Generating Exception Structures for Legal Information Serving

R. Winkels, D.J.B. Bosscher, A.W.F. Boer, J.A. Breuker
Dept. of Computer Science & Law, University of Amsterdam
PO Box 1030, 1000 BA Amsterdam
Netherlands
Tel: +31 20 525.3485
{winkels,bosscher,aboer,breuker}@lri.jur.uva.nl

ABSTRACT

More and more legal information is available in electronic form, but traditional retrieval mechanisms are insufficient to answer questions and legal problems of most users. In the ESPRIT project CLIME we are building a "Legal Information Server" (LIS), that not only retrieves all relevant norms for a user's query, but also *applies* them, giving the normative consequences of the 'situation' presented in the query. Typically, these queries represent very general and underspecified cases. Underspecification may lead to 'overlooking' of relevant norms, in particular those norms that directly change the legal status of a case: *exceptions*. Most exceptions in legislation however, are implicit, i.e. will only be detected *after* trying all norms for a particular case and resolving conflicts between applicable norms. For LISs we suggest to make the exception relations between norms explicit in off-line mode, so that we can use these *exception structures* to warn users about potential exceptions to their queries.

Keywords

Legal Information Serving, legal database, exceptions.

1. INTRODUCTION

Regulations, laws and precedent cases grow in number and complexity, and become more and more available in electronic form. Access to electronic legal sources has thus far been handled in the same way as - or rather following - information retrieval in general by using databases or (structured) text bases (see *e.g.* [14] for an overview). The search engines at the WWW are a good example of the state of the art of this kind of information retrieval. For answering legal questions or solving legal problems, key word matching has serious limitations, even if supported by conceptual retrieval techniques. Typically, the input query combines key words through Boolean and proximity operators, and the output is a list of (ranked) (parts of) documents. The user will have to read and interpret the output himself to answer the question. Moreover, the quantity and quality of the search result leaves much to be desired. One may find a lot of irrelevant documents (low precision) and probably not all relevant ones (low recall).[1] These techniques are directly borrowed from information retrieval, but miss out what is the crucial issue in legal information serving: reasoning about the legal consequences of the query [2]. In legal information serving the user is interested in the question whether some situation is allowed or required.

ICAIL-99 Oslo, Norway Copyright (c) 1999 ACM 1-58113-165-8/99/6 $5.00

The LIS presented in this paper is under construction as part of the ESPRIT project CLIME ("Computerized Legal Information Management and Explanation", Esprit P25.414; for more information on the project see e.g. [17][2] and [18]). Besides delivering a generic architecture for LIS, a demonstrator will be delivered for the domain of ship classification, a requirement for access to ports, insurance etc. Every ship classification society maintains a set of rules for assessing ships. These regulations cover a few thousand of pages of text and are available in electronic form to the society's employees, who are spread world wide, and their clients (ship owners).[3] Besides these regulations, international treaties, *e.g.* on safety (SOLAS) and preventing maritime pollution (MARPOL) are applicable. Typical LIS requests are "What is the minimal number of bilge pumps that is required on a cargo-ship", "Are passengers allowed on a bulk-carrier?" Answering these questions involves *assessing* the normative status of the request, i.e. matching norms to a situation description, in the same way as in the assessing the legal consequences in a legal case. However, legal information serving generally differs from the legal assessment of cases because a LIS query may be incomplete and underspecified. Exactly what we mean with incompleteness and underspecification will be explained later. It should be noted that this incomplete and non-specific nature of LIS queries is often explicitly intended. If one asks about passengers, one is not interested in bilge pumps. Moreover, if bilge pumps happen to be related to passengers in (one of) the norms, the LIS should make this explicit to the user.

---

[1] Blair & Maron [1] found that 'full-text' databases provided only 15% of all relevant documents and 30% of critically relevant documents, while at the same time the users thought they had found 80% or more of all relevant documents.

[2] This paper is also available on-line at: http://www.lri.jur.uva.nl/~winkels/papers/AIL98_Clime_Paper.html

[3] The classification regulations in CLIME are those of Bureau Veritas, one of the oldest classification societies. The regulations are electronically accessible on CD-ROM or via internet, coded in SGML.

The normative nature of these requests transpires in the appropriate answer, e.g. for the first question the LIS may explain that a bulk carrier is a cargo ship, and that cargo ships may carry a limited number of passengers according to rule so-and-so.

In order for the LIS to retrieve relevant norms for a user's query and apply them – i.e. assess the normative status of the situation put forward in the query – the content of the legislation has to be represented in an appropriate way. Not only the concepts that are used (as in conceptual retrieval), but also the normative content. We will describe our way of modeling the legal knowledge in legislation in the next section.

2. MODELLING LEGISLATION

Any regulation refers to objects in the real world. Hence there is a need to represent these objects and the way in which they are related. This conceptualization of what we call 'world knowledge' is the starting point of legal modeling. An example:

Art 8:1 - the first article of book eight of the Dutch civil code - starts with the definition of a ship: "In this book ships are all goods, which are not aircrafts, that are constructed to float, float, or have floated."

This gives rise to a number of concepts and relations which have to be taken into account. E.g. a ship is a 'good', it is not an 'aircraft', it is 'constructed to float', etc. In order to model articles in the civil code, which use the word 'ship', the knowledge base of the LIS must contain these notions. In a class hierarchy the ship is subclass of good, and so is aircraft, and they are mutually exclusive in the context of the civil code. Once we have modelled the concept ship, we can say that an object is a ship, a good or an aircraft.

Of course, 'world knowledge' does not (cannot) represent the world in all its aspects; it is a (legal) abstraction of the world. Moreover, it is incomplete and not self-contained, but refers to and depends on a large amount of common-sense knowledge.[4] Legal sources may define numerous concepts and their relations, but in the end these will be defined in terms of primitives that have to be interpreted by humans (the typical 'end-users' of legislation). In the ship classification domain for instance, many types of ships are defined, based on their constituent parts, cargo, etc., but the term 'ship' itself is not, neither the terms 'cargo' or 'repair'. These are left to common-sense interpretation by humans. In this sense, the world knowledge, and the models that it constitutes, are an interface between the 'real world' and the 'legal world'.

The LIS reasons about queries. Queries are descriptions of situations phrased in terms of the world knowledge. E.g. once the notions relating to a ship are modelled, one can present a query to the LIS of a ship which is in a harbor (maybe to check port regulations). Of course the LIS operates on a formalized representation and not on natural language; an input query could be something like:[5]

Query1: {ship ("HMS Amsterdam") $\wedge$ in-location("HMS Amsterdam", harbor)}

World knowledge is prerequisite for modeling *norms*. Norms are the smallest entities in legal information serving which give a normative status to a situation, i.e. state whether a situation is allowed or disallowed. We discern two main aspects of a norm. First, a norm obliges, forbids or permits situations. We call this the normative function of a norm. Second, a norm refers to one or more situations. Here lies the connection to world knowledge. The collection of situations to which the norm refers is phrased in terms of the world knowledge. An example: Suppose that there is a (mock) regulation: "It is permitted for a ship to be in a harbor". The norm is a permission and refers to any situation with a ship in the harbor. It presupposes a representation of ship, locations and harbor in world knowledge. A situation with a ship in the harbor, is thus explicitly permitted by this norm. For the purpose of this paper, we will represent this as:

Norm1: P{ship(X) $\wedge$ in-location(X, harbor)}

where 'P' stands for 'Permission'. We call the general situation to which the norm applies the 'generic case' of the norm. When a user's query matches the generic case of a norm, or to put it differently, when the truth of the generic case can be derived from the situation presented in the query, given the world knowledge, the norm is applicable. It can easily be seen that 'Query1' above matches 'Norm1' above and the resulting normative qualification will be 'allowed'. Most of the time however, the LIS will have to do a lot of reasoning with the world knowledge to be able to decide whether a case matches a norm.[6] This is because the terms used in queries (or legal cases in a straightforward legal assessment task) are seldom at the same level of abstraction as the terms used in the norms. In the example query about whether passengers are allowed on a bulk carrier (see Introduction), the relevant norm refers to cargo ships, not to bulk carriers. Norms are formulated abstractly to provide a large coverage of situations. It is i.a. for that reason that the use of database technology or text-retrieval methods leads to low precision and low recall scores. Conceptual front-ends[7] may improve the matching with the abstractions in legal sources, but they do not cover in a principled way how implied knowledge is handled.

When more than one norm is applicable and their resulting qualifications conflict (i.e. allowed versus disallowed), meta-norms are tried to resolve the conflict (see below).

In the case of Norm1 above there can be a direct correspondence between the regulation ("It is permitted ...") and the norm in the

[4] Part of this 'incompleteness' of world knowledge is due to what is known in legal theory as the 'open texture' of legal concepts (cf. [4]). We do not offer any (new) solutions to that problem. We model what is available in the legal sources and leave the rest to the users of the LIS – which in our case are no novices in the domain. The quality of the things we *do* model obviously has implications for the quality of the reasoning of the LIS; see the Discussion.

[5] For the purpose of readability, we keep all representations in this paper semi-formal.

[6] We use a description classifier to see whether a query matches the generic case of a norm. Currently we use LOOM (see: http://www.isi.edu/isd/LOOM/LOOM-HOME.html), but we are also investigating other systems.

[7] As *e.g.* in the FLEXLAW system [13].

legal information server. In legal text, regulations usually give rise to several norms. The connection between the norms and the regulations they are modelled after, is kept in referential links.[8]

Note that we label the *entire* generic case as being permitted, contrary to most approaches that would represent the regulation as 'concluding' that some aspect of the situation is permitted given some condition(s), e.g.:

Norm1b: Ship(x) $\rightarrow$ P{in-location(X, harbor)}[9]

As we will see later, this difference has repercussions when distinguishing and representing exceptions.

3. ASSESSING SITUATIONS IN QUERIES

In assessing a legal case, the assumption is that the case description is fully specified and complete. With complete we mean that all legally relevant facts have been described. However, in a typical query to a LIS the user may not be interested in a full description of the case, but only in some aspect, e.g. how many bilge pumps are needed. Therefore, queries are almost by definition incomplete cases. Besides, queries may lack detail. Because in a case things have (hypothetically) happened, i.e. the facts of a case are instantiated facts, the case can be described at the lowest level of specificity. Describing a case in too general or incomplete terms may easily lead to a different outcome. For instance, when we state in the query that it concerns 'a ship', instead of the 'HMS Amsterdam' which is a liquid-gas carrier of a gross tonnage of 10,000 etc., we will miss out all norms which are applicable to liquid-gas carriers, tankers, and cargo ships, and only norms about ships in general will be applied. Of course, we could have the LIS infer all possible specifications of the abstract case description, but this would be combinatorial and it would also be infeasible for a user to handle all these largely non-relevant extensions of the case presented.

As we stated in the Introduction, in legal information serving the specification of complete cases is hardly ever relevant. Therefore almost by definition the situations in a LIS query are incomplete, or rather: focussed on only one or a few topics. Moreover, many typical LIS questions may not be specific at all. The user, who asks whether it is allowed to have passengers on board a bulk carrier, may not have a specific bulk carrier in mind, but as owner of a fleet of bulk carriers, he may consider additional exploitation of this fleet. Therefore, most LIS queries refer to general situations rather than to fully instantiated cases. As we have seen above, to assess the normative consequences of a legal case, the terms in the case description normally have to be *abstracted* to the level of the terms used in norms. Now in assessing the normative consequences of situations described in LIS queries, we see that the user may use too general terms in his query, which suggests we have to *specialize* the terms used in the query in order for the right norms to match. Of course, this is much more of a problem, since abduction does not guarantee correctness.

[8] These links are not only handy for maintaining the LIS, but are also used for explaining the answers to queries by referring to, or stating the original sources.

[9] We could also write our representation as: {ship(X) $\wedge$ in-location(X, harbor)} -> P to show that we only 'conclude' that the situation is permitted, not which aspect(s) of it.

To indicate this crucial difference between legal case assessment and legal information serving, we will use the term *'query'* rather than 'case' for the incomplete and underspecified cases presented to LISs.

That LIS queries are limited to only a few topics is an advantage rather than a problem for the assessment algorithm: in general, the time spent on abstraction and matching of a case is exponential with respect to the size of a case description. Underspecification is not a problem either: it means less steps in the algorithm to find matching norms. However, underspecification may give rise to another kind of problem. The user may not understand: (1) what the required level of specificity is given his intentions and the general problem situation at hand, and (2) that the outcome should be interpreted with the caution that it is only correct with respect to his specified request.

To prevent the first problem, the CLIME user interface is constructed in such a way that the user, who inputs his request in a structured natural language format, is shown more specific options for the terms he uses.[10] Moreover, the user may start a follow up dialogue, when the answer to his request is not what he thinks he requested or needs. However, the user may be too easily satisfied with an answer, and in particular he may not be aware that further specification may trigger exceptions. Therefore, in co-operative legal information serving, the user should be warned about potential exceptions.

A simple example may illustrate what is meant. Assume we have the following two norms: Cargo ships are not allowed to have more than 13 passengers aboard; A cargo ship in harbor may have more than 13 passengers aboard. In our more formal notation:

Norm2: F{ cargo-ship(X) $\wedge$ nr-passengers(X,Y) $\wedge$ Y>13 }
Norm3: P{ cargo-ship(X) $\wedge$ nr-passengers(X,Y) $\wedge$ Y>13 $\wedge$ location(X, harbor) }

Obviously, the second norm is an exception to the first – though this is not stated in the norms. Let us further assume we have an explicit meta-norm that states that Norm 3 is stronger than Norm 2. This meta-norm is an expression of the general '*lex specialis derogat legi generali*' principle.

Now a user enters the query:

Query2: { cargo-ship("HMS Amsterdam") $\wedge$ nr-passengers("HMS Amsterdam", 15) }

and wants to know whether this situation is allowed. The query matches the generic case of the first norm and will be assigned the qualification 'disallowed'. In this case the LIS may correctly inform the user that the situation is disallowed because of Norm2, and refer to the original text. This may however, not be the most useful answer. He or she may be helped more with an answer like: "That is not allowed unless your cargo ship is in a harbor". In order to provide the second answer, the LIS has to: (a) recognize that Norm3 is an exception to Norm2, and (b) compute the

[10] This is achieved by the "What You See Is What You Meant" technology, developed by the University of Brighton [9]). Basically, users are directly editing a 'knowledge base', but the results are constantly shown in structured natural language.

difference between the generic cases of the two norms. In the example these two steps are rather straightforward: The meta-norm indicates that Norm3 is an exception to Norm2 and the generic cases are identical except for the 'location' predicate. In most cases, however, it is not as simple. Exceptions come in many forms, and generic cases may be very different.

4. IDENTIFYING EXCEPTIONS BETWEEN NORMS

An exception is an instance that does not conform to a rule or definition that apparently applies to it. In a legal context, we assume that (most) exceptions to rules result from deliberate acts of exclusion on the part of the legislator. Using exceptions has a pragmatic, not a principled reason. It is possible to re-phrase any otherwise consistent piece of legislation containing exceptions in one without exceptions. However, the result is a far less abstract and almost unreadable version of the regulation: the 'qualification model' [3]. A legal case is an exception to a norm if the norm apparently applies to it, but the case does not conform to the norm. The idea is that there is an exception rule that applies to the case in a more specific way and defeats the more general norm. Thus, the more general norm and its exception are in conflict with respect to the case and the LIS has to choose.

Following Pollock [10], many researchers distinguish 'rebutting defeaters' from 'undercutting defeaters'. The first type is an exception that explicitly negates the 'conclusion' of the 'rule' it is an exception to. The second type just states that some other 'rule(s)' is/are not applicable, without itself giving a different 'conclusion'. An example of a rebutting defeater in our domain of ship classification is Norm3 given in the previous section. An example of an undercutting defeater might be a statement like "The provisions in this section do not apply to passenger ships". One way to see these undercutting defeaters is that they 'defeat' an assumption or implicit 'condition' in the rule(s) they 'undercut' (cf. [12]), e.g. the assumption that "the provisions in this section apply to all ships". This could be represented by making the 'corrected' assumption explicit. In the example given, all rules in the section have an additional condition that the ship at hand is not a passenger ship. Most researchers agree that this is not an elegant solution (cf. [11, 16]). Another way to handle them, is to represent this kind of scope restrictions for the applicability of norms as meta-norms. We do not consider these type of 'undercutter' defeaters to be exceptions, since there is no conflict, i.e. opposing normative qualification of a case (see next section for a formal definition of what we consider to be an exception). What we are looking for are exceptions of the 'rebuttal' type, and only between norms (e.g. not between defintions in the world knowledge). How do we detect these in legislation?

Sometimes, regulations explicitly state that a particular norm is an exception to another norm with phrases like "in contrast to article..." or "...unless article X is applicable". Most exceptions however, are left implicit, i.e. will only be detected *after* trying all norms for a particular case and resolving conflicts between applicable norms. Take the example from the previous section again. The fact that Norm3 is an exception to Norm2 will be detected after a case with a cargo ship in a harbor with more than 13 passengers has been judged. In that case, both norms are applicable, but their normative qualifications ('conclusions') conflict. The meta-norm has to be used to decide that Norm3 is stronger and the final qualification is 'allowed'.

A slightly more complicated example will show that detecting exceptions may also need additional, non-normative, i.e. 'world knowledge': Suppose we have a fourth norm that states that "Bulk carriers are allowed to have more than 13 passengers". This norm is an exception to Norm2, if we know (from the world knowledge) that a bulk carrier is a specific type of cargo ship. In our representation:

Norm4: P{bulk-carrier(X) $\wedge$ nr-passengers(X,Y) $\wedge$ Y>13}
World Knowledge: bulk-carrier(X) $\rightarrow$ cargo-ship(X)

That Norm4 is an exception to Norm2, can again be seen when we assess a case of a particular bulk-carrier that has 15 passengers. Given that a bulk-carrier is also a cargo-ship, Norm2 will qualify this case as disallowed, while Norm 4 will qualify it as allowed. Conflict resolution will have to be applied. In this case there is no explicit meta-norm, so we have to resort to general principles. Since there is no information that one norm is from a 'superior' law than the other, or from a later date than the other, the principle of 'lex specialis derogat legi generali' will be used. This leads to the conclusion that Norm4 is more specific than Norm2 and the final qualification will be 'allowed'. Norm4 *is* more specific than Norm2 because all situations that are covered by it are also covered by Norm2, but not the other way around. In other words, the generic case of Norm4 'implies' the generic case of Norm2.[11]

From these examples it can already be seen that an important heuristic or clue to trace exceptions is the use of P(ermissions) as normative qualification: Most P-norms are exceptions to some O- or F-norm.[12] However, not all exceptions are P-norms. For instance:

Norm5: F{liquid-gas-carrier(X) $\wedge$ nr-passengers(X,Y) $\wedge$ Y>13}

is an exception to Norm4 if we know that a 'liquid gas carrier' is a kind of 'bulk carrier'.

5. A METHOD FOR MAKING EXCEPTIONS EXPLICIT

The typical way to detect exceptions is by finding conflicts between two norms that apply to a case. When this conflict can be resolved by the specificity principle we say that one norm - the prevailing one - is the exception to the other one. Exceptions are not specific to the legal domain: the non-flying penguin is an

[11] This resembles the definitions of specificity of Poole [8] and its adaptation by Prakken [11], but the representations on which they work are very different. They do not distinguish 'world' from 'normative' knowledge as we do, and therefore their conflicts can be about either type, while we are only interested in normative conflicts (cf. the difference in representation discussed in Section 2).

[12] There are also '*inactive permissions*', not linked to any existing obligations or prohibitions, that are part of so called '*liberty-rights*' (Bentham as stated by Hart [5]).

exception to birds that are supposed to fly, and therefore, sets of norms are as 'non-monotonic' as any other kind of knowledge.

Note that we define exceptions as a specific type of conflict, i.e. that type of conflict where one norm is more specific than another and the norms qualify the 'case' in opposition (disallowed versus allowed). Other types of normative conflicts that can only be resolved by the 'lex superior' or the 'lex posterior' principle, are not considered as exceptions. These principles deal with the validity of a regulation, respectively the limitation of the scope of lex specialis notions. The distinction should be evident from the nature of these conflicts. We do not think that a more recent norm is an exception to an older norm: the older one is simply outdated. Similarly, a norm that dominates another norm is a more dominant one and not an exception. One may argue that lex superior has been 'invented' by the legal system to counteract the all pervasive predominance of the lex specialis principle. In this way, we can protect important norms or principles from being 'violated' by less important, but more specific ones and protect the integrity of the legal system.[13]

In fact, as humans draft the law, it can also contain conflicts that are not solvable by principles: 'genuine' logical conflicts (e.g. due to error).

Usually, exceptions are identified on line, i.e. a case is presented, conflicting norms match the case and the conflict is resolved by the specificity principle. However, this is not very cost effective. Whether two norms conflict is independent of a case:[14] cases help us to make us aware of the conflict. Therefore, we can try to infer all exceptions in a regulation off-line. This has important benefits. For us, the most important one is that it speeds up the on line assessment of the query, but an explicit exception structure may also be inspected to separate intended ones from not intended ones (cf. [3]).[15] Moreover, these explicit exceptions can be used to warn users about potential exceptions to their abstractly specified request in legal information serving as was stated earlier.

Above we have defined a norm as a normative function that maps the instances of some generic world description, i.e. the generic case, to a normative qualification. We can write a norm as a tuple (Φ,Γ), where $\Phi \in \{P,F\}$, and the generic case, Γ, is some first order formula. 'P' stands for 'Permission' and 'F' for 'Forbidden', see below for more details. To indicate norms we use the letters *N*, e.g. *N1, N2,...*

A norm is more specific than another norm - $N1=(\phi 1,\gamma 1)$ is more specific than $N2(\phi 2,\gamma 2)$ - if $\gamma 1 \Rightarrow_T \gamma 2$, where T stands for the shared world knowledge according to the model. Now *N1* is an ***exception*** to *N2* if the normative functions are different, but the (generic cases of the) norms imply one another:

$$exception(N1,N2) \equiv implies(\gamma 1,\gamma 2) \wedge \phi 1 \neq \phi 2$$

[13] This is why we often find civil rights in constitutions, the 'highest' legislation in a legal system, so that they cannot be overruled by lower more specific rules (perhaps at most restricted).

[14] Cf. Poole [8] and Prakken [11] who also handle 'specificity' of conflicting rules independent of case data.

[15] Not intended ones are in fact errors. They are not rare in law.

We assume that generic cases are cast in disjunctive normal form, which means that disjunctions in the description of a single generic case are not allowed. Indeed, if disjunctions are part of the apparent generic case of a norm, this norm can be rewritten as two or more norms containing no disjunction. For instance: "a bulk carrier or liquid gas carrier should have two additional fire pumps" is represented as two norms that F(orbid) that a bulk carrier, respectively a liquid gas carrier does not have two additional fire pumps.[16]

In the example we have translated the norm into a prohibition. In the model we use only two normative functions: P and F.[17] The function that represents obligations (O) can be rewritten as a combination containing a permission (P), a prohibition (F) and a negation. This means that there is only one type of exception in our model, i.e. between 'F' and 'P'.

The next question is, how do we get the implications. These have to be computed from the representations of the world knowledge, i.e. for each pair of norms we have to compute whether one generic case $\gamma 1$ is implied by another one $\gamma 2$. We use an active description classifier[18] to obtain these implications. We introduce Skolem constants for the existentially quantified variables in $\gamma 2$ in such a way that they satisfy the formula $\gamma 2$: disjunctive normal form makes this possible (see also the discussion in Section 7). Then we query the knowledge base to obtain a binding list for $\gamma 1$. The idea is to assert the facts of a generic case that make a norm apply, in order to find the norms *to which it is an exception*. If these assertions cause other norms to match as well,[19] then the first norm is an exception to these norms.

We demonstrate the idea using Norm2 and Norm4. We set out to prove that the second norm indeed specializes the first norm. We first assert that two objects x0 and y0 exist such that the properties bulk-carrier(x0), nr-passengers(x0,y0) and y0>13 hold. Notice that the second norm is now applicable. Next we query whether the generic case of the *first* norm holds. Since we can obtain bindings for all of the variables in the first generic case, the first generic case holds as well. Hence the second generic case implies the first modulo the world knowledge, since the world knowledge gives that 'cargo ships subsume bulk carriers'. We have *detected* that the second norm is an exception to the first norm.

When we do that for all pairs of norms, we gradually reveal the (hidden) exception structure between the norms in the legislation. The resulting structure will probably consist of sets of trees or graphs (DAGs) as depicted in Figure 1. From this example exception structure it can be seen that there may be several 'entries' in the structure that are no exceptions to any other norm

[16] Note that the usual formulation would be: "bulk carriers AND liquid gas tankers...", due to the ambiguity of the natural language term 'and' when mapped to a first order logic. The generic case cannot be based on the surface textual structure of the norm. The generic case is an intensional description of the cases the norm intends to allow or disallow.

[17] The P(ermission) is implied by the notion of obligation, where what is obliged is also permitted [15].

[18] E.g. LOOM.

[19] NB: Other norms that do not apply to exactly the same generic case.

in the legislation, but only to the normative default of the normative system (usually weakly allowed). In this case the norm that 'forbids' generic-case-1 (FGC1) and the norm that 'obliges' generic-case-8 (OGC8). FGC1 has two exceptions, OGC2 and PGC3, etc. For readability we have kept the 'O(bligations)' in tact, but as was discussed above, the exceptions actually concern the prohibition of the opposite generic case and the permission in which we transform an obligation.

Collecting the exceptions of a normative system is most probably a time consuming process. The result is an overview of the implications between the generic cases of norms that give rise to different normative qualifications. The implication relation or exception structure generated by the implication between generic cases can be used in a straightforward way to speed up testing. An algorithm to select the primary norms to be tried, starts with norms that are not exceptions, then proceeds with exceptions to these norms etc. Depending on the tree like structure of the exception structure this should reduce testing effort considerably.

are assigned to the variables of the generic case of the norm. Last, we make an inventory of which additional properties the objects have, e.g. belong to more specialized sub-concepts or sub-roles. Furthermore, we look at which new objects are demanded by the exception and how these new objects are related to the original objects.

For the example of Norm2 and Norm4, we first assert that there exist two objects x_0 and y_0. They have the properties bulk-carrier(x_0), nr-passengers(x_0,y_0) and y_0>13. The difference between the application of the Norm2 and its exception - Norm4 - is that x_0 moreover is a bulk carrier. Since there are no additional conditions, this gives that for the exception to be triggered, the cargo ship has to be a bulk carrier as well.

For a slightly more complicated example with three objects we use Norm2 and Norm3. Besides x_0, y_0 we would assert the existence of a third object z_0, location(x_0,z_0) and harbor(z_0). This is the minimal set of assertions to trigger the application of the exception - Norm3. The difference between Norm2 and its exception is now that the exception requires an additional object

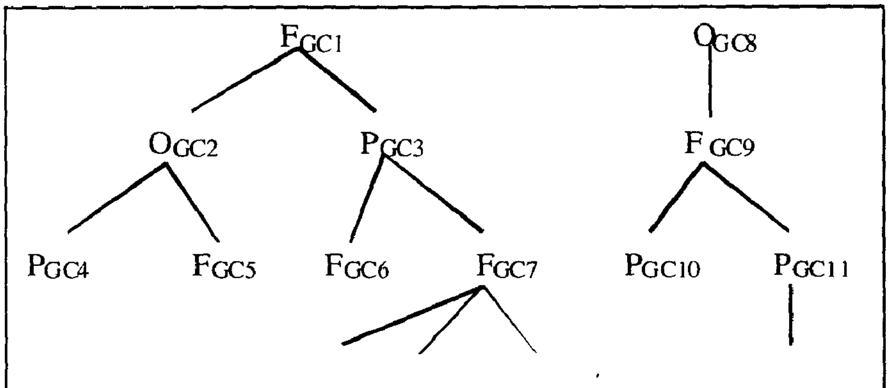

Figure 1: An example exception structure between norms

and a relation between an (already existing) object and this new object. So if Norm2 is triggered, then the conditions for the exception are met if the cargo ship is in the location of a harbor.

We see that the method for obtaining the differentiating information between a norm and its exception is similar to that of finding the exceptions to norms themselves, as described in the previous section. Computing this information can also be done beforehand. The relation between the objects mentioned in the two generic cases of the norm and the exception is static (i.e. independent of actual case or query data). If a norm is trigged, then asking for more specific data about existing objects and (relations with) new objects is a matter of substitution.

6. USING EXCEPTIONS TO OBTAIN MORE SPECIALISED QUERIES

We expect the legal information serving to be most practical if queries are 'small'. That way fewer primary norms, and consequently fewer meta-norms are triggered. The danger is, as was mentioned in the Introduction, that the query does not reflect the problem in reality good enough, i.e. does not trigger the 'right' norms that would have been triggered by a more elaborate description. How can we find the difference between an applicable norm and its exception to check whether or not in fact the situation of the exception is the case?

We use again the Example of Norm2 and Norm4 given above. If we have a query concerning a cargo ship with more than 13 passengers, i.e. a query that triggers an application of Norm2, then the missing fact to establish whether the exception can also apply is whether the ship is also a bulk carrier. We can use the exception structure described above to derive this information automatically. To obtain the discriminating facts between an exception and the norm 'to which it is an exception', the minimal number of facts is asserted that trigger the exception. Next, we establish how objects

7. CONCLUSIONS AND DISCUSSION

Legal information serving is different from information retrieval in general, because obtaining the right information about legal or normative issues invariably involves assessing the legal status of a situation description in a query. In many respects, this assessment procedure is identical to evaluating legal cases. As a consequence, the results and procedures in legal information serving differ dramatically from those obtained, respectively used in traditional information retrieval, in particular text retrieval. While in text retrieval there is strong negative relationship between the number of cues in a query and the number of matching documents or pieces of text, the opposite holds for legal information serving as discussed and explained here. In fact, we work from the assumption that the traditional metrics used in information retrieval – recall and precision – are both 100% in legal information serving.[20] The reasons are many, but the main ones are: (1) There is a full specification of terminology (ontology) required as a knowledge base, and this terminology is the (only) one that is to be employed for entering queries. This is similar to using conceptual retrieval front ends to text bases. (2) The assessment function in LIS is not concerned with matching terms, but with matching situation descriptions (cases) and solves the deontics involved in applying norms to cases. (3) Implied knowledge is taken into account in the assessment procedure (see e.g. [18] for more details).

On the other hand, we also pointed out that legal information serving differs from straightforward assessment of legal cases, because a query is in general focussed on a particular topic. The topic may be underspecified, so that special care has to be taken that the user will not misunderstand the scope of the results. As a consequence, a co-operative dialogue may not only prevent underspecification of queries, but is sensible to point to potential exceptions to the result presented. We discussed the nature of these exceptions and how an exception structure could be generated off-line.

This procedure for detecting exceptions between norms by using a description classifier like LOOM, does not ensure that we will find *all* exceptions in a regulation. First, there is the classical problem that a description classifier is not complete. Neither would a theorem prover be, given the minimal level of expressiveness that we need, but we have no precise information how the LOOM classifier compares with such a benchmark. Second, and of more practical importance is the modeling problem. Incomplete results may be rather due to inaccurate or erroneous modeling of the world knowledge than to the classical expressiveness versus completeness trade-off (cf. [6]). What will count and will be computed as implication is dependent on how terms are defined. For instance, in modeling a world in which (qualitative) spatial relations play a role, we may easily forget to include axioms like that right, middle and left are exclusive, so that one can infer that 'right' implies $\neg$('middle' $\wedge$ 'left'). However, within the scope of these classical limitations we claim that we will identify all exceptions of the specified type between norms.

In fact, we may identify too many exceptions. We have formalized the notion of 'specificity' as logical implication, but it is not obvious whether all implications reflect the notion of specificity. Specificity is probably one of the vaguest terms in common sense, but also in law. For instance, in many representations of knowledge, causality is viewed as a (material) implication, so that 'A causes B' is expressed as 'A implies B', but we probably do not want to derive that 'A is more specific than B'. In our approach, definitional knowledge in legal sources is modelled separately from causal knowledge, so we do not have this problem [15].

We have also sketched a method to obtain the more specializing or differentiating information between a norm and its exception. As one can see, this computation is similar to the detection of the exception to the norm itself. This shows the limit of its use in legal information serving: if the computation of exceptions is not guaranteed to be complete, then computation of more specializing information suffers the same fate.

A more complicated situation arises if a norm has more than one exception to it (and these cannot be ordered by implication of their generic cases). In this situation we have no clue for which exception information should be asked. Also the differentiating information of the separate exceptions cannot be merged, due to possible inconsistencies. It is even harder when more than one norm with an exception applies to the user's query. We have to develop heuristics to select the norm of which exceptions are tested and the order in which the exceptions are checked for dependent incompleteness. These problems are highly related to planning and managing dialogue and explanation between the LIS and its users, the responsibility of other partners in the CLIME project.

[20] Of course, like in traditional information retrieval, it is not completely clear what the standard is with which to compare 'retrieval' or LIS results. It is relative to the user's needs and typically operationalized by human experts in the field.

8. ACKNOWLEDGMENTS
CLIME is sponsored by the EC ESPRIT programme with project number P25.414. The CLIME partners are: British Maritime Technologies (UK), University of Brighton (UK); Bureau Veritas (France); TXT (Italy), and University of Amsterdam (Netherlands). The official CLIME WWW-site is at: http://www.bmtech.co.uk/clime/.

9. REFERENCES
[1] Blair, D.C. and M.E. Maron (1985). An evaluation of retrieval effectiveness for a full-text document retrieval system. *Communications of the ACM 28(3)*:289-299.

[2] Breuker, J.A. (1992). On Legal Information Serving. In *Proceedings of JURIX'92, Information Technology and Law.* Koninklijke Vermande, Lelystad, pages 93-102.

[3] Haan, N. den and Breuker, J. A. (1996). Constructing Normative Rules. *Proceedings of JURIX'96*, pages 135-147.

[4] Hart, H. (1961). *The Concept of Law.* Clarendon Press, Oxford.

[5] Hart, H. (1973). Bentham on Legal Rights. In A. Simpson (ed). *Oxford Essays in Jurisprudence.* Clarendon Press, Oxford, pages 171-201.

[6] Levesque, H.J. and Brachman, R.J. (1987). Expressiveness and tractability in knowledge representation and reasoning. *Computational intelligence, 3*:78-93, 1987.

[7] Lindahl, L. (1992). Conflicts in systems of legal norms: A logical point of view. In P. Brouwer, T. Hol, A. Soeteman, W. van der Velden and A. de Wild (eds.). *Coherence and Conflict in Law*, pp. 39-64. Kluwer, Deventer, 1992.

[8] Poole, D.L. (1985). On the comparison of theories: Preferring the most specific explanation. *Proceedings of IJCAI-85*, pp. 144-147.

[9] Power, R., D. Scott, and R. Evans (1997). *What You See Is What You Meant: direct knowledge editing with natural*

language feedback, Technical Report, ITRI-97-03, University of Brighton, 1997.

[10] Pollock, J.L. (1987). Defeasible Reasoning. *Cognitive Science 11*:481-518.

[11] Prakken, H. (1993). *Logical Tools for Modeling Legal Argument*. PhD Thesis, Free University of Amsterdam, 1993.

[12] Prakken, H. and Sartor, G. (1995). On the relation between legal language and legal argument: assumptions, applicability and dynamic priorities. In: *Proceedings of the fifth ICAIL*, ACM Press, Washington DC, pp. 1-9, 1995.

[13] Smith, J.C., D. Gelbart, K. MacCrimmon, B. Atherton, J. McClean, M. Shinehoft, and L. Quintana (1995). Artificial Intelligence and Legal Discourse: The Flexlaw Legal Text Management System. *Artificial Intelligence and Law 3(1-2)*: 55-95.

[14] Turtle, H. (1995). Text retrieval in the legal world. *Artificial Intelligence and Law*, 3(1-2):5-54, 1995.

[15] Valente, A. (1995). *Legal Knowledge Engineering. A Modeling Approach.* PhD Thesis, University of Amsterdam. IOS Press, Amsterdam, 1995.

[16] Verheij, B. (1996). *Rules, Reasons, Arguments. Formal studies of argumentation and defeat.* PhD thesis, University of Maastricht, 1996.

[17] Winkels, R.G.F. (1998). CLIME: Legal Information Serving Put to the Test. In: *Pre-proceedings of the Second French-American Conference on AI and Law*. Available on-line at: http://www.lri.jur.uva.nl/~winkels/papers/AIL98_CLIME_Paper.html

[18] Winkels, R.G.F., Boer, A., Breuker, J.A. and Bosscher, D. (1998). Assessment Based Legal Information Serving and Cooperative Dialogue in CLIME. *Proceedings of JURIX-98*, pp. 131-146. GNI, Nijmegen. Available in PDF format at: http://jurix.bsk.utwente.nl/papers_pdf/j98-pdf/j98-10.pdf