CONCLUSION

Drafters of legislation are normally expected to formulate the law as clearly and precisely as possible. This makes legislation an ideal application for logic programming. In particular, legislation which is definitional in character can often be formalized as rules in logic programs. When executed by an augmented PROLOG system such as APES, the formalization can be queried as though it were a database. The system in turn can explain and justify its conclusions in terms of the law as logic programs in APES has been applied in such areas as social benefits regulations and the British Nationality Act. These systems have many of the features associated with expert systems, but they can be regarded more usefully as precise, and executable, specifications of what the legislation tries to express. This suggests that executable formalizations of the law can aid the drafting process itself, and that the techniques have application outside the law for formulating and applying regulations in all kinds of organizations.

SELECTED BIBLIOGRAPHY


Toward a Rule-Based Representation of Open Texture in Law

Trevor Bench-Capon and Marek Sergot

ABSTRACT

Legislation which is definitional in character can often be formalized as rules in logic programs. When executed by an augmented PROLOG system such as APES, the formalization can be queried as though it were a database. The system in turn queries the user for additional information which it needs, and original legislation. At Imperial College the representation of law as logic programs in APES has been applied in such areas as social benefits regulations and the British Nationality Act.

We are interested in building computer systems which represent the law in computer-intelligible form, so that we can test how the law applies to specific cases, or provide systems that give advice on various legal problems. In this chapter we develop a proposal for a rule-based treatment of the open texture of
legal concepts. We begin by providing some background.

At Imperial College we have represented several fragments of legislation as logic programs. In general terms, we use some written legal source, such as a statute or a set of regulations, and express its provisions in a formal, logical language. The resulting representation is a "formalization" of the law (or more precisely, a formalization of the letter of the law) or a "literal interpretation" of the law. The formalization provides a logical model of the law which can be used in various ways. In particular, it is possible to derive logical consequences of the formalization by means of a mechanical theorem-prover. Deriving logical consequences of a particularly simple kind corresponds to executing the formalization as a program which tests the derivability of the law in specific cases. Deriving more general logical consequences corresponds to solving the implications of the legislation, or to solve other kinds of legal problems.

In many cases, the logic programming language PROLOG provides a simple theorem-prover which is powerful enough to execute such formalizations directly. In practice, it is necessary to use an extended form of PROLOG, such as APES, to handle any missing information which is required, and to provide explanations. APES itself is implemented in PROLOG.

This approach has been applied with reasonable success for several fragments of real legislation. Introductions to logic programming and PROLOG are available.

The British Nationality Act 1981, which defines the various categories of British citizenship, is an example of legislation which is suited to this kind of treatment. Many of its provisions have been expressed in a form of logic that can be implemented in PROLOG, and this formalization runs as a program in APES.

An obvious application of our formalization of the British Nationality Act is to determine whether in a given circumstance a particular individual is or is not a British citizen. To do so, we need to have access to facts concerning the individual as well as to other information not supplied in the Act. APES generates the appropriate queries to the user, and provides explanations, in particular how a given solution was obtained. In fact, the explanations produced by APES are logical proofs: the conclusion it produces are logical consequences of the rules contained in the formalization together with any extra information obtained from the user. The output which is computed by the program is not just the answer "yes" or "no" therefore, but rather the whole set of proofs that the program can construct.

Suppose then Peter is interested in discovering whether or not he is a British citizen. Our British Nationality Act program can help him, for several reasons. First, our program has no legal authority to decide whether Peter is a British citizen, but it can never give a false or a false answer. Furthermore, our program is able to tell Peter what conditions are necessary for him to be a British citizen, and what conditions are sufficient for him not to be a British citizen. It can also provide a set of formal proofs for its answers.

The British Nationality Act itself is flawed, that is, it conflicts with more general legislation concerning basic human rights, and that, whatever the Act says about the status of Peter as a citizen, it may not be perfectly acceptable to Peter's reasonable standards. In the case of Peter's interest in British citizenship, the program allows him to decide whether or not he has a reasonable claim to British citizenship. The program allows him to separate literal applications of the Act's provisions from other considerations. The program could be used, for example, in a preliminary analysis of Peter's chances of establishing his citizenship. Peter, or his advocate, might use the program to identify possible lines of reasoning to be expanded on in court. And even if Peter never needs to argue his case in court, an explanation in the form of a proof serves to indicate the proviso's of the Act which apply in his particular circumstances.

There is furthermore one set of circumstances, but only one, in which the British Nationality Act program might be used for deciding questions of British citizenship directly. Suppose, for example, that the Government of the United Kingdom became so impressed by the performance of our program that it appointed the program itself as the adjudicating body with authority to decide, for example, that Peter is a British citizen, but we can never give a false or a false answer. Furthermore, our program is able to tell Peter what conditions are necessary for him to be a British citizen, and what conditions are sufficient for him not to be a British citizen. It can also provide a set of formal proofs for its answers.
prescribed amount of income tax before crediting an employee’s bank account with the net monthly salary. The ability of such systems to generate proofs would also be useful, for example, claims are rejected by the machine. Indeed we could argue that criticism, so that we could question somewhat of the system’s working to program that pays us less this month than it did in the last.

Nevertheless, it is totally unacceptable in general that legal decisions should be taken by machine, whether the machine itself limits the usefulness of computer programs that are in law become more widely applicable if they are regarded not this way, they are tools for the analysis and solution of legal problems. The construction of proofs, with a view to identifying possible lines of reasoning, is the principal aim of consulting such a program.

The formalization of the British Nationality Act works reasonably well both as a program for applying the provisions of the Act, and as a system for helping lawyers to solve legal problems, is due to the nature of the legislation itself. The difficult even for lawyers to assess the implications for reached, therefore, by a mechanical, literal application of the Act are complex, so complex that it becomes specific individuals. Useful and nontrivial conclusions can be ruled. Citizenship of the place of a person’s birth, the citizenship of his parents, and Nationality Act, but they are isolated at the lowest level of more concrete items of information have been determined for a Act well suited to the treatment we have described. Not all legislation is like this however.

OPEN TEXTURE IN LAW

Let us consider as an example a fragment of the regulations relating to the award of a heating addition to Supplementary United Kingdom. The relevant legislation states that a person hard to heat, and that this addition will be at the higher rate of place of residence is extremely hard to heat. This could be formalized as logical implications, for example, in predicate logic:

\[ x \text{ is entitled to heating addition at normal-rate} \]
\[ \iff \]
\[ x \text{ has place of residence } y \]
\[ y \text{ is hard to heat} \]
\[ \text{and } \neg y \text{ is extremely hard to heat} \]

But would such a representation be capable of giving useful answers? Consider the problem of trying to establish that a person would be entitled to heating addition if his house is hard to heat or if it is extremely hard to heat, but the regulations are silent, and so too therefore is the formalization of the regulations.

The interpretation of the phrases “hard to heat” and “extremely hard to heat” is not spelled out in the legislation. They have left the application of the phrases to be determined by the appointed adjudicating authorities in the context of individual cases, and in consequence we must look beyond the regulations if we are to give useful advice on the application of the law.

At this point some might wish to argue that the legislation is flawed in that the legislators have failed in their duty to provide unambiguous legislation, and that what is required are additional regulations that will provide clarification. This is a tempting argument, but the temptation should be resisted. The reason for this may be illustrated by the consideration of another example drawn from the legislation concerning another of the United Kingdom’s Social Security benefits, Housewives Non-Contributory Invalidity Pension (now defunct).

A key provision of this legislation was that a person would be entitled to HNCIP if she was incapable of performing her normal household duties to a substantial extent. She was therefore not incapable of performing her household duties to a substantial extent, and was thereby entitled to benefit. Or one could hold (with the other United Kingdom commissioners) that a woman who was incapable of a substantial amount of her normal household duties was incapable of performing her household duties to a substantial extent, and was thereby entitled to benefit. Since a woman would be at the same time both capable of a substantial amount of her normal household duties and incapable of a substantial amount of her normal household duties, the reading of the provision was crucial. The provision was genuinely ambiguous, and it was subsequently amended so as to make it clear which interpretation should be taken (the less popular one in fact). But even when the provision had been thus disambiguated, the vagueness inherent in the word “substantial” remained. This important that the vagueness was no accident. Indeed, if one considers what might be involved in attempting to spell out what was meant by “substantial” in this context, one sees that the task would be impossible. The meaning is so dependent on
individual circumstances, which inevitably will be widely disparate, that no legislator could hope to foresee every possible circumstance and make provision for it. There is no alternative to allowing decisions to be taken on the basis of individual facts. And precisely similar considerations apply to "normal household duties" and "hard to heat," and many other phrases in legislation.

It is at this point that we must reintroduce the notion of open texture. A concept is said to be open-textured if its extension is not determined for all cases in it. Thus when we are asked to apply such a concept in whether or not the concept is applied. Not that its application seems arbitrary, for if this were so, then the concept would have no meaning as would make no difference. Rather there is whether the concept applies in a particular case. Although the extension is not fixed in advance, the method for its determination is obvious: the applicability of the concepts is determined by the appropriate courts or other adjudicating authorities.

The notion has been found to be useful in the field of jurisprudence. Hart argues that legal concepts are incurably open-textured. In cases of legal disputes, say, about the meaning of a particular word in a particular section of the law, there will be no unarguably correct answer until the case has been brought to court and a judge has decided what that word should mean. This is a simple result of the impossibility of the legislator being able to envisage at the time he passes the applicable law. Indeed, a law is passed and its interpretation is left to the courts, and as time passes a body of case law will be built up that will supply precedents for other cases. But such precedents will not be exhaustive. It will always be which no precedents have been set, and such precedents will be able to be overruled. Thus in any hypothetical case one cannot say what the outcome should be, although legislation and case law may provide guidelines. Until the case has come to matter. Legal concepts exhibit open texture: cases that never come to court can never be said to fall either within or without the extension of the concept. There is an obvious similarity between this open texture of legal concepts and ordinary vagueness. The difference lies in the fact that in legal cases we may be forced to make decisions one way or the other, in a way in which we are not in the case of the judiciary to provide arbitration when and if we require it. Even then it must be remembered that there is no provision to arbitrate in hypothetically incurable cases. Open texture in the law may be seen as vagueness plus a machinery for making a decision when one is required.

Hart's views, although persuasive to many philosophers of law, have not commanded universal acceptance. Notably, among his opponents has been Dworkin. Dworkin argues that there is no open texture because any case, no matter how abstruse, can be decided by the proper process of legal reasoning and argumentation. When a judge decides, his choice is not free but constrained by what is the fact of the matter. It follows course that a judge may decide wrongly (although in practice it will be impossible to overturn his decision if he represents the highest level of appeal).

The controversy seems to us to be of little practical importance. There is no conflict in what processes the judge should use in coming to his verdict but rather whether he is disregarding or creating legal truth. Although philosophically nice, the distinction need concern those interested in formulating legislation as little as the truth of Platonist mathematics should concern someone writing a program to perform some number of calculations. We can proceed as if legal concepts did have incurable texture, because a truth that may not exist in no more assistance than a truth that does not exist, especially since the processes of discovering and creating the required truth are likely to be identical.

We suggest therefore that the concepts of law are, or may be taken to be, incurably open-textured. Case law closes the texture gradually, but only in the circumstances of a real case. Moreover, any decision may or may not constitute a precedent, and any precedent set in this way will always be defeasible.

We now discuss ways in which we might seek to handle such concepts. We can imagine a computer system, and perhaps later some programs, which interpret paragraphs of text and the like for, say, a legal decision maker, and thus avoid some of the vagueness. We can proceed as if legal concepts did have incurable texture, because a truth that may not exist in no more assistance than a truth that does not exist, especially since the processes of discovering and creating the required truth are likely to be identical.

We suggest therefore that the concepts of law are, or may be taken to be, incurably open-textured. Case law closes the texture gradually, but only in the circumstances of a real case. Moreover, any decision may or may not constitute a precedent, and any precedent set in this way will always be defeasible.

We now discuss ways in which we might seek to handle such concepts. We can imagine a computer system, and perhaps later some programs, which interpret paragraphs of text and the like for, say, a legal decision maker, and thus avoid some of the vagueness. We can proceed as if legal concepts did have incurable texture, because a truth that may not exist in no more assistance than a truth that does not exist, especially since the processes of discovering and creating the required truth are likely to be identical.

We suggest therefore that the concepts of law are, or may be taken to be, incurably open-textured. Case law closes the texture gradually, but only in the circumstances of a real case. Moreover, any decision may or may not constitute a precedent, and any precedent set in this way will always be defeasible.
panel of experts from the appropriate government department, we could consult the explanatory documentation which is issued to the adjudicating officers, and we could assess the decisions made in all previous cases of this sort. Eventually we could propose a model of what "hard to heat" and "extremely hard to heat," or he can choose to reject it, in situations where the circumstances of some particular case have clearly been overlooked. In a later section, we shall suggest how this approach can be modified to give a more useful system. For now, we stress that approximation merely avoids all the problems.

We do find vague concepts extremely useful, which is why we have so many of them. It is often the case that something vague is precisely what we want to say, so that no attempt to approximate our statement in terms of sharp concepts will maintain the truth of the statement. To adopt approximation in general would be to restrict our powers of expression to an unacceptable degree. Moreover, and this is the fundamental objection, if we believe that legal concepts generally open-textured, then there is no right answer. Any translation into approximating sharp concepts would suggest that there is: the nature of open-textured concepts has not been represented at all.

UNCERTAINTY, PROBABILITY AND FUZZY LOGIC

It is sometimes suggested that the attraction of expert systems can be attributed to the uncertainty of the knowledge they contain, and the ability to cope with this uncertainty. The open texture of legal concepts is often confused with uncertainty, and sooner or later it may be suggested that the techniques for handling uncertainty in expert systems could be adapted to a representation of open-textured concepts in law. We shall argue otherwise.

Typically, an expert system associates some kind of "certainty factor" with every rule in its knowledge base and with every item of data which the user has supplied. The system's " inference engine" then manipulates these various certainty factors according to some calculus to arrive at an estimated certainty factor for one or more of its conclusions. We do not give a more precise description since details vary between implementations. We shall argue in any case that the approach is inappropriate in principle both for treating vagueness in general and the open texture of legal concepts in particular.

This is not to suggest that expert systems and their techniques for handling uncertainty have no role to play in legal applications. One could imagine, for example, including certainty factors in the expert system which approximates the concepts "hard to heat" and "extremely hard to heat." It may be useful to include some measure of confidence in the accuracy of such an approximation. Rather, we shall argue that this treatment of uncertainty is not particularly suited to the representation of open texture.

Although details will depend on the particular expert system, certainty factors are always based, usually very loosely, on probability theory, on fuzzy logic, or on some combination of the two. When the user of a typical expert system is asked if a claimant's house is hard to heat, he will answer "yes" or "no" or "maybe" or something similar. Now this factor 0.3 may indicate that the house is hard to heat with probability 0.3, or that it is 0.3 true that the house is hard to heat, or something in between.

Those who implement most expert systems do not specify exactly what these certainty factors mean. Nevertheless, whether a particular version of uncertainty in a particular expert system is based on fuzzy logic or more on probability theory does not concern us here. We shall argue that neither of these subjects is relevant to the treatment of vagueness.

We take probability first and consider an example to illustrate the point. We could say that our neighbours' son (whom we have never met) is probably tall on the basis that his parents are tall. We can do this without taking any account of the vagueness of "tall." Problems with the vagueness of "tall" one only arises when we are confronted by the neighbours' son and now we must try to decide whether he really is tall or not.

We have exactly the same situation when we need to decide whether a claimant's house is hard to heat. We might have statistics which tell us that 10 percent of houses are hard to heat, and that 40 percent of houses in the claimant's town are hard to heat, and that 90 percent of houses in his street are hard to heat. These figures might allow us to estimate the probability that this claimant's house is hard to heat, or whether this particular claimant is entitled to the heating addition.

Probability theory is without doubt one of the most powerful and important components of any decision support
system, and legal decision support systems are no exception. Given a client that has chances of establishing membership in fine is low, or whatever. But none of this has anything to do with the likelihood he will escape a parking with vagueness or with its representation. In any case, faced with deciding how to answer the question of what is the claimant’s house hard to heat? we still have no idea what the question means in practice. The ability to give an answer anywhere between 0 and 1 gives us a freedom that we have no idea how to use. Our fuzzy logic is no more than (over)
sophistication of the approximation approach, that it may give good results in certain very special applications, but that its philosophical basis is uncertain generally and very uncertain when applied to open-textured legal concepts. Both the appearance of precision and the appearance of generality are spuri-

Examples

The examples approach differs from the previous treatments discussed in that it is primarily designed to leave the actual
decision making to the user of the system. The basic idea is that the system contains a database of examples together with the
truth profile of the examples. The examples are either
stereotypical cases, or actual cases that have been decided.
When presented with a new case for decision the system will
attempt to match the case under consideration with the examples
in its database to extract those which appear to be most
similar. If the matching examples all point the same way then
the decision will be clear cut; if not it will be up to the user
to decide which precedent he wishes to follow.
There is obviously the problem of finding a method for
storing the examples themselves, and it is difficult to imagine
how any general method will represent adequately the infinite
values of a concept or of a legal concept in a legal case. But
we have encountered the suggestion that the problem
becomes tractable if attention is restricted to some specific
open-textured concept ("hard to heat") would be a typical
example, for one could surely draw up a list of attributes for a
particular concept, which would be large enough and flexible
enough to capture all the possible variations whilst still
remaining manageable. Now there is an immediate objection to
this proposal. Problems of open texture arise precisely because
it can never be predicted what features of a case will turn out
to be relevant. The objection might be countered by proposing
an altogether more sophisticated representation, in which the
list of possible attributes is not now fixed but instead dynam-
ically as cases arise which cannot be fitted into the existing
framework.
Leaving that aside, there is the related difficulty of
producing a good matching algorithm and defining what is meant

value for the statement that the law applies to the situation.
As legal judgments can only have the truth value "true" or
"false" there would need to be a further rule to determine which
rule of combination is to be adopted. This would be the rule
of "true" and "false" which is one of the properties of a fuzzy
logic treatment, and one that legal concepts blatantly fail to
fulfill.

by "similar." The simplest proposals would associate some kind of weight with each attribute as a way of indicating which sort of features of an example were seen to be most relevant. In this calculation of weighted sum: the examples with scores above a certain threshold are taken to be the most "similar" to the situation at hand. However, no matching algorithm can be considered acceptable unless it is based on some coherent model of what it means for cases to be "similar."

What we are proposing is a system which does not reason directly from examples, but reasons instead with the general rules that these examples are able to generate. Examples are stored, not only for the purpose of explaining and justifying the validity of general rules. This proposal can be viewed simply as a way of implementing the database of examples, and as such it has some immediate advantages. We now have a framework, a conceptual framework of what "similar" means; two cases are more similar if they are both instances of some abstraction; in other words, if the same rule applies to them both. Of course this merely shifts the problem to defining what is meant by "appropriate abstraction." We shall remark on this in a moment. Notice that an explicit matching process is now unnecessary. (Alternatively, we could think of the matching algorithm as distributed through the database, so that every example in effect contains its own specialized method of matching.) Finally, because general rules express the reasons for a decision, they give some indication why their particular precedent should be followed in preference to another.

To automate this process entirely would require a computer system that can induce general rules from specific examples. Such systems do exist in certain restricted domains, although they do not seem to be capable of handling the range of examples we can expect to offer in the near future. At present the system is not a practical proposition, therefore, at least not for the foreseeable future. Moreover, we have to ensure that all the derived general rules manage to capture an "appropriate" abstraction, and not merely some more specific one. This would be well beyond the capabilities of current techniques. Nevertheless, there is a practical solution which we can adopt for the present, and one which recognizes the difficulties involved. We can insist that a suitable general rule is formulated with every new example to be stored. This throws the burden of constructing the "appropriate" abstraction on the constructor of the database. In fact, this proposal is very different from the weighting of attributes which we mentioned earlier. Attributes we weighted to gain some flexibility and to indicate which of the features in a particular case are considered to be most relevant. What we are actually proposing is that the constructor of the database should be allowed to make this information explicit, by formulating a specific general rule, instead of representing it implicitly by choosing the way that he weights the attributes. This would insist that formulating a rule, tentative or not, will always be more difficult than weighting attributes in an example. This is not necessarily so. There is always the possibility that attributes may be more difficult to weight than another, and also never have been anticipated. In the simplest of domains, this complication is unlikely to arise, it would be just as likely to regard the assignment of weights as an alternative way of describing a general rule and storing it in the database.

There remains the problem of discovering the rule for the appropriate abstraction. In practice, this may not be as difficult as it seems, particularly for applications in law.
Cases in law are recorded and their decisions are documented. If the decision which is taken in a particular case is intended to be used as a precedent then some justification for the decision must be made. In other words, the justification itself will already contain the ingredients of a suitable general rule. In other circumstances, when the adjudicator is reluctant to formulate the reasons for his decision, an expert commentator will be required to propose a tentative rule which attempts to explain the decision. Of course, rules which are generated in this way do not have legal authority. They may be too general or too narrow, and they may conflict with rules derived from other examples. In the next section we develop this idea, because it applies to the treatment of open texture. For now we support our proposal by referring to the way in which legal problems are generally solved in practice.

Suppose we are given the problem of deciding a particular case. We could seek a solution by first compiling a list of those previous cases which seem to resemble the case we have under consideration. We could then examine each of these cases in turn, to estimate how closely they actually correspond to the facts of the new case in question. Eventually we would identify the one with the best apparent match and its decision could then be applied. This may be exactly the way some lawyers approach the problem. It corresponds quite closely to the process of consulting examples in a database. But many legal scholars would suggest that this is the wrong approach altogether.

The method of legal argument, or at least of presenting a legal argument, is to propose a rule of law that lies in the narrative in question; and then to cite previous cases, not because they have some passing resemblance, but because they can be used to persuade that the proposed rule of law does actually exist. It is considerations such as these that motivated our proposal for using rules instead of examples. What we need to do now is consider how this proposal adapts the treatment of open texture.

A TREATMENT USING RULES

What we have with an open-textured concept is something like an upper limit where the concept can be applied with confidence, a lower limit where the concept can be denied with confidence, and a grey area in between. What is now crucial is that every context that we adopt toward this grey area. This in turn is influenced by the nature of the computer system we are attempting to build.

Most of the treatments that are commonly proposed suggest how we might eliminate the grey area altogether by coming to a decision of some sort for the objects that fall within it. What is happening here is that vagueness is identified with a situation in which the Law of Excluded Middle is not applicable. Some method is devised that enforces this law, and then ways of resolving the problems that flow from this are suggested. But we have already argued that for open-textured concepts at least, there is no fact of the

matter before a proper decision is made. This suggests that enforcing the Law of Excluded Middle is not the right approach. The treatment that we are about to propose is about the exploitation that we have already argued that, for open-textured concepts are cases where both the assertion and the denial of the concept may reasonably be thought to be true; and that the way we might have for thinking this that are of paramount importance.

We can illustrate this idea by considering again the process of matching examples. In consulting a database, we are interested in an approximating concept which is sharp instead of the vague concepts are cases where both the assertion and the denial of the concept may reasonably be thought to be true; and that the way we might have for thinking this that are of paramount importance.

RULE-BASED REPRESENTATION OF OPEN TEXTURE

We can illustrate this idea by considering again the process of matching examples. In consulting a database, we are interested in an approximating concept which is sharp instead of the vague concepts are cases where both the assertion and the denial of the concept may reasonably be thought to be true; and that the way we might have for thinking this that are of paramount importance.
of relevant experts, and what it represented was their collective opinion. Thus in constructing it, we would have been concerned mainly with arriving at some consensus as to what these vague concepts mean. In the case of disagreement about some specific point, we would have been forced to reach some compromise before being able to incorporate a suitable rule in the system. What we are proposing here is in direct contrast. In cases of disagreement where one expert's opinion conflicts directly with that of another, we now incorporate both interpretations of the law, provided of course that both experts can produce good reasons to support the claim that their particular interpretation is the right one to take.

Before pursuing these ideas further, there is a technical objection to our proposal which should be considered immediately, because it is well known that a logical system of rules which is already inconsistent can never be used to derive meaningful consequences. We can describe how this objection can be overcome, and at the same time indicate how such a system might be constructed and used in practice, by considering a simple instance of imprecision in the law, drawn from the provisions of the British Nationality Act 1981 and its formalization in logic.

The first clause of the British Nationality Act 1981 states that:

1.1-(1) A person born in the United Kingdom after commencement shall be a British citizen if at the time of birth his father or mother is (a) a British citizen; or
(b) settled in the United Kingdom.

This clause can be formalized fairly naturally as a logical implication. Elsewhere in the Act it is important to know the section by which an individual becomes a British citizen and the date on which he does so, so following the formalization described elsewhere.

For every individual x and z, date y, and section of the Act 21,

\[ x \text{ acquires British citizenship on date } y \text{ by section 1.1 if } \]
\[ x \text{ was born in the UK and } y \text{ is after or on commencement and } z \text{ is a parent of } y \]
\[ \text{or } (z \text{ is a British citizen on date } y \text{ by section 21 and } y \text{ is settled in the UK on date } y) \]

In practice, the rule for section 1.1 was more complicated than the one we have shown. There is also the need for a rule which relates to the possession of British citizenship to its acquisition. What is important for our present purposes is to notice that the formalization of section 1.1 makes an assumption, second, which is not stated explicitly in the Act: that an individual who acquires citizenship by section 1.1 does so at birth. Now this may be a very reasonable assumption, and it may be exactly what the draftsman intended. Nevertheless, it is not

what he wrote.

In particular we might argue that citizenship imposes duties as well as granting rights; that duties cannot be imposed on a minor; and therefore that British section 1.1 is not acquired at birth, but only when the individual reaches full age. This may or may not be a reasonable way of thinking (later sections of the Act make it clear that a minor can be a British citizen), but if we accept its argument for the moment we are led to an alternative formalization of section 1.1:

For every individual x and z, date y1 and y, and section 21,

\[ x \text{ acquires British citizenship on date } y1 \text{ by section 1.1 if } \]
\[ x \text{ was born in the UK and } y1 \text{ is after or on commencement and } \]
\[ y \text{ is a parent of } y1 \]
\[ \text{or } (z \text{ is a British citizen on date } y \text{ by section 21 and } \]
\[ y \text{ is settled in the UK on date } y) \]

To claim now that this second formalization is not what the draftsman intended is really to miss the point. The fact remains that on the face of it this second interpretation of 1.1 is just as good as the first. Until a case is decided the correct interpretation we have two formalizations of the British Nationality Act. Let us call them BNA1 and BNA2 respectively. Let us suppose further that BNA1 and BNA2 differ only in their treatment of section 1.1 because the persons who are responsible for the formalizations agree on the reading of every other section. If we genuinely cannot decide which of the two versions more accurately represents the legal state of the law, then we have no choice: we must incorporate both versions in any computer system which attempts to give advice on British citizenship. We have a system which computes two distinct opinions of the law in two separate formalizations. If we ask such a system to help establish a particular individual's citizenship on some given date we may get conflicting advice. In most cases the two formalizations and BNA2 will agree. In other cases, where the timing is more critical, version BNA1 will conclude that the individual is a British citizen, and BNA2 will conclude he is not.

To make a decision for such an individual (let us call him Peter) we need to compare the arguments for these two conflicting conclusions. What we have are two similar, but separate, formalizations, both of which express their own particular interpretation of the Act. We also have, included in the formalization, a set of rules (say P-data) expressing the informal cases specific to Peter (where he was born, his date of birth, and so on) which would have been added on his behalf. If the conclusions had been derived by a suitable theorem prover, we would also have two proofs: one that demonstrates why Peter's British citizenship is a logical consequence of BNA1 together with P-data; another that demonstrates that the denial of this is a logical consequence of BNA2 together with P-data.
Now the examination of these two proofs is what allows us to reach a decision. The proofs are guaranteed to be valid, but one may provide a more persuasive argument than the other. The only components of these proofs that we are allowed to question are the premises on which they are based. And the only difference in the premises is in their respective treatments of section 1.1. Given the task of establishing Peter's citizenship in court, all we have to do now is persuade the court that BNA1's interpretation of section 1.1 is the right one to take, in particular that it is better than BNA2's version.

We have considered here an unrealistically simple example. In practice, we can expect many conflicting formalizations instead of just two, and the implementational problems that arise are more severe. We could consider the separate formalizations which are distinct but which nevertheless contain a vast amount of duplication. Such issues are important and interesting in their own right, but they are mostly irrelevant to the main theme of this chapter.

The example we have just considered is also atypical. We shall usually be interested in producing arguments, not from conflicting formalizations of some fragment of written legislation (which is normally a relatively precise definition of some legal concept), but from conflicting rules for a vague concept left undefined by the legislators and for which case law has subsequently been established. Nevertheless, the processes which are involved are essentially the same, whatever the source of the rules.

THE ADVERSARIAL NATURE OF LEGAL ARGUMENT

We have suggested that vague concepts can be identified with situations where we have reasons for believing that both the assertion and the denial of the concept are true, and that these reasons can be captured by a system of conflicting rules. The key requirement is that these rules should be capable of producing arguments, both for and against the required conclusion. The approach may be illustrated by considering an example which is often used for explaining the essence of vagueness. We begin by proposing the following rules, all of which seem to express something true about the concept of "tall":

i) a person is tall if he is over 6' (183 cm);
ii) a person is not tall if he is under 5' (152 cm);
iii) if A is taller than B and A and B differ in height by only a small amount, then B is tall if A is tall, and A is not tall if B is not tall.

Let us now see how we might use these rules. Suppose we had a platoon of soldiers standing in a line, arranged by height standing. If the tallest in the rightmost and the shortest was the leftmost. Moreover, suppose that the rightmost person was over 6' (183 cm) and the leftmost was under 5' (152 cm), and that all the soldiers in the line differed in height from their neighbours only by a small amount. Now to decide whether a person in the middle, who is between 5' (152 cm) and 6' (183 cm) in height, is tall or not tall we could use rule (iii). If we start at the right of the line we can show he is tall. If we start at the left of the line we can show he is not tall. What we have are two arguments: one to the effect that he is tall; and one to the effect that he is not tall. It is now up to us to choose which of these arguments we want to accept. Of course, in this example, we have an obvious way of evaluating the argument, since each step forms a link in a chain that grows progressively weaker as new links are added. But we should also be wary of evaluating this argument in the absence of knowing why we need to know whether the person is tall or not tall. For it is characteristic of arguments that we may choose to reject a strong argument, and accept a weak one instead, providing we are doing so.

The argument in the example above has a form that resembles a proof. Although it is not usual to apply adjectives like "persuasive" and "weak" to a proof, we now suggest that this resemblance is more than superficial. When a system of conflicting rules is used to generate contradictory conclusions, then the proofs which are constructed in the process do take on the nature of arguments. An argument, like a proof, starts from some assumptions or premises and moves by rules of inference to a conclusion. In the case of a proof we know that the rules of inference are truth-preserving: if we accept the premises it is not open to us to deny the conclusion. But we can properly refuse to accept a proof, by denying the premises on which it is based. It is for this reason that arguments can be identified with proofs: that arguments are persuasive rather than sound arguments that they are sound arguments of valid rules of inference to the premises they are given) but weak (in that the premises may be questionable); and that two equally sound arguments may give contradictory conclusions. It is for this reason that we can accept the conclusion of an argument.

If we now apply this line of thought to the open texture of law we will begin to see very close correspondences. For it is typically the case that the open texture of law is resolved by argument. The legal decision-making process is essentially adversarial in which arguments are presented for both sides of the question. The judge must be the judge to arbitrate the two sets of arguments. The judgement is an evaluation of the arguments with which the judge has been presented; he will not necessarily dismiss one set of arguments as invalid, but he will say why. Sometimes more persuasive than the other. Moreover arguments are usually reasoned. Which is to say that the judge will not simply pronounce his conclusion, but will say why he concluded as he did, citing the arguments for his conclusion and saying why he found them persuasive or found the contrary arguments less persuasive. It is these reasons, not simply the fact that he decided as he did, that will form the basis of any precedent that might be set by the judgement.

And the main constraint is that the use of open-textured concepts are not such as would determine whether or not an object fell under the concept, but rather such as would enable us to argue that it did and that it did not. The main constraint is that the open nature of the concept in a matter of choice, although in the law, a machinery for arbitration may be established. To represent such concepts one needs to represent these rules, accepting that they will
produce arguments for contradictory conclusions. The role of the system will be to produce the arguments, and the user of the system will then choose which arguments he finds the more persuasive.

We suggest therefore that a computer system that is designed to give advice on matters of law and open texture should be concerned not with the production of a conclusion but rather with presenting the arguments on which the user may base his conclusion. And it should not be forgotten that in many applications of legal systems it will be the arguments and not the conclusion in which the user is interested. An individual wishing to support his claim for an additional benefit will want to know what reasons he can adduce to show the proposition is entitled; and insurance officer writing a submission to an appeal tribunal will wish to know what arguments he can put forward to support the decision that he made; and an advocate who must present his client's case in court will want to anticipate the opposing arguments he is likely to meet. If we want to decide support, then arguments putting both sides of the question, showing what sort of considerations should be taken into account and what sort of arguments were found persuasive in the past, should be what we provide. If we follow Hart in particular, there really is no matter of the fact, and it is unreasonable to expect the system to produce a conclusion. Resolution of open texture lies at bottom a matter of choice, and it is the duty of the user to make that choice; the duty of the system is to ensure that the choice is as informed as possible.

Of course it now becomes critical to see what assistance could be given in the evaluation of arguments. It is always possible to give some help with the assessment of a generated argument the user may accept or reject. For example, we might output either an explicit or an implicit rule, and every premise can be justified to some extent from supplementary documentation included in the system. In the case of written law, this documentation would normally compare the chosen formalization with the original text, and would refer any other sections of legislation which had been considered relevant. It should also point out, as in the case of rules for section 11 of the British Nationality Act, any assumptions which had been made in its construction. For rules generated from case law, the justifying documentation would include at the very least a reference to the case from which it is derived, for some indication of which particular features of this case had been adjudged to be most relevant. This kind of help with the assessment of arguments may well be adequate for many applications in simple domains, and we shall be proceeding on that basis, for the moment. Nevertheless, many important issues remain to be investigated because arguments which have been found persuasive and unpersuasive in the past would have to be shown as such. Not only is there an intriguing possibility that the very notion of what it means for an argument to be persuasive is itself vague. Not only is it vague, it is presumably open-textured. For example, sometimes the precedent-setting decision of a particular case is not regarded as relevant to the facts of the case itself, but also with establishing constraints on what will be regarded as a persuasive argument in the future. If our proposal is coherent it should be possible to apply the treatment to representing the "persuasiveness" of arguments. And since this itself would involve generating conflicting arguments, which could be more or less persuasive, there is a danger that the system will eventually collapse into a kind of infinite regress. We can be less pessimistic, however. We have in this chapter introduced a number of alternative treatments of open-textured legal concepts, including what we call approximation. We have described how it might be possible to construct a sophisticated approximation, of almost unlimited accuracy, by choosing the methodology of the "classical" expert systems. One obvious way of proceeding, therefore, is to construct an expert system that attempts to describe the user's intention. Moreover, even apparently sharp concepts may, because of the essentially open-textured nature of law, require similar treatment.

For applications where we require legal decision support we have proposed a system of conflicting rules. These rules are designed to present the relevant arguments for and against a case. The conclusion as a basis on which the user can make his own decision. In the law, questions of open texture are resolved by the judge and the user's decision on the basis of what is called a reasoned decision to accept an argument. We have further suggested that generating arguments is a process which is essentially identical to generating proofs so that the same techniques could be employed in both cases. This makes it possible to define vague concepts using rules, providing that we account for the technical difficulties attendant on maintaining a database capable of generating contradictory conclusions.

Clearly what has been presented here is only a strategy, but one which seems to have promise, particularly for legal applications where it corresponds closely to the actual process of reasoning with premises of law. It is the purpose of this paper to try these ideas on a real application. We plan to separate our investigation into three phases. First, we need to identify suitable areas of law. Ideally, we would choose two
complementary and contrasting domains: one, dealing with the resolution of such low-level concepts as "hard to heat," would be intended to help official arrive at more informed and considered decisions by simulating for them the "due process of law"; the second, treating a "real" legal concept, would be intended to help lawyers in their search for relevant case law and precedent. In the second phase, we need to investigate whether the generation of conflicting arguments is of real practical help, and for which of our intended users. Third, we need to consider how our system could be improved to aid in the evaluation of arguments. In the longer term, we hope to pursue what we have identified as a critical requirement: a representation in computer-intelligible terms of what it is that makes a legal argument persuasive.

NOTES


