Sergot 1986]. In spite of this, the use of such shells in practical systems has been limited (see the Proceedings of the First International Conference on Practical Applications of Prolog [PAP 1992] for an indication of current applications). It is possible that coupling logic based shells with hypermedia will allow more attractive systems to be built and so give rise to wider exploitation of their reasoning capabilities.

We have implemented the kbh framework using third party software for the hypermedia manager [Hutchings *et al.* 1991] and the kbs shell [Sergot and Cosmadopoulos 1990]. Implicit in the design of any particular application using this composite hypermedia-kbs framework is the relation between the contents of the document base and the contents of the knowledge base. This can take many forms depending on the purpose of the application. Our studies have centered on a UK Social Security Benefit, called Mobility Allowance.

The intended informal relation between the document base and the kb is as follows: the document base contains a hypertexted version of a leaflet concerning Mobility Allowance and the kb contains a logical representation of the underlying legislation.

In the following two sections we will now discuss the two example applications, based on the Mobility Allowance domain, which exploit features of the kbh framework. We begin with the animation of advice leaflets.

3. Animating Advice Leaflets

Expert systems have the potential to support a variety of tasks relating to the legal domain. Much attention has focussed on the provision of support for adjudicators and advisors, but another important task is that of informing lay members of the public as to their rights and duties. This is particularly important in the field of transfer payments-tax and welfare benefits – where public awareness of their rights and obligations are crucial to the smooth functioning of the system. In these areas it is important that citizens understand the effect of the law as it applies to them: welfare benefits must be claimed, and taxable income declared.

Currently the main burden of disseminating such information falls on advice leaflets of one form or another. These leaflets contain a simplified exposition of the law, and are often associated with a form, claim forms in the case of welfare benefits, or forms declaring income in the case of income tax. A further type of leaflet gives assistance when difficult choices are to be made, such as when people are to begin a period of working abroad and need to know how they should arrange their affairs with regard to tax. Often citizens will need to seek further advice, but without some understanding gleaned from the advice leaflet they will not know of this need for advice, nor be likely to assimilate advice when given. Given the widespread use of such leaflets in providing information to members of the public, we believe that a system which is intended to support the dissemination of such information must start by taking these leaflets seriously.

Two things are critical: first that the readers of the leaflet should gain some understanding of the area of concern, since they are discharging a legal action for which they must take responsibility, and second that they should be in a position to attempt to apply this understanding of the general provisions to their own particular cases. Where advice leaflets have been used in the past, they have normally been used only as a source of information to be restructured as the knowledge base of a conventional consultative expert system. This is less than satisfactory, however, since the oracular nature of such systems may leave users with an answer meeting their circumstances, but only an imperfect understanding of how that answer was arrived at. Such systems therefore undervalue the text. Moreover this lack of understanding militates against the ability of the user to answer correctly questions posed by the system, and there must therefore be a corresponding doubt in the conclusions reached by the system.

Our proposed solution to these problems is to use our composite kbh framework to animate these leaflets in a way which makes manifest the implications of the legislation for a particular case through the use of an underlying executable model. The resulting

system provides all the advantages of the traditional expert system, but in a way designed to promote understanding through giving due weight to the text.

3.1. PROBLEMS WITH ADVICE LEAFLETS

Before going on to describe the system we should briefly review some problems with advice leaflets which lead us to think that the leaflets require animation. When composing advice leaflets a number of difficult choices need to be made, which tend to reduce the efficacy of such leaflets. Among the major problems are:

Detail Problems:

Only a certain amount of detail can be included in a leaflet if it is not to become unacceptably long. This means, to take an example, that if it is important whether a person lives in an E.E.C. country, a Commonwealth country, or elsewhere, these are the terms that are likely to be used because to list all the individual countries would result in confusion. Not listing the countries, however, does require that the reader know which countries are members of the various organisations.

Thesaurus Problems:

Sometimes different people will typically associate a different term with some key concept used in the leaflet. Thus in the case of Sickness Benefit there is something called a "Doctor's Statement" which is often referred to as a "Medical Certificate", a "Doctor's Certificate", a "Certificate" and a "Sick Note". Clearly the leaflet needs to settle on one of these terms and to use it consistently. Since, however, misunderstanding will result if the reader fails to respond to the term used, this is less than satisfactory.

Overlap Problems:

Often the leaflet will need to refer to some other area of law in the course of its explanation. Duplicating this in the leaflet would make the leaflet unwieldy, and so there is usually cross reference to some other leaflet or leaflets. This is, however, frustrating when these leaflets are not immediately available.

Different Circumstance Problems:

Typically the law will need to cover a number of different circumstances; the married and the single, those with children and those without, the old and the young. Where different provisions apply to different groups, the result will be that much of the leaflet will be irrelevant to a given reader. Picking a way through the leaflet so as to find all and only the relevant material is a non-trivial task, and a reader may well get confused by irrelevant material, or miss something that does matter.

Use of the Information:

When they have read the leaflet the readers need to be able to go on to apply the information to their own cases. Where this is complicated – as it often is in such areas of law – they may either fail to apply some rule, or misapply some rule. Morever, when they go on to complete the associated form, they may well have difficulties in relating a particular entry of the form to the correct part of the document.

Some of these problems can be addressed by the conventional use of hypertext while others, especially the last two, can be addressed by knowledge based system support. However neither of these technologies on its own, nor the two in simple juxtaposition, can deal effectively with all the problems [Bench-Capon *et al.* 1991]. Let us now consider how we can use a composite hypertext and knowledge based system, in the form of the kbh framework, to exploit the complementary aspects of the two technologies.

One of the main problems with using a stand alone knowledge based system to impart information is the difficulty of managing the system-user interaction in a satisfactory way. A key problem is that users often do not have a well-defined question, or, if they do, that question is modified or transformed during the session. Also users frequently want to gain an understanding of the domain rather than answer a particular question. While some applications may lend themselves to a restricted system-user interaction, the general problem is a hard one of modelling human-like communication.

The difficulty of managing the two-way interaction of expert, or knowledge based, systems is in marked contrast to the situation for hypertext and hypermedia systems. The latter are essentially passive collections of documents, each a piece of text, a photograph, or an item in some other medium. The communicative function of each document is not problematic: it has been constructed by humans for humans; a well-written text, for example, will be intelligible to its intended audience. Furthermore in a hypermedia system users are free to follow any one of a rich collection of inter-document links. The only potential difficulties arise from the fact that there may be too much freedom so that the users may become disorientated, or otherwise distracted from the task at hand.

We propose the use of hypertext as a top-level interface to a knowledge based system (kbs). Entry points to the kbs are from buttons in the hypertext. In this way we hope to achieve four things: first, to accord the text its due prominence; second, to circumvent the fundamental difficulties of communicating with the kbs; third, to produce an enhanced hypertext which, by querying the kbs, can specialise the general understanding conveyed by the text to particular circumstances supplied by the user; and fourth, to have the ability to make the text available to clarify questions posed by the kbs to the user. We use the term 'animated hypertext' to describe such a composite system.

We will illustrate out ideas by describing a prototype implementation of animated hypertext for the Mobility Allowance, a benefit paid under the United Kingdom welfare system to people who are unable, or virtually unable, to walk. Our aim is to demonstrate the main features of animated hypertext, to explore its feasibility and to identify areas needing further research.

3.2. SOURCE TEXTS ON MOBILITY ALLOWANCE

As one would expect there are several types of leaflet explaining the Mobility Allowance legislation going into various levels of detail. For example there is a four page leaflet with a six page application form attached [Department of Social Security 1991] aimed at members of the public who wish to claim the allowance. There is also a guide to non-contributary benefits for disabled people [Department of Social Security 1990] with a nine page section explaining mobility allowance legislation aimed primarily at advisors. It references the relevant sections of the much more extensive legislation concerning mobility allowance: the Social Security Act 1975 (as amended) and the related Regulations and Schedules approved by the British Parliament.

Naturally hypertext affords a means of integrating the different levels of advice into one system and indeed integrating the text of the legislation itself. It is not our purpose at this juncture to produce such a complete system. We have attempted rather to prototype a system which demonstrates in a plausible way the novel features of animated hypertext. We have chosen the guide as our primary text for this purpose. Apart from being structured into a hypertext, the text of the guide has been left essentially the same. Concepts in the guide such as 'presence condition' have been associated with predicates in the kbs. The kbs is a logical model of the underlying legislation.

3.3. Description of the Hypertext Component

The primary text is the part of the guide leaflet [Department of Social Security 1990] describing mobility allowance. This has been hypertexted using a third party hypermedia and authoring system called StackMaker [Hutchings *et al*, 1991] implemented in HyperCard and running on the Mac SE/30 under System 7. The first document of the hypertext (Figure 2) indicates the main criteria which are taken into account in deciding eligibilty for the mobility allowance. The overall structure of the leaflet is reflected in the hypertext contents list (Figure 3): each leaflet sub-heading has been associated with a document heading.

It will be seen in Figure 2 that some words and phrases are emboldened. These are buttons. Clicking on 'where you live' gives a menu of links which can be followed. The menu is similar to that shown in Figure 6, but only 'expand' is available. If 'expand' is selected we arrive at the document shown in Figure 4. Other documents can be reached in a similar fashion by following 'expand' links.

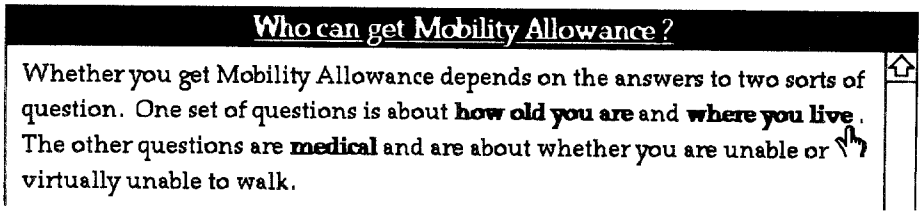

Fig. 2. First document in the hypertext.

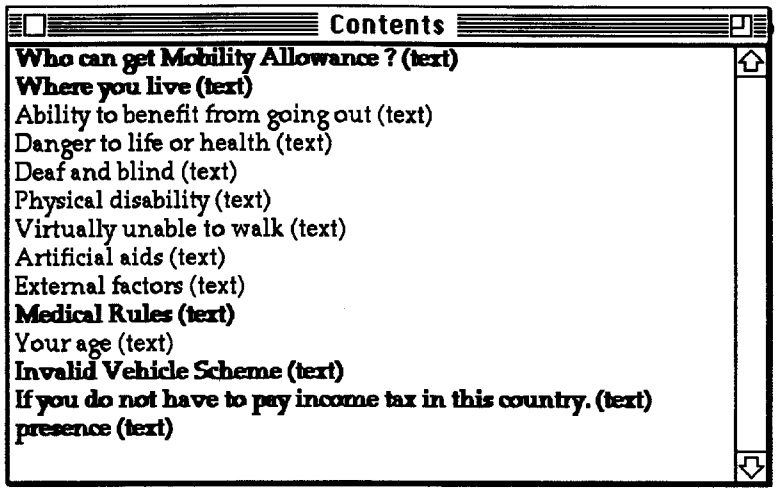

Fig. 3. Contents of the hypertext.

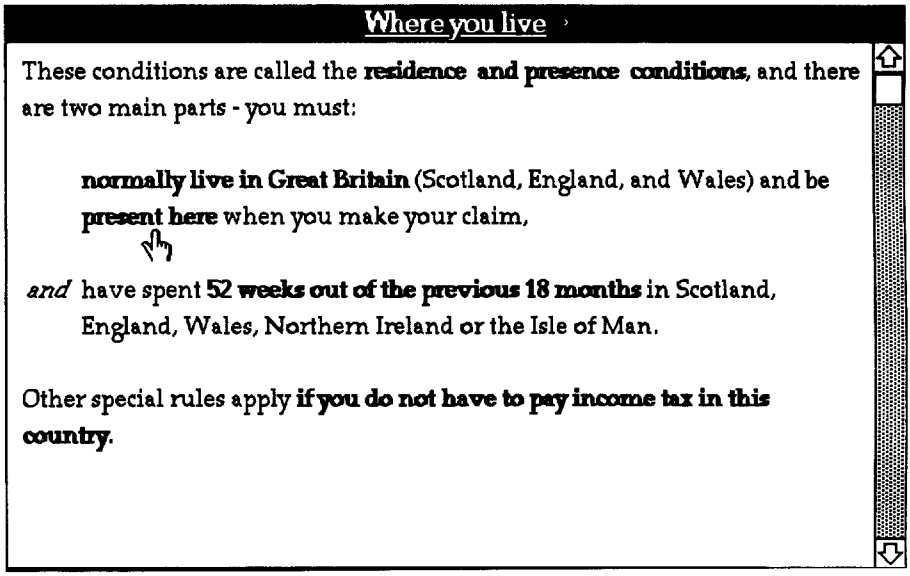

Fig. 4. Hypertext document.

3.4. DESCRIPTION OF THE KBS COMPONENT

The legislation concerning the mobility allowance has been modelled in Horn clause logic augmented with negation by failure [Sergot *et al.* 1990]. The resulting formalisation is executable

```
allowance (Person,Date,Duration,Amount):–
        claim_made(Person,Date),
        eligible (Person,Date,Duration),
        rate_allowance (Date,Amount).

eligible (Person,Date,Duration):–
        age_condition(Person,Date),
        residence_and_presence_condition(Person,Date),
        medical_condition (Person,Date,Duration).

considered_present_gb (Person,Date):–
        present_gb(Person,Date).

considered_present_gb(Person,Date):
        not present_gb (Person,Date),
        special_case (Person,Date).

special_case (Person,Date):–
        special_occupational_ (Person,Date).

special_case (Person,Date):–
        temporarily_abroad (Person,Date).

special_occuption (Person,Date):–
        in_or_staying_with_forces (Person,Date).

special_(occupation (Person,Date):–
        mariner_or_airman (Person,Date).
```

Fig. 5. Part of the knowledge base concerning residence and presence condition.

as a logic program in Prolog. Such direct execution in Prolog, however, would be too inflexible for our purposes. We require an interactive knowledge based system shell which can query the user for appropriate information not found in the knowledge base. Such information generally concerns the particular circumstances of the user. We have used a third party shell called Skilaki [Sergot and Cosmadopoulos 1990] implemented in MacProlog. At this stage we have adopted the Skilaki interface without modification in order to rapidly identify features which need improvement.

In order to use the Skilaki shell the knowledge base is expressed as Prolog rules. Figure 5 shows a fragment of the knowledge base selected for the purposes of exposition. The top-level predicate 'allowance' holds if an applicant 'Person' makes a claim for Mobility Allowance at time 'Date', they are eligible at that time for the allowance for a period of months 'Duration', and the weekly rate of allowance at that time is 'Amount'. The predicate 'eligible' holds if the three types of condition referred to in Figure 2 are satisfied. In the next section we describe an interaction with the kbs concerning part of the residence and presence conditions, specifically whether an applicant can be considered to be present in Great Britain. Figure 5 also includes some of the rules needed for this interaction, enough to give the flavour of the representation. The advice leaflet explanations of this fragment of the legislation appear in Figures 2, 4 and 7. Notice the prominence of the date of application in the formalisation, in contrast to the leaflet where it is largely implicit.

3.5. ANIMATED HYPERTEXT

We can now introduce buttons into the hypertext which are associated with predicates in the underlying kbs. From these buttons we can follow a link called 'find value' which initiates a query to the kbs. An example is the button 'present here' in Figure 4. Clicking on this button gives the menu shown in Figure 6. If 'expand' is selected we get more text as shown in Figure 7, but if 'find value' is selected a query is launched to the kbs, that is to the Skilaki shell.

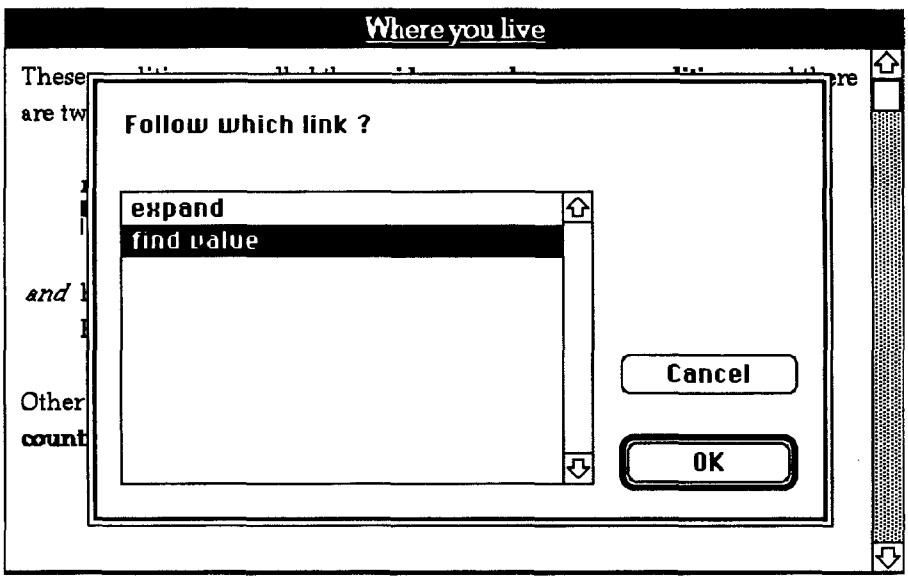

Fig. 6. Follow link menu.

The particular query launched will concern the associated predicate, in this case 'considered_present_gb (Person, Date)'. If we assume that prior interaction with the kbs has established that the applicant is Fred and the date of application is 26 February 1992 then 'Person' will be instantiated to 'fred' and 'Date' to '[26, 2, 92]'. The query launched to the Skilaki shell is shown in Figure 8, meaning 'Is it the case that Fred can be considered to be present in Great Britain on 26 February 1992?' The standard Skilaki query is shown, but note that this will not be seen by the user who will be thrown immediately into a dialogue with the kbs, starting with the query to the user shown in Figure 9. Figures 10–12 show the interaction which might occur if Fred was not in Great Britain nor serving in Her Majesty's Armed Forces overseas (or staying with a relative so serving) on the date of application, but was a mariner or airman on that date. Figure 12 shows the answer 'yes' to the original query.

Several comments are in order on the example interaction given. First, we have left the Skilaki dialogue boxes in their standard unfriendly form. An obvious improvement would

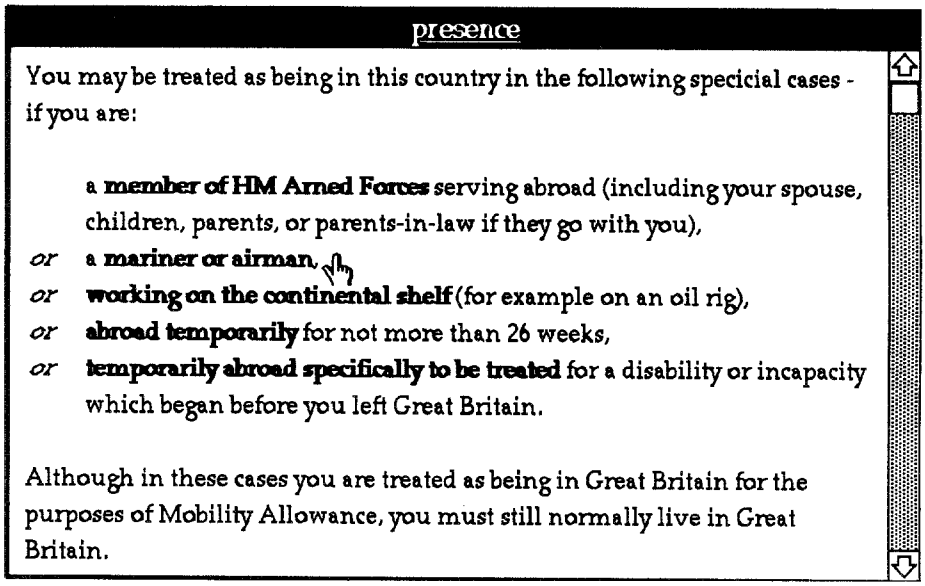

Fig. 7. Hypertext document concerning presence in Great Britain.

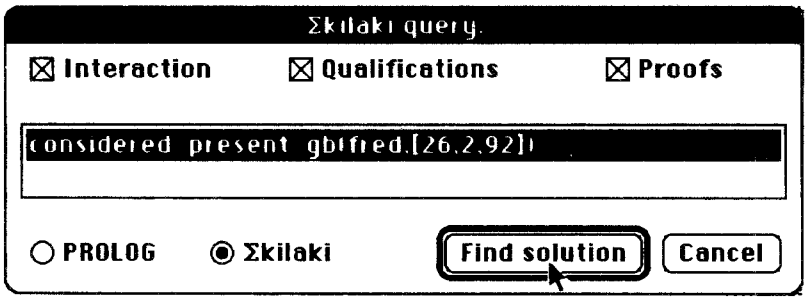

Fig. 8. Standard Skilakli query.

be to introduce natural language templates for these queries, re-expressing the predicate and eliminating internal variable names. Second, the dialogue with the kbs can be interrupted at any stage by the user returning to read more text and any information that has been imparted to the kbs will be remembered in subsequent interaction. In our framework the communication channel between hypertext and kbs is bi-directional, and so it is possible to follow a link from the kbs to a particular text selection in the hypertext. In this paper we have not explored such reverse direction links, but preliminary studies [Soper *et al.* 1993] indicate that they have many uses. The final comment we wish to make is that we have so far made no use of the ability of the Skilaki shell to handle conditional answers and defaults.

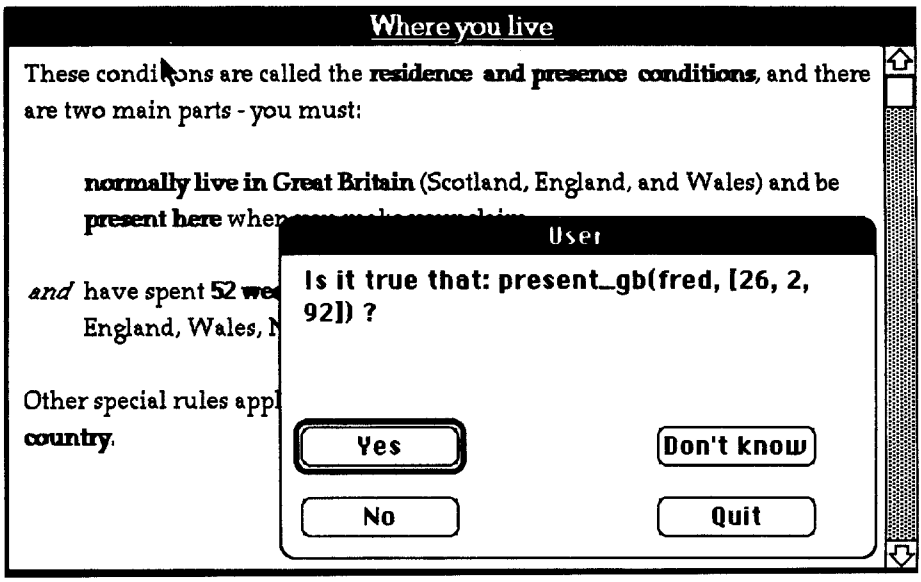

Fig. 9. Query to user: 'Was Fred in Great Britain on 26/2/92?'

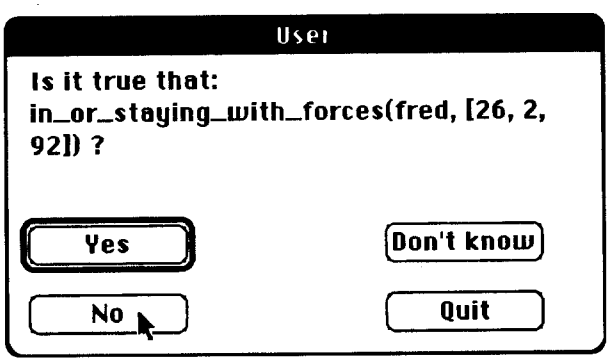

Fig. 10. Query: 'Was Fred serving in HM Armed Forces on 26/2/92?' (or staying with a relative who was so serving).

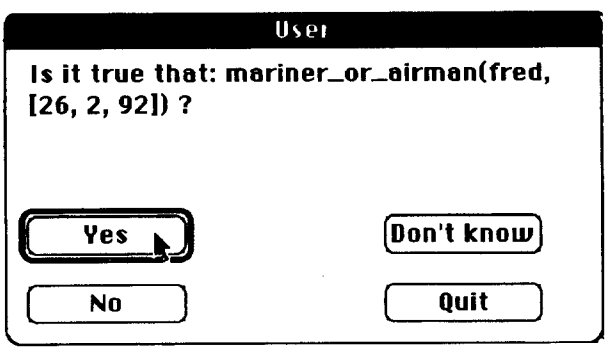

Fig. 11. Query: 'Was Fred a mariner or an airman on 26/2/92?'

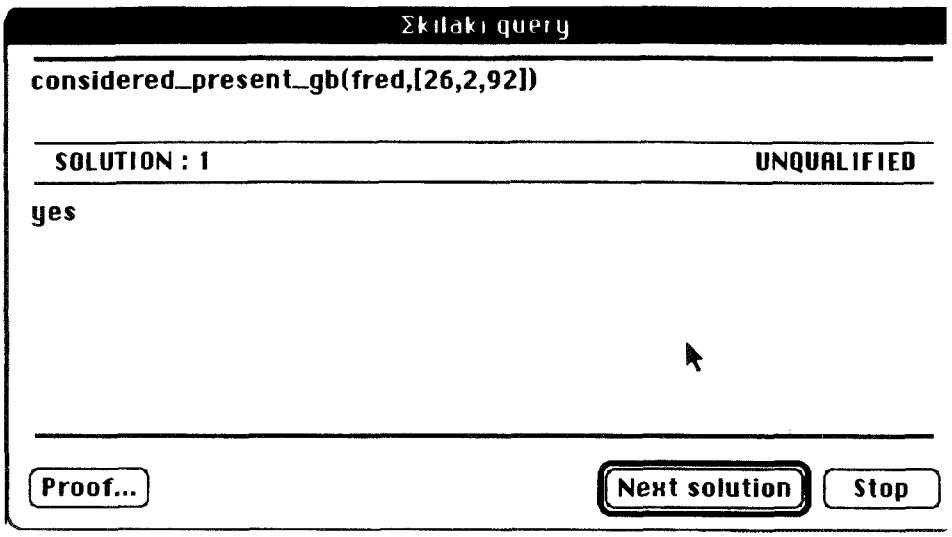

Fig 12. Standard Skilaki answer box.

3.6. SUMMARY OF ANIMATED HYPERTEXT

We have thus implemented a prototype animated advice leaflet in which a hypertext is used to interface to a kbs. The prototype clearly demonstrates the feasibility of our approach, and shows that the strength of a kbs – that it can provide answers to specific questions from the user – can be combined with the strength of hypertext – that it puts the user in control of the interaction allowing free exploration of the domain.

Some cosmetic improvements are needed: in particular modifications to make the standard interaction windows of Skilaki express questions and answers to the user in a style matching that of the text documents. Attention was not paid to this issue in the prototype since it can be simply effected through the use of natural language templates. Work is also needed to make the interface of the shell back to StackMaker cleaner: this is a straightforward implementation issue.

Obvious possibilities for further development include extending the hypertext to include different levels of leaflet that relate to the benefit, all of which can be interfaced to the same underlying kbs and incorporating texts and supporting kbs's relating to other benefits for disabled people, such as attendance allowance, to make the coverage of the system more comprehensive.

4. Information Retrieval

The legal domain contains a wide variety of documents: quoting a few examples, such as legislation, commentaries, case reports, guidelines given to those who apply the law, and information supplied to members of the public, will give a flavour of the diversity of the material available. Additionally, general principles, cases relevant to other areas of law, and other legislation can throw important light on an issue. Accessing this material can

present problems: in particular the reader will typically have a legal issue in mind for which clarification is needed, and the information available to the reader to bring to bear on the task will differ.

The conventional solution to this problem has been to use free text databases searched by Boolean keyword in context queries, but for many purposes this has been shown to be unsatisfactory [e.g. Bing 1987]. The main alternative has been conceptual retrieval [e.g. Hafner 1987, Dick 1992], but this approach has also encountered problems, and no large practical applications have been built. Recently hypertext has offered the potential for a different kind of solution: by hyperising the documents conceptual links can be shown to the reader, and the material found by browsing these links.

Where efforts at automated indexing have been made, as in the FLEXICON project [Gelbert and Smith 1991], it has been mainly based on word frequencies and similarities. Our method is based not on the words of a document, but on the way in which a case is processed by an adjudicator. For the problem remains that the words in the document may not be the best guide to the topics which they address. Legal problem solvers must move through a set of issues in order to resolve a case, and such issues can be – and are – stated in a variety of ways. Given this, how can the appropriate semantic links be created without the painstaking – and impracticably expensive – attentions of an expert?

In this section we describe a method which identifies the issues through the use of an underlying logical model of the pertinent legislation. The approach has its roots in the normative thesaurus work of Bing [1987], but it is adapted for use within a hypertext environment. This method – especially applicable to case law – works by running the case through the logical model, so that the issues relevant to the case are identified. These issues can then be used to annotate the hyperised text of the case, so that the facts of the case are available to a reader exploring that issue.

Our method again uses our general framework for coupling hypermedia and knowledge based systems. The knowledge base contains the logical model which structures cases in terms of the important issues, and the hypermedia contains the texts of the cases. In Section 4.1 we show how the knowledge based system can be adapted so that execution of the model provides the semantic links between like cases. An advantage of the approach is that it provides a natural ordering through which the notion of 'like' cases can be systemically explored by the reader. This section also contains illustrations of how the system functions by reference to the domain of the UK Mobility Allowance.

4.1. USING THE KBS TO SUPPLY SEMANTIC LINKS

Begin by recalling the intended use of the system described in Section 3. A user, perhaps a member of the public or perhaps an advice worker, wishes to obtain or give advice on a particular case, concerning say – "Is Fred eligible for mobility allowance?" The user enters the system via the hypertext and by reading the texts may be able to form a conclusion on this question from a general understanding of the texts. If the case is not straightforward, however, the user can query the kb by following links from the text. Generally this will lead to a dialogue, managed by the kbs shell, concerning the specific facts of Fred's case.

The shell will remember any information supplied and will request any further information needed. In this way the user can find out how the legislation applies to Fred's particular case and, if desired, the system's response to whether Fred is eligible for mobility allowance.

How is the above procedure adapted for information retrieval? Suppose that we want to enhance the mobility allowance system by making available relevant case reports. These are to be stored in the document base of the hypertext. We would like a user who engages in a dialogue with the kbs, for example someone considering Fred's eligibilty for Mobility Allowance, to be able to retrieve case reports which are relevant to (or like) the specific case being considered in the dialogue. We would also like a user who is considering an existing case report to be able to retrieve other relevant cases. The problem, as outlined above, is how to link new documents, the case reports, into the hypertext. Once appropriate links are established a means of information retrieval has been provided. The general idea of our approach is to use queries to the kb to generate semantic links for incoming case reports. This requires the agency of a human reader but obviates the need for the painstaking attentions of a legal expert.

In more detail the links for an incoming case report, say case33, are generated as follows. The user reads case33 and from a general understanding of the hypertexted advice leaflet decides that it concerns, say, eligibilty for mobility allowance. A link from the hypertext relating to eligibilty is followed thus posing the query, "?-eligible (case33)", to the kb. This leads to a dialogue between the system and the user during which the user answers questions according to the information in case33. If the query succeeds then it means that eligibilty is deducible, that is:

kb + answers (case33) |= eligible (case33)

In obtaining this result the kbs shell constructs a proof of eligible(case33), the proof being a successful branch of the search tree rooted at the original query. This is illustrated in Figure 13: the triangle represents the whole SLDNF search tree [Lloyd 1987] and the branch of this or-tree ending at case33 represents the proof. We use this proof as a label for case33, say label33, when storing the text of case33 in the document base. This is useful for two purposes: first, if several case reports get an identical label they can be retrieved together; second, if a user is consulting the system for advice and their case matches a stored case report the latter can be retrieved. The real power of the system, however, comes because there is a natural way to generalise the notion of 'like' cases based on the proof tree in Figure 13.

In Figure 13 consider a second case, say case56, and the corresponding query "?-eligible(case56)". Suppose the query also succeeds with a proof which branches from the previous proof for case 33 at node "a". As we move down a branch of the search tree a proof becomes progressively more specific. Conversely, the convergence of proofs 33 and 56 down to "a" means that their justifications coincide at a certain level generality. Thus by comparing the proof labels for two cases we can obtain a measure of how 'like' two cases are. In general, as we step up a branch in the search tree (a proof) from a given case we find cases which are 'like' the given case at successively higher levels of generality.

Let us take an example to see explicitly how an appropriate label is constructed. Suppose case33 concerns Joe's application for Mobility Allowance, and that Joe is a continental shelf worker who satisfies the age and medical conditions but is not present in Great Britain at the date of application. The kbs shell, as well as querying the user, keeps a trace of the proof of Joe's eligibility. This trace, which takes the form of an and-tree, is precisely the expanded form of the branch in Figure 13. It is shown pictorially in Figure 14. Now consider the case of Fred as described in Section 3.5 and assume that he satisfies the age and medical conditions in the same way as Joe. The proof of Fred's eligibility will be exactly the same as Joe's, in Figure 14, except that the argument will now be f=Fred and the node shelf_worker(j) will be replaced by mariner(f).

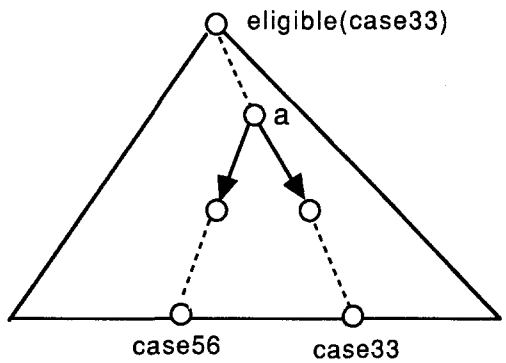

Fig. 13. Outline of search tree.

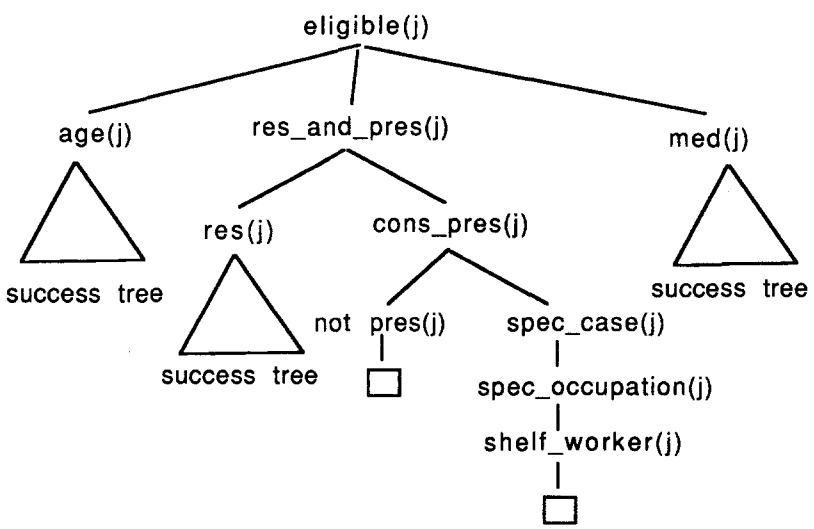

Fig. 14. Outline of proof tree for eligible(*j*), where j = Joe, using kb shown in Figure 5. Predicate names have been abbreviated and Date, Duration and Amount arguments suppressed.

In practice the proof trees stored by our kbs shell (a modified Skilaki shell) are the explanation traces normally provided by interactive kbs shells. We have also implemented as part of the shell preliminary versions of retrieval tools [Soper *et al.* 1993]. These tools allow a new case report in the hypertext to be associated with a proof-tree label: a fact is stored in the kb of the form 'label (CaseNumber,Label)' where 'CaseNumber' is a unique identifier for the case report and 'Label' is a proof tree of the type shown in Figure 14. The tools also allow relevant or 'like' cases to be retrieved. Given a case report with label L1 cases which are relevant or 'like' it can be retrieved via a predicate 'match (L1,L2,D)' which is true if label L1 matches label L2 at depth D, where L1 and L2 match if they have exactly the same predicate structure regardless of argument instantiations. For example if label33 is the label of Joe's case as shown in Figure 14 and label 56 is the label of Fred's case then 'match (label33,label56,D)' is true for D=4 and smaller depths, and false for D=5, ie an exact match. It is important to realise that argument instantiations are needed for the generation of labels, since they may effect the shape of the proof tree, but that they are often irrelevant for retrieval, as is the case for names for example. Our preliminary 'match' ignores arguments. Tools based on the 'match' predicate allow users to retrieve cases which match theirs in various ways: an exact match can be requested followed by stepping up the proof tree by decrementing the depth; or the matching cases can be ranked according to depth and the best five (if they exist) requested, followed by the next five and so on.

Several points need to be made about our scheme. First, the scheme as stated above deals only with eligible cases. If a case, say case45, cannot be deduced then it will lead to a finite failure search tree. Taking a similar example to those above, suppose Pete is a diplomat but otherwise similar to Joe and Fred. This type of special occupation is not known to the kb and so the query "?-eligible (pete)" will fail producing the finite failure

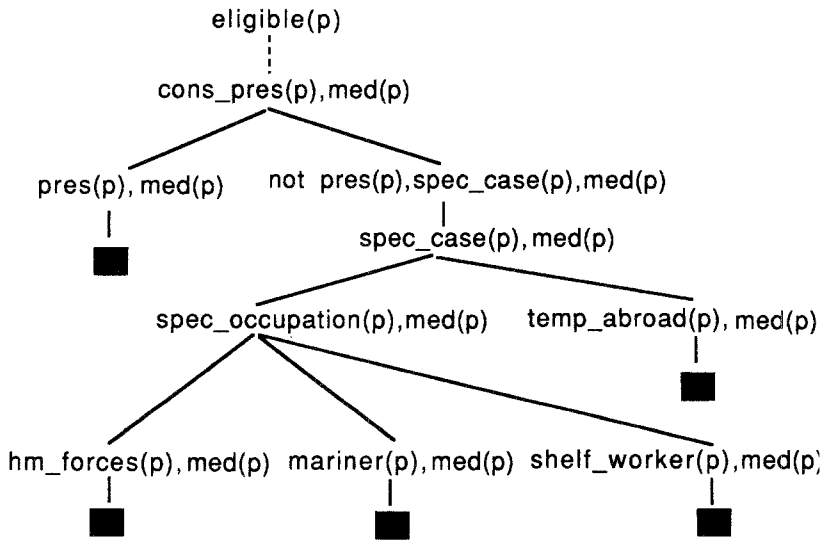

Fig. 15. Finitely failed search space for eligible(p), where p=Pete. (Dashed line indicated omitted section.)

or-tree shown in Figure 15. We now have many branches all of which fail. What should the label be for this case?

The finite failure tree means that, for the particular query, all possible ways of showing eligibilty fail, according to the kb. This manifests itself as failure at the leaves of the or-tree. In the example, Figure 15, there are four branches which fail because the following fail:

pres(p) hm_forces(p) mariner(p) shelf_worker(p)

We do not need to look at med(p) to be sure that eligibilty does not apply.

We could label cases for which eligibilty fails by the finite failure or-tree, but we prefer to select one of the failing branches as the label. We choose the deepest (longest) branch following an idea of Wolstenholme [1990] that this branch is more likely to contain the interesting reason for failure. Referring to the example there are three equal deepest branches. We select all of these, but our program recognises this by backing up to their common parent node, 'spec_occupation(p),med(p)'. In this way equal length branches acquire a single branch label. The more important point is that our selection has eliminated shorter branches. Is this a reasonable thing to do?

In the example, Pete fails to be eligible because he is not present in Great Britain at the time of application and he is not a special case. In fact he is not one of the special cases because he is neither temporarily abroad nor has he one of the designated special occupations. By eliminating the branch 'pres(p)' we are saying that Pete's case is less like cases in which the applicant is present in Great Britain than it is cases in which the applicant qualifies under a special occupation. Note that by backing up the tree other cases will be retrieved, including those with 'pres(p)'.

The key reason for selecting a single branch rather than the whole or-tree is so that failing cases can be related to succeeding ones – the latter always receive a single branch label. By doing this a case is considered to be 'like' another case, from the point of view of retrieval, on the grounds of structural similarity and not dependent on the result (eligible or not). Thus, in our example, Pete's case will aquire a label which matches Joe's and Fred's down to the level of 'special_occupation'. The elimination of shorter branches does not prevent these cases being retrieved at a coarser level of generality, but it does represent a particular measure of 'likeness'. We do not believe that we have exhausted the possibilities of such measures, based on proof trees, and the development of other possibilities is an interesting avenue for further research.

The second point we need to mention is related to the first. A case report may include the legal decision (eligible or not) reached in the actual case. There are four possibilities as regards eligibilty in a particular case: the system can find the case eligible or not; and the actual decision can be eligible or not. We take the same view of all four possibilities: than in determining 'like' cases it is structural similarity which matters and not the result (eligibilty or not). So, taking into account these points, the proof labels generated by the system provide our measure of likeness.

The final point we need to discuss concerns the structure of the logical model in the kb. For our scheme the logical model should reflect the important legal issues of the cases.

Figure 2 showed the top level hypertext of the Mobility Allowance leaflet. Roughly speaking there are three issues: how old you are; where you live; and your medical condition. We have structured our logical model so as to distinguish these issues at the top level [Soper and Bench-Capon 1992]. In the rather simple case of the Mobility Allowance legislation we believe it is possible in this way to structure the logical model so as to have useful retrieval properties.

In the above discussion we have considered the case where the knowledge base is derived from a formalisation of legislation. The consequence of this is that the retrieval is likewise structured by issues as they emerge from the legislation. There are, however, other perspectives that might be used. For example, an adjudicator employed by the Department of Social Security to decide Mobility Allowance claims will, in practice, work not directly from legislation, but from the manual issued by the Department. This may suggest setting about the adjudication task in a way other than that which might be expected from considering the legislation alone.

Formalisation of legislation is not, therefore, the only way to build a knowledge base in the legal domain. It is equally possible to represent the knowledge that can be derived from the adjudicator's manual, or elicited from a skilled adjudicator. If such a knowledge base were used instead of one based on legislation, the resulting issue structure would accord with this perspective, and the retrieval would be aligned with the expectations of a user performing this particular task.

Interestingly, we should note that it is quite possible to model more than one issue structure in the system, resulting in the generation of more links between the cases in the hypertext, and providing the flexibility to view the text from several perspectives.

4.2. SUMMARY OF INFORMATION RETRIEVAL

There is no shortage of information in the legal domain: the problem is to deploy it effectively. Representation in the form of knowledge based systems, whilst offering the advantages of executability and customisation to a specific set of case facts, often requires the support of the actual underlying texts themselves, and this gives rise to a variety of problems of selection and presentation. Document retrieval systems on the other hand often fall between conventional retrieval methods, which fail to take account of the conceptual structure of the domain, and conceptual retrieval methods, which require unacceptable depth of analysis and are also as yet imperfectly understood. Whilst hypertext offers promise, in that it provides greater flexibility of retrieval and supports conceptual association of text fragments, it is a paradigm directed towards browsing, whereby the user follows links in an undirected manner, and so is not entirely suited to tasks which require the thorough exploration of a text in order to solve some specific problem.

The method we have described shows how a kbs can be used to supply semantic links reflecting the conceptual structure of the domain encapsulated in the kb. These links will then provide a task directed structure to the traversal of the text. A strength of the approach is its generality. The kb affords a powerful representation for modelling structure, and once a knowledge based model is provided the proof tree method supports a uniform and

largely domain independent means of information retrieval based on the model provided. We do not wish to claim that the proof tree method is a universal panacea, however, and would emphasise that it can be used in conjunction with other methods of information retrieval.

5. Conclusion

We believe that there are several potential advantages to be gained from coupling hypertext and knowledge based systems which will help to address problems of information management in the legal domain. Our framework for effecting the coupling of knowledge based systems and hypermedia, restricted to hypertext in the legal domain, was described in Section 2. A point of particular interest is the architecture of the composite system since it clearly separates the domain dependent parts, the document base of the hypermedia system and the knowledge base of the kbs, from the domain independent parts. The latter form a framework which is expected to have wide application.

In Section 3 we described a system of animated hypertext, which has particular application where the need for the user to reach an understanding of the domain gives the text as important a role as the answer that might come from a kbs. In Section 4 we focussed on another particular advantage, whereby the navigation of the hypertext can be directed by the domain structure as expressed in a knowledge based system. In the legal context this has the particular virtues of traversing the hypertext in a systematic, issue based, manner and of allowing new cases to be assimilated into the system so that when they are retrieved they are placed in an appropriate context and with appropriate relations to other cases. Additionally there is the scope to allow for the co-existence of several issue structures, which can permit retrieval according to the several corresponding perspectives on the domain.

Building systems to provide effective support for tasks in the legal domain is not a simple matter. Nor is it, we believe, one that can be accomplished by either knowledge based systems or by information retrieval systems, in isolation. We have therefore described a way of coupling these two styles of system together in a synergistic manner, and have described two applications of such a composite system. These two applications are chiefly interesting because they depend entirely on the availability of both kbs and hypertext, and thus begin to explore the possibilities that are made available by thinking about how to exploit the composite system, rather than merely using one technology to enhance the other without rethinking the nature of the interaction.

Acknowledgements

This paper represents a consolidation of recent work. The animated hypertext work was introduced at JURIX 1991 in Rotterdam [Bench-Capon *et al.* 1991] and the original prototype described at DEXA 1992, held in Valencia [Soper and Bench-Capon, 1992]. The information retrieval work was the subject of a presentation at a conference "Towards a Global Expert System in Law", Florence 1993.

We would particularly like to thank Gerard Hutchings for his invaluable help with the StackMaker system. We have also benefited from many useful conversations with Wendy Hall and other members of the Multimedia Group at Southampton. We wish to thank Marek Sergot and Yannis Cosmadopoulos at Imperial College for their generous advice on the Skilaki system, and also Muhammad Pasha at Southampton for useful conversations. We are also grateful to the anonymous referees for their helpful comments.

References

Bench-Capon, T., Soper, P. & Coenen F. 1991. Animation of Advice Leaflets Using Hypertext and Knowledge Based System Techniques. *Legal Knowledge Based Systems. Model-Based Reasoning*. Koninklijke Vermande BV, Lelystad.

Bing, J. 1987. Designing Text Retrieval Systems for Conceptual Searching. In Proceedings of *The First International Conference on AI and Law*. Boston: ACM Press.

Dick, J.P. 1992. A Conceptual, Case Relation Representation of Text for Intelligent Retrieval. *Computer Systems Research Institute, University of Toronto*: Technical Report CSRI-265.

Eshghi, K. & Kowalski, R.A. 1989. Abduction Compared with Negation by Failure. In Proceedings of *The Fifth International Conference on Logic Programming*. Cambridge, Massachusetts: MIT Press.

Gallaire, H., Minker, J. and Nicolas, J.M. 1984. Logic and Database: A Deductive Approach. *ACM Surveys* 16(2): 153–185.

Gelbart, D & Smith J.C. 1991. Beyond Boolean Search: FLEXICON, A Text Based Intelligent System. In Proceedings of *The Third International Conference on AI and Law*, 225–234. Oxford ACM Press.

Greenleaf, G., Mowbray, A. & Tyree, A. 1991. Datalex Workstation – Integrating Tools for Lawyers. In Proceedings of *The Third International Conference on AI and Law*, 215–224. Oxford: ACM Press.

Hafner, C.D. 1987. Conceptual Organisation of Case Law Knowledge Bases. In Proceedings of *The First International Conference on AI and Law*. Boston: ACM Press.

Hammond, P. & Sergot, M. 1984. *APES: Augmented Prolog for Expert Systems*. London: Logic Based Systems Ltd.

Hutchings, G.A., Carr, L.A. & Hall, W. 1991. StackMaker: An Environment for Creating Hypermedia. *Department of Electronics and Computer Science, University of Southampton*: Technical Report, CSTR 91–11.

Kowalski, R.A. & Sergot, M.J. 1986. A Logic-Based Calculus of Events. *New Generation Computing* 4 (1): 67–95.

Department of Social Security (UK) 1990. *Leaflet HB 5*. London: HMSO, Dd 8240169 HSSSJ1300NE 277M October 1990.

Department of Social Security (UK) 1991. *Leaflet NI 211*. London: HMSO O/N 12550 JI396NE (HSSS) 790M March 1991.

Lloyd, J.W. 1987. *Foundations of Logic programming*. Berlin: Springer Verlag.

Nielsen, J. 1990. *Hypertext and Hypermedia*. London: Academic Press.

Proceedings of *The First International Conference on Practical Applications of Prolog*. 1992. London.

Sergot, M.J. 1983. A Query-the-User Facility for Logic Programming. In *Integrated Interactive Computer Systems*, eds. Degano, P. & Sandewall, E. New York: North-Holland.

Sergot, M.J., Sadri, F., Kowalski, R.A., Kriwaczek, F., Hammond, P. & Cory, H.T. 1990. The British Nationality Act as a Logic program. *Communications of ACM* 29 (5): 370–386.

Sergot, M.J. & Cosmadopoulos, Y.A. 1990. The Logic Programming System Skilaki: Design and Implementation. *Department of Computing, Imperial College*. Technical Report.

Soper, P. & Bench-Capon, T. 1992. Using Hypertext to Interface to Legal Knowledge Based Systems. In Proceedings of *DEXA 92 (in Valencia)*. Berlin: Springer Verlag.

Soper, P., Hall, W., Pasha, M. & Heath, I. 1993. Knowledge Based Hypermedia. In Proceedings of *The Workshop on Logic Programming Support Environments*. University of Edinburgh: September, 1993.

Vossos, G., Zeleznikow, J., Moore, A. & Hunter, D. 1993. The Credit Advisory Act System (CASS): Conversion from an Expert System Prototype to a C++ Commercial System. In Proceedings of *The Fourth International Conference on AI and Law*, 180–183. Amsterdam: ACM Press.

Wolstenholme, D. 1987. Saying "I don't know" and Conditional Answers. In *Research and Development in Expert Systems IV*, ed. Moralee, D.S. Cambridge, England: Cambridge University Press.

Wolstenholme, D. 1990. External Data in Logic Based Advice Systems. PhD Thesis. Department of Computing, Imperial College, London, pp. 107–109.