Approximating Knowledge in a Multi-Agent System

Miroslav Kubat* and Simon Parsons†‡

*Ludwig-Boltzmann Institute of Medical Informatics and Neuroinformatics,
Department of Medical Informatics,
Institute of Biomedical Engineering
Graz University of Technology
Brockmanngasse 41, A-8010 Graz, Austria
email mirek@dpmi.tu-graz.ac.at

†Advanced Computation Laboratory,
Imperial Cancer Research Fund,
P.O. Box 123, Lincoln's Inn Fields,
London WC2A 3PX,
United Kingdom.
email sp@acl.lif.icnet.uk

‡Department of Electronic Engineering,
Queen Mary and Westfield College,
Mile End Road, London E1 4NS,
United Kingdom.

Keywords: Artificial Intelligence, abstraction, granularity of knowledge, dl-cut, rough concepts

Edited by: Matjaž Gams
Received: ??????, 1993 **Revised:** ??????, 1994 **Accepted:** ??????, 1994

This paper is concerned with establishing a common language that can be used to communicate between the different members of a multi-agent system. We suggest that this may be done by successively approximating the concepts that each agent in the system deals with, and the paper gives algorithms which make this possible. Along the way we introduce the notion of a description language cut, or dl-cut, which is an abstraction to which a rich class of languages may be mapped. The idea of a dl-cut is then used to introduce rough concepts— rough descriptions of the concepts used by the agents. Finally we discuss the way in which rough concepts can be logically combined and used in deductive reasoning, also debating the scope of the validity of inferences using the concepts.

1 Introduction

Over the last twenty years, techniques from artifcial intelligence have been successfully applied to problems ranging from factory scheduling [23], to process control [15], and the diagnosis of faults in complex systems [12]. Expert systems have been developed which can replicate or exceed the accuracy of human experts [3], and which have bodies of knowledge that make them as knowledgeable as the

best informed human expert [16]. With the increasing power and sophistication of these systems have come a number of well-documented problems — in general intelligent systems do not scale up easily, they tend to be brittle so that their performance breaks down as they leave their domain of expertise rather than degrading slowly as that of a human expert would, and it is difficult to ensure that they are consistent.

A number of solutions have been suggested to remedy these ills, each stemming from a major research effort. One is to try to ensure consistency by constructing intelligent systems in a more rigorous way, structuring the knowledge that they contain and applying techniques from software engineering. This work is typified by the KADS project [22] and has led to interesting developments in areas such as the formal specification of knowledge-based systems [5].

A second approach is to reduce brittleness by building intelligent systems around a vast body of commonsense knowledge that approximates the kind of knowledge that people use in their interactions with everyday situations. This, in theory, will allow such intelligent systems to fall back on more general ideas when their specific expert knowledge fails. For instance, when a medical diagnosis system is queried about the ailments suffered by someone's car it should be able to transfer some of its basic knowledge about diagnosis and use this with commonsnese knowledge of how cars work to attempt an answer. The CYC project [14] which aims to do precisely this has recently released the first version of its knowledge base, and it will be interesting to observe whether the claims made for it are justified.

A third approach, and the one that we will consider in this paper, is to build communities of small, and therefore more manageable, systems. Because the individual systems contribute different skills and knowledge, together they are capable of handling problems that are beyond the scope of a single system. This is the approach of the ARCHON project [11] which has proved itself in the area of industrial process control in general, and in the construction of a co-operative system for electricity distribution management [4] in particular.

Now, one of the most interesting things about the ARCHON system is that it provides a framework for combining together existing systems. The motivation for this is the promotion of code and resource reuse, which is clearly a worthy aim, but in doing so raises a new and difficult problem. Different systems developed at different times may use different languages for knowledge representation. If this is the case, how should they be combined? Work on ARCHON understandably stopped short of providing an answer to this question, and it seems that little has yet been published on general solutions to the problem, though there has been some work on translating between different uncertainty handling formalisms in this context [19, 25].

However, some preliminary work has been published on related subjects. Huhns et al. [9] describe ways of integrating different information models of businesses, that is they discuss the problems of relating such models and resolving incompatibilities between them. To do this they make use of the CYC ontology, which is postulated to be of sufficient extent to encompass any notion in any business model, and integrate different models by integrating them into CYC. The result of this work is a system called Carnot, which provides an architecture and tools for integrating the information models of large businesses. Neches et al. [17] make a similar suggestion but from the more general perspective of integrating knowledge representation systems irrespective of domain or interpretation. To do this they suggest the idea of an "interlingua" which is a general language for knowledge interchange. At first blush, such a language certainly seems to be a good idea, but, as Ginsberg [6] argues, there are reasons why the definition of a standard interlingua seem premature.

This work on interlinguae assumes that there will be some underlying ontology, some

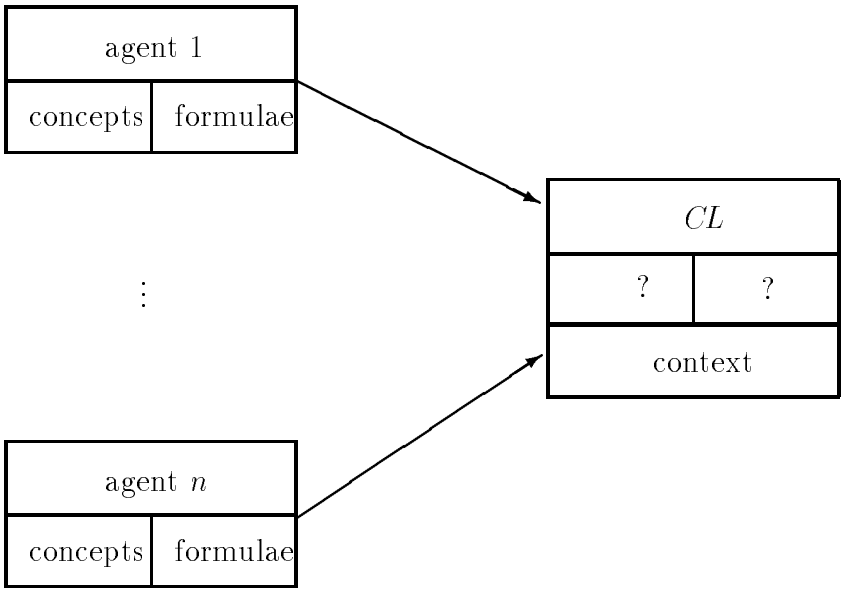


Figure 1: The system under consideration

basic set of concepts and their inter-relations, that is understood by all systems. Now, while such ontologies exist in some domains [17], they are far from universal, so there are domains in which the approaches discussed above will founder. In this paper we present some initial ideas about the way in which a model that is common to a number of intelligent systems that do not have a known common ontology might be constructed automatically from the models of individual agents within the group.

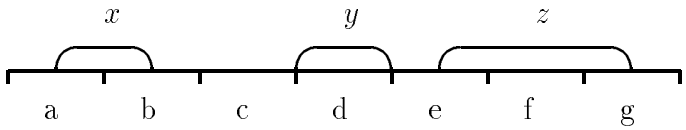
2 Basic concepts

Our inspiration to develop methods to obtain a common interpretation of knowledge from multiple sources stems from the domain of *multistrategy learning*, first proposed by Brazdil [1]. The ability of this principle to improve performance was demonstrated by Brazdil and Torgo [2], and by Torgo and Kubat [24]. In the particular case we will consider here, the problem involves several agents and a central system.

The agents possess knowledge which is expressed in a particular language and the task of the central system is to combine the knowledge into a general structure. The problem in doing this is that the language used by any agent may be different to the languages used by any other agent, and, if this is the case, the central system will need to translate the information obtained from the individual agents into some common basis which we will refer to as the central language *CL*.

Figure 1 illustrates the situtation we will consider. Each agent has a language which expresses a set of concepts and a set of logical formulae concerning those concepts. The central system contains the *context* of the overall system which plays an important role in any application of the knowledge of the multi-agent system since the kind of concepts with which the system will deal have connotations which vary widely according to the context. Thus, for instance, the concepts 'fertile land' and 'warm day', have widely different meanings in Central Africa and in Sweden.

Note also that in this case, unlike the agents, the central system has no direct access to the environment, however there is no theoretical reason why the central system should not have access, nor, for that matter why there should be a "central" system with a distilled common language. Instead the common language could be replicated in each agent, producing a group with no central focus, but whose mem-



$$x \subset \{a, b\}, y = \{d\}, \{f\} \subset z \subset \{e, f, g\}$$

Figure 2: Roughly described concepts x, y and z

bers could all understand each other.

Now, when the system of agents is initially set up, the central system has no understanding of the languages used by the various agents. However, it is possible that it can establish a common language by approximation. That is, it is possible for each agent to describe its knowledge to the central system in terms of its set of concepts. Depending upon the wealth of concepts available to the agent this may be a very precise or a very coarse description of its knowledge so that there is no guarantee as to the precision of the translation that is possible between the agent and the central system.

When all the agents do this the central system will end up with a language which can deal to some degree with all of the knowledge dealt with by all the agents, and so is broader than that of any single agent. In addition, due to the intersection between concepts, it may be more detailed than that of any agent.

What we propose in this paper are some thoughts as to how this might be achieved within the framework of rough concepts which we have developed [13] from ideas on rough sets [21].

A few informal definitions are needed to clarify some of the notions that we will operate with. A language, often called a *description language*, is understood as a set of well formed formulae (*wff*). We will only deal with languages with a finite number of *wffs*. Each *wff* represents a *concept* captured by the language. Each concept, in turn, is interpreted as a set of relevant objects assigned to it by its context. Thus, when speaking about students, we usually do not mean all students in the world. Rather, we implicitly constrain ourselves to the students of our university, students of Computer Engineering, students from the secondary school in the neighbourhood, and the like. The context represents additional information common to all concepts, and in this case we assume that the context is common to all agents.

Figure 2 presents a graphical representation of possible relationships between *CL* and concepts known by an agent. Each segment a through g stands for one *wff* of a simple *CL*, and consists of objects that cannot be further discerned in *CL*. Each *wff* is true for one or more objects. The lozenges x, y and z are concepts known by an agent. Note that, due to the different languages used by the agent and the central system, the boundaries of x do not coincide with those of the *wffs* of *CL* so that x is not described as precisely by *CL* as by the language of the agent. However, without any additional information, the classification of the objects from the segment b as positive or negative instances of x is completely unknown and cannot be quantified by a probability or a fuzzy degree of membership, so that this imprecise classification is still very useful. In addition, as concepts y and z illustrate, *CL* may be able to precisely distinguish the concepts of an agent, or even some subsets of some the concepts understood by an agent.

This issue is closely related to work on granularity such as that by Hobbs [8] who defined an *indistinguishability* relation for unary pred-

icates and Imielinski [10] who extended Hobbs work so that the idea can be applied to approximate reasoning. Our proposal also has notions in common with Carnot [9] which uses the idea of finding the best generalisation of a given concept, and with Ginsberg's [6] discussion of KIFs in which he proposes discarding details such as probabilities in order to facilitate interchange between agents that quantify their knowledge and those that do not.

For simplicity, we assume that the information possessed by the agents is noise-free and relevant. Readers interested in a method for pruning out noisy and irrelevant knowledge can find details in work by Brazdil and Torgo [2]. Thus the task that we will address is defined as follows:

Given:

A definition of the central language CL;

The general context expressed either as a set of constraints on the set of objects with which the multi-agent system will deal (such as the set of all types of car manufactured in Europe), or as a list of possible objects (such as Rover Metro, Nissan Micra, Ford Escort, ...);

For each agent, the descriptions of concepts and formulae in the agent's language which allow the agent to classify objects in terms of the concepts;

Find:

The description of all concepts in CL;

The scope of validity of the old as well as newly inferred propositions in CL.

The essence of this translation process is *abstraction*, a phenomenon that has been widely studied in artificial intelligence. A comprehensive analysis is provided by Giunchiglia and Walsh [7] where three types of abstraction are defined, depending on the ability of the source language L_1 and the object language L_2 to distinguish objects. Informally, an abstraction is *constant* if both languages discern the same objects; an abstraction is *decreasing* if L_1 is able to distinguish the same objects as L_2 and possibly some more; an abstraction is *increasing* if L_2 is able to distinguish the same objects as L_1 and possibly some more. The preceeding discussion makes it clear that, depending upon the exact concepts available, our method may give any of these types of abstraction, and indeed may give a mixture of different types for different agents in the same system.

3 Translating into CL

In this paper, no strong requirement is made on the syntax of the well-formed formulae— we use a logic-like notation to describe the attributes of the objects which exemplify the concepts the various agents deal with. However, this notation is used purely for convenience since it allows us to write down ideas such as "the shape of a certain class of object is either a cube or a pyramid" in a concise way as, for instance:

$$\Big(shape(x) = cube\Big) \vee \Big(shape(x) = pyramid\Big)$$

or:

$$shape(x, cube) \vee shape(x, pyramid)$$

and it should not be seen as a fundamental limitation on the approach— the results presented in the paper hold whatever form the *wffs* are written in.

To get an idea of the motivation for the dl-cut and rough concepts, consider a simple example.

Example 1.
Let a set of toy blocks be described by their shape: cube, pyramid, ball, and prism. The CL-language capable of describing the shape by means of these terms decomposes the set into four disjoint subsets, each of which is represented by at least one object. Suppose the concept to be translated into CL is 'stable in an earthquake.' Cubes and pyramids are stable, balls are not stable, and the stability of a prism depends on the ratio of its base area

to its height. Since no distinction is made between the different types of prism, *CL* cannot discern short, fat prisms (stable) from tall, thin prisms (unstable). If no additional information is available, the concept 'stable in an earthquake' can only be approximated by its lower bound (sufficient condition) and upper bound (necessary condition):

lower bound:

$\forall x$ *stable(x)* $\equiv$
shape(x, cube) $\vee$ *shape(x, pyramid)*

upper bound:

$\forall x$ *stable(x)* $\equiv$
shape(x, cube) $\vee$ *shape(x, pyramid)*
$\vee$ *shape(x, prism)*
□

The lower-bound description (*core*) is true for cubes and pyramids, whereas the upper-bound description (*envelope*) is true for cubes, pyramids, and prisms. Obviously, the 'distance' between the core and envelope depends on the language *CL*. Concepts expressed by the pair [core, envelope] are called *rough concepts*.

The notion of a *dl-cut*, defined below, will facilitate the formalization of the approach that we have just outlined. In the following definition, the universe U is the set of all objects seen by the central system.

Definition 3.1 (dl-cut) *Denote by Dl the set of all* wffs *of a language. A subset* $dl \subseteq Dl$ *is called a* dl-cut *iff it decomposes U into a system of pairwise disjoint sets such that each set is assigned precisely one wff* $f \in dl$ *that is true for all elements of this set.*

Thus in the simple system from Example 1, the universe of all blocks can be decomposed into four disjoint sets each of which is assigned one of the following predicates:

shape(x, cube), shape(x, pyramid),
shape(x, ball), shape(x, prism)

Each predicate is a *wff* and the set of four predicates is a dl-cut. In general there is not a unique dl-cut for a given universe. In this case, an alternative dl-cut is made up of the following three *wffs*:

shape(x, cube) $\vee$ *shape(x, prism),*
shape(x, pyramid), shape(x, ball)

The elements (*wffs*) of a dl-cut are *description items* or *generic concepts* which may be distinguished by the central system, and may be used by it to approximate the concepts handled by the various agents with which it communicates. Knowing that any disjunction of description items can be considered to be a concept, we can discern, by means of the dl-cut, 2^N different concepts, where N is the number of *wffs* in the dl-cut. The notion of a dl-cut facilitates a mapping of a rich class of languages onto easy-to-handle boolean expressions (for a deeper analysis, see [13]).

Basically, there are two possible approaches to the construction of a dl-cut from the underlying language— a naïve approach, and a concept-oriented approach. We only cover the naïve approach in detail, contenting ourselves with hinting at how the concept-oriented approach can be employed.

The naïve method uses a hill-climbing search technique. The initial state is the empty set of *wffs*, the final state is a set of *wffs* that form a dl-cut, and the search mechanism is to augment the current set of *wffs* with a *wff* that does not overlap with any previous *wff*. Obviously, an exhaustive search would reveal many different dl-cuts whose capacity to model concepts varies. Hence, the search must be made heuristic by adding some criterion for selecting which *wff* to add to the dl-cut. To be useful, this criterion should reflect the ability of the dl-cut that results from the addition the *wff* to model concepts. This approach leads us to propose Algorithm 1.

The prerequisite for this algorithm is the availability of a *generality criterion* which gives some idea of the quality of a dl-cut, and of a mechanism for a *subsumption test* to establish if one *wff* can replace another. The *generality criterion* in a system such as ours which is based upon the manipulation of at-

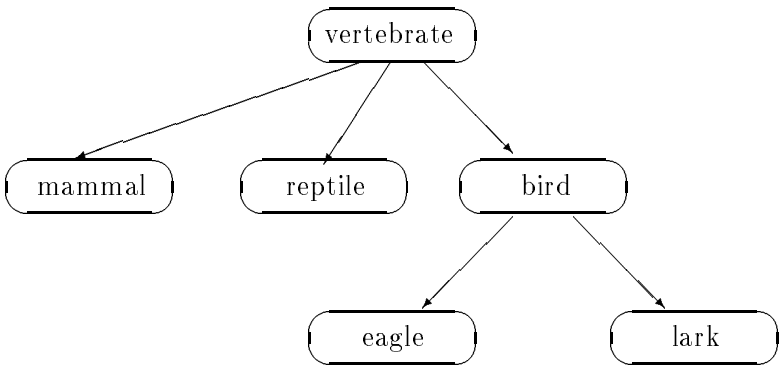


Figure 3: Example of a generalization tree

tributes naturally reflects the number of literals in an expression since each of these corresponds to an attribute of the domain. Thus A is more general than $A \wedge B$ and $A \vee B$ is more general than A. The *subsumption test* can be based on knowledge of the domain, so that 'bird' subsumes 'eagle' as in Figure 3 or on the explicit listing of the concepts represented by a *wff*. Thus if all objects representing concept A also represent concept B, then B subsumes A. By convention, we write $p \subseteq q$ if q subsumes p, and $p \supseteq q$ if p subsumes q.

A similar notion to subsumption is that of overlapping. Two *wffs* l_1 and l_2 are said to *overlap* if $l_1 = l_{11} \vee l_{12} \vee \ldots \vee l_{1n}$, $l_2 = l_{21} \vee l_{22} \vee \ldots \vee l_{2m}$, and $l_{1i} = l_{2j}$ for some i and j. Finally, the language CL is assumed to be sufficiently rich that at least some of its *wffs* l_i must be subsumed by some concepts s_j, that is $l_i \subseteq s_j$, and for each object $u_k \in U$, a *wff* l_p can be found such that l_p is true for u_k.

The input to the algorithm is the set S of all concepts, the context which defines the extent of the universe U, and CL, which when the first dl-cut is constructed will be empty, but for other dl-cuts will be a set of *wffs*. Further, let $L^{(CL)} = \{l_1, \ldots, l_n\}$ be the list of *wffs* from CL, in descending order according to some generality criterion so that for $l_i, l_j \in L^{(CL)}$, if $i < j$, l_j is not more general than l_i.

The algorithm terminates when S is empty or when the ability of $dl^{(CL)}$ to discern concepts cannot be increased without backtracking from the current state.

Algorithm 1

1. Let $dl^{(CL)} = \emptyset$;
2. Starting with l_1, search for the first $l_i \in L^{(CL)}$ such that an $s_j \in S$ can be found for which $l_j \subseteq s_j$. If no such l_i can be found, go to 5;
3. Let $dl^{(CL)} = dl^{(CL)} \cup \{l_i\}$. Discard all $l_j \in L^{(CL)}$ overlapping with l_i;
4. Delete from S all concepts s_i such that $s_i \subseteq d_j$ where d_j is any disjunction of *wffs* from $dl^{(CL)}$. If S is not empty, go to 2;
5. Find a *wff* that is true for the rest of the universe U, add it to $dl^{(CL)}$ and *stop*.

Of course, many different procedures using more powerful heuristics and search techniques can be proposed, and their detailed analysis is an open research topic. The algorithm we have presented can serve as a guideline.

Before we proceed to the illustration of the algorithm by a simple example, a few comments are necessary. Firstly, an additional requirement in step 2 can demand that the concept s_j subsuming l_i is not allowed to subsume any other concept s_k. This requirement makes sense if agents are able to order their concepts by subsumption.

Secondly, since the explicit storing of $L^{(CL)}$ would limit the utility of the procedure to very small languages, the list is intended to be implicit. Thus in the language based on conjunctions of unary predicates, the algorithm would begin with single predicates, then, when the ability of the predicates to describe concepts has been exhausted, proceed to conjunctions of pairs of predicates selected by a suitable heuristics.

It is also possible to derive an alternative algorithm driven by the concepts s_I instead of the *wffs*. This is the 'concept-oriented' approach hinted at above.

Finally, it should be noted that, in general, subsumption checking is NP-hard for first-order logic and must be assisted, in realistic applications, by background classification information based on notions of generality of concept, such as that depicted in Figure 3. A similar problem can arise with the discarding of overlapping *wffs* in step 3.

Example 2.

Consider once again the blocks world of Example 1 which is extended so that the blocks can be described in terms of the material from which they are made as well as their shape—all cubes, prisms, and pyramids are metallic while balls are wooden. The agents understand the concepts 'stable' and 'belongs to Tom', and are able to classify the objects in U as positive and negative examples of the concepts. In the first step, the central system picks the unary predicates one by one until one of them turns out to be subsumed by any of the two concepts.

Suppose that the concept 'stable' subsumes the predicates *shape(x, cube)* and *shape(x, pyramid)* while 'belongs to Tom' subsumes *shape(x, pyramid)*. The system selects *shape(x, pyramid)* as the first *wff* of $dl^{(CL)}$. From now on, all predicates overlapping with *shape(x, pyramid)* will be discarded so that only the predicates *shape(x, cube)*, *shape(x, prism)*, *shape(x, ball)*, *material(x, wood)* and their conjunctions are allowed to appear in any of the future *wffs*. Thus we might end up with $dl^{(CL)}$ being:

$$shape(x, pyramid), shape(x, cube),$$
$$shape(x, ball) \wedge material(x, wood),$$
$$shape(x, prism)$$

□

Now, we can define the important notion of a *rough concept*.

Definition 3.2 (rough concept) *Let* $x^R(dl) = [x^C(dl), x^E(dl)]$ *be a rough set. A* rough concept *is the pair* $[des(x^C), des(x^E)]$, *where* $des(x^C)$ *is the description of the core of* x *in* Dl_{dl} *and* $des(x^E)$ *is the description of the envelope of* x *in* Dl_{dl}.

Note that the core description does not apply to any negative instance of x, the envelope description applies to all positive instances of x, and the complement of the envelope applies only to negative instances of x. Beware, however, that any pair [core,envelope] pertains to a particular dl-cut. Different dl-cuts tend to imply different rough concepts since the core and envelope are *wffs* from $dl^{(CL)}$. In this respect, the idea of rough concepts departs from Pawlak's rough sets [21]. Even though a *wff* can be understood as a set of objects for which it is true, the symbolic interpretation of an approximation of a concept is dominant.

The next algorithm translates the concepts from the agent's language into CL. The input is formed by $dl^{(CL)}$ and by the concepts to be translated into CL. As output, the algorithm produces rough concepts in CL.

Algorithm 2

For each concept C of an agent, and for any $d_{c_i}, d_{e_j} \in dl^{(CL)}$:

1. If d_{c_i} is subsumed by C, then d_{c_i} belongs to the core;
2. If d_{e_j} overlaps with C, then d_{e_j} belongs to the envelope;

3. The core (respectively, the envelope) is the union of all items d_{c_i} (respectively, d_{e_j}), so that $C^C = \bigcup_i d_{c_i}$, (respectively, $C^E = \bigcup_j d_{e_j}$).

Example 3.
Thus in our running blocks world example, we can write down the rough description of the concepts "stable" and "belongs to Tom". Applying Algorithm 2 we find that for the dl-cut described in Example 2:

$$stable(x)^C = \{shape(x, cube), shape(x, pyramid)\}$$

$$stable(x)^E = \{shape(x, cube), shape(x, pyramid), shape(x, prism)\}$$

Thus:

$$stable(x)^R = \Big[\{shape(x, cube), shape(x, pyramid)\}, \{shape(x, cube), shape(x, pyramid), shape(x, prism)\}\Big]$$

Similarly since the only objects that are known not to belong to Tom are prisms, we have:

$$belongs\_to\_Tom(x)^C = \{shape(x, ball) \wedge material(x, wood), shape(x, pyramid)\}$$

$$belongs\_to\_Tom(x)^E = \{shape(x, ball) \wedge material(x, wood), shape(x, pyramid), shape(x, cube)\}$$

Thus:

$$belongs\_to\_Tom(x)^R = \Big[\{shape(x, ball) \wedge material(x, wood), shape(x, pyramid)\}, \{shape(x, ball) \wedge material(x, wood), shape(x, pyramid), shape(x, cube)\}\Big]$$

□

4 Reasoning with Rough Concepts

The work presented in previous sections makes it possible translate concepts from the languages of various agents into *CL*. If we consider that the central system that uses *CL* will need to reason with these concepts, a natural question arises— how can one logically manipulate rough concepts?

Well, if we take x and y as concepts roughly defined in $dl^{(CL)}$ by means of cores and envelopes it can be easily shown that for the cores and envelopes of their disjunction, conjunction, and negation, the following relations hold where the dl in the parentheses is a shorthand for $dl^{(CL)}$, and V is the set of all *wffs* in the language *CL*:

$$\begin{array}{rcl} (x \vee y)^E(dl) & = & x^E(dl) \cup y^E(dl) \\ (x \vee y)^C(dl) & \supseteq & x^C(dl) \cup y^C(dl) \\ (x \wedge y)^E(dl) & \subseteq & x^E(dl) \cap y^E(dl) \\ (x \wedge y)^C(dl) & = & x^C(dl) \cap y^C(dl) \\ (\neg x)^E(dl) & = & V - (x^C)(dl) \\ (\neg x)^C(dl) & = & V - (x^E)(dl) \end{array}$$

For instance, the envelope of a disjunction of two concepts is equal to the union of the envelopes of the individual concepts. The envelope is understood as a subset of V and is subject to unions, intersections, and subtractions; the concepts themselves are subject to disjunctions, conjunctions, and negations.

The relation of *implication* can easily be defined by means of the partial ordering $\leq$ imposed on the space of all *wffs* such that for *wffs* l_i, l_j and l_k: $l_i < l_j$ iff $l_j = l_i \cup l_k$ and $l_i \leq l_j$ iff $l_i < l_j$ or $l_i = l_j$.

Definition 4.1 (implication) *Let* $x^R = (x^C, x^E)$ *and* $y^R = (y^C, y^E)$ *be rough concepts. The operation of* implication *is defined as follows:*

$$(x^R \rightarrow y^R) \Leftrightarrow (x^R \leq y^R) \Leftrightarrow [(x^C \leq y^C) \wedge (x^E \leq y^E)]$$

From the well-known properties of partially ordered sets it follows that $(x^R \rightarrow y^R) \Leftrightarrow (\neg x^R \vee y^R) \Leftrightarrow [\neg x^E \cup y^C, \neg x^C \cup y^E]$.

$R(p)$	$[\emptyset,\emptyset]$	$[\emptyset,X]$	$[\emptyset,V]$	$[X,V]$	$[V,V]$
$RV(p)$	false	roughly false	unknown	roughly true	true

Table 1: Rough truth values ($\emptyset \subset X \subset V$)

We can consider the elements of CL, which are the rough concepts translated into CL by the algorithms in Section 3, as the set of language elements that form the basis of a formal logic. These may be propositional constants or predicate symbols. Denoting this set of language elements as $\mathcal{P}$ we can then consider the set of well formed formulae of this logic, denoting this set as $\mathcal{L}(\mathcal{P})$ [18, 20]. For any $p \in \mathcal{L}(\mathcal{P})$ we define the *rough measure* $R(p)$ of p which is the rough description of the concept or combination of concepts that correspond to p. More precisely:

$$R(p) = \left[p^{\leq C}, p^{\geq E}\right]$$

where $p^{\leq C}$ is the lower bound on the core of p and $p^{\geq E}$ is the upper bound on the envelope.

Example 4.
To delve a little further into our blocks world example,consider the compound concept p which represents $stable(x) \vee belongs\_to\_Tom(x)$. Now:

$$stable(x)^R = \Big[\{shape(x,cube), shape(x,pyramid)\}, \{shape(x,cube), shape(x,pyramid), shape(x,prism)\}\Big]$$

$$belongs\_to\_Tom(x)^R = \Big[\{shape(x,ball) \wedge material(x,wood), shape(x,pyramid)\}, \{shape(x,ball) \wedge material(x,wood), shape(x,pyramid), shape(x,cube)\}\Big]$$

so that:

$$R(p) = \Big[\{shape(x,ball) \wedge material(x,wood), shape(x,cube), shape(x,pyramid)\}, \{shape(x,ball) \wedge material(x,wood), shape(x,cube), shape(x,prism), shape(x,pyramid)\}\Big]$$

□

One way of interpreting the rough measure of an element $p \in \mathcal{L}(\mathcal{P})$ is the degree to which p is true in the universe of rough concepts. That is how universally it is true amongst the rough concepts. Obviously, for $p^C = p^E = \mathcal{L}(\mathcal{P})$, the proposition is always true and we define the *rough truth measure* $RV(p) = true(t)$. For $p^C = p^E = \emptyset$ the proposition is always false. Three other important truth values of $R(p)$ may be posited, and these are summarized in Table 1 (for more detailed discussion, see Parsons et al. [20]).

The symbolic values in Table 1 indicate to what degree a proposition is true in V. However, the pair $R(p) = [p^C, p^E]$ can also be understood as a more general measure since it explicitly determines in what part of the universe of rough concepts the proposition is true, roughly true and so on. More specifically, the expression $R(p) = [p^C, p^E]$ says that p is true in p^C and roughly true in p^E. Similar considerations enable us to define a truth-ordering on the set of propositions.

Definition 4.2 (truth ordering)
Let p_1 and p_2 be propositions. We say that p_1 is more true *than p_2 iff $p_1^C \supseteq p_2^C$ and $p_1^E \supseteq p_2^E$.*

By now this section has introduced a rough measure of truth, a set of basic symbolic truth values, the scope of validity of a proposition, and the ordering of propositions based on their truth value. With this background, we can study what happens with the truth measure if we subject the formulae of $\mathcal{L}(\mathcal{P})$ to rules of inference. This is expressed by Theorem 4.1 which is proved in the Appendix:

Theorem 4.1 *For formuale p, q and $r \in \mathcal{L}(\mathcal{P})$, variable x and a constant symbol a, modus ponens (a), modus tollens (b), resolution (c), syllogism (d), and universal instantiation (e) have the following effect on rough descriptions:*

$$a)\quad \begin{array}{lcl} R(p \to q) & = & [\alpha, \beta] \\ \underline{R(p)} & = & \underline{[\gamma, \delta]} \\ R(q) & = & [\alpha \cap \gamma, \beta] \end{array}$$

$$b)\quad \begin{array}{lcl} R(p \to q) & = & [\alpha, \beta] \\ \underline{R(\neg q)} & = & \underline{[\gamma, \delta]} \\ R(\neg p) & = & [\alpha \cap \gamma, \beta] \end{array}$$

$$c)\quad \begin{array}{lcl} R(p \vee q) & = & [\alpha, \beta] \\ \underline{R(\neg p \vee r)} & = & \underline{[\gamma, \delta]} \\ R(q \vee r) & = & [\alpha \cap \gamma, \beta \cup \delta] \end{array}$$

$$d)\quad \begin{array}{lcl} R(p \to q) & = & [\alpha, \beta] \\ \underline{R(q \to r)} & = & \underline{[\gamma, \delta]} \\ R(p \to r) & = & [\alpha \cap \gamma, \beta \cup \delta] \end{array}$$

$$e)\quad \begin{array}{lcl} R(\forall x P(x)) & = & [\alpha, \beta] \\ \underline{} & & \underline{} \\ R(P(a)) & = & [\alpha, V] \end{array}$$

For instance, for modus ponens the theorem reads as follows:

> If the implication $p \to q$ is true in $\alpha \subseteq V$ and roughly true in $\beta \subseteq V$, and if the formula p is true in $\gamma \subseteq V$ and roughly true in $\delta \subseteq V$, then q is true in $\alpha \cap \gamma$ and roughly true in β.

In this respect, the theorem gives truth-preserving inferential rules for automated reasoning and says in what part of the universe described by *CL* the rules are really deductive.

Example 5.
Suppose that the dl-cut obtained from *CL* by Algorithm 1 consists of the following four predicates: $material(x, wood)$, $material(x, stone)$, $material(x, metal)$, and $material(x, plastic)$. Furthermore, let the background knowledge contain the decision tree from Figure 4 (see overleaf). After simplification with respect to an agent that understands the concepts *interesting* and *tedious* and whose knowledge is summarised by Figure 5 (see overleaf), the dl-cut becomes $material(x, natural)$, $material(x, metal)$, and $material(x, plastic)$, because the background knowledge says that natural material in our universe is either wood or stone and because this simplification does not interfere with the system's ability to discern the concepts *interesting* and *tedious*. The concepts *interesting* and *tedious* are then translated into *CL* as follows:

$$interesting(x)^R(dl) = \Big[\{material(x, natural)\}, \{material(x, natural), material(x, metal)\}\Big]$$

$$tedious(x)^R(dl) = \Big[\{material(x, metal)\}, \{material(x, metal), material(x, plastic)\}\Big]$$

Thus:

$$\neg interesting(x)^R(dl) = \Big[\{material(x, plastic)\}, \{material(x, plastic), material(x, metal)\}\Big]$$

and:

$$(\neg interesting(x) \to tedious(x))^R(dl) = \Big[\{material(x, natural), material(x, metal)\}, V\Big]$$

If some piece of knowledge says that, with the exception of metal objects, it is always the case that $interesting(x) \to foobar(x)$ (where $foobar(x)$ is a concept unknown to the agents) so that $(interesting(x) \to foobar(x))^R(dl) = [\{material(x, natural), material(x, plastic)\}, material(x, natural), material(x, plastic)\}]$, then it is possible to conclude, using Theorem 3.1 (a), that

$$foobar(x)^R(dl) = \Big[\{material(x, natural)\}, \{material(x, natural), material(x, plastic)\}\Big]$$

□

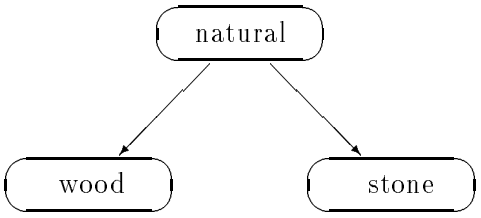


Figure 4: A classification tree

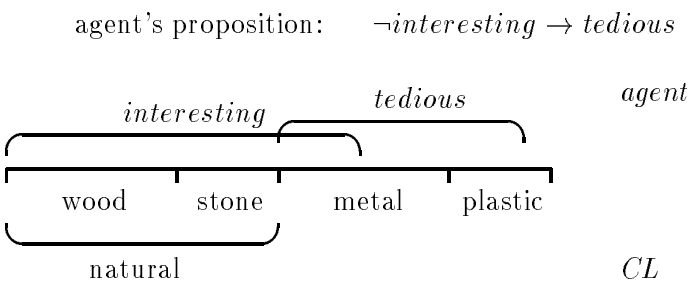


Figure 5: Translation of *interesting* and *tedious* into *CL*

5 A larger example

In this section we give a more extensive example than any so far in order to bring together all the ideas introduced in the paper.

We will consider the construction of a common language from the knowledge of two simple agents. In doing so description language cuts are built from both sets of concepts and their exemplary objects. We will then use the overall language *CL* which understands the concepts known to both agents to build rough descriptions of all concepts, and show how these might be combined in the central system to learn things that were not apparent to the individual systems. The example is kept simple to make it easy to follow, and aims to elucidate new features of our work rather than duplicate previous examples.

Since both Huhns et al. [9] and Pawlak [21] have used similar examples, it seems entirely appropriate that the agents we consider should be concerned with motor vehicles. The first agent understands three concepts, $small\_car(x)$, $fast\_car(x)$, and $slow_car(x)$which are described in terms of the objects in Table 2 (see overleaf). where the table should be read as saying, for instance, that a Rover Metro is an example of both a small car and a slow car. The second agent also knows about slow cars, but also understands the concepts $family\_car(x)$ and $van(x)$, defining these with the examples in Table 3 (see overleaf).

The context of this example is precisely the set of vehicles known by both agents, so that between them they know of every object in U. The set of concepts and objects gives us a

	$small\_car(x)$	$fast\_car(x)$	$slow\_car(x)$
$rover\_metro(x)$	$\star$		$\star$
$austin\_mini(x)$	$\star$		$\star$
$ford\_escort(x)$	$\star$		
$lotus\_eclat(x)$		$\star$	
$reliant\_robin(x)$			$\star$
$vw\_golf(x)$		$\star$	

Table 2: The concepts known by the first agent

	$family\_car(x)$	$van(x)$	$slow\_car(x)$
$ford\_escort(x) \vee ford\_cortina(x)$	$\star$		
$toyota\_corrola(x)$	$\star$		
$ford\_transit(x)$		$\star$	
$vauxhall\_astra(x)$		$\star$	
$rover\_metro(x)$			$\star$
$nissan\_micra(x)$			$\star$

Table 3: The concepts known by the second agent

deliberately simple dl-cut with which to construct CL in the hope of making the example reasonably transparent. Applying Algorithm 1 to the knowledge possessed by the first agent, we initially have $dl^{(CL)} = \emptyset$. Then, one by one we add *wffs*, each of which in this case is a simple term such as $rover\_metro(x)$. Since each *wff* is this simple, $dl^{(CL)}$ increases with each iteration:

iteration 0
$dl^{(CL)} = \emptyset$

iteration 1
$dl^{(CL)} = \{rover\_metro(x)\}$

iteration 2
$dl^{(CL)} = \{rover\_metro(x), austin\_mini(x)\}$

$\vdots$

iteration 6
$dl^{(CL)} = \{rover\_metro(x), austin\_mini(x),$
$ford\_escort(x), vw\_golf(x),$
$reliant\_robin(x), lotus\_eclat(x)\}$

This is a dl-cut that is suitable for describing all the concepts known to the first agent. We then apply the same algorithm to the *wffs* that are formed by the objects known to the second agent. All this second application of the algorithm does is to extend $dl^{(CL)}$ at every iteration, with the exception of the time the *wff* $ford\_escort(x) \vee ford\_cortina(x)$ is considered since this subsumes and thus replaces $ford\_escort(x)$, and the time that $rover\_metro(x)$ is considered since it is already in $dl^{(CL)}$. Thus we have:

iteration 7
$dl^{(CL)} = \{rover\_metro(x), austin\_mini(x),$
$ford\_escort(x) \vee ford\_cortina(x),$
$lotus\_eclat(x), reliant\_robin(x),$
$vw\_golf(x)\}$

$\vdots$

iteration 15
$dl^{(CL)} = \{rover\_metro(x), austin\_mini(x),$
$ford\_escort(x) \vee ford\_cortina(x),$
$lotus\_eclat(x), reliant\_robin(x),$
$vw\_golf(x), toyota\_corrolla(x),$
$ford\_transit(x), vauxhall\_astra(x),$
$nissan\_micra(x)\}$

Given this dl-cut, we can then use Algorithm 2 to build rough descriptions in CL of the concepts known by the two agents. In this case the concepts are quite precisely known, and we have:

$slow\_car(x)^R =$
$\Big[\{rover\_metro(x), austin\_mini(x),$
$reliant\_robin(x), nissan\_micra(x)\},$

$$\{rover\_metro(x), austin\_mini(x), reliant\_robin(x), nissan\_micra(x)\}\Big]$$

$$fast\_car(x)^R = \Big[\{lotus\_eclat(x), vw\_golf(x)\}, \{lotus\_eclat(x), vw\_golf(x)\}\Big]$$

$$van(x)^R = \Big[\{ford\_transit(x), vauxhall\_astra(x)\}, \{ford\_transit(x), vauxhall\_astra(x)\}\Big]$$

$$family\_car(x)^R = \Big[\{ford\_escort(x) \vee ford\_cortina(x), toyota\_corrolla(x)\}, \{ford\_escort(x) \vee ford\_cortina(x), toyota\_corrolla(x)\}\Big]$$

$$small\_car(x)^R = \Big[\{rover\_metro(x), austin\_mini(x)\}, \{rover\_metro(x), austin\_mini(x), ford\_escort(x) \vee ford\_cortina(x)\}\Big]$$

which neatly illustrates the point made in Section 2 about the nature of the abstraction attainable by the use of rough sets. *CL* contains knowledge of more concepts than either of the agents, it can define most of them as precisely as either agent, though it has a coarser knowledge of what constitutes a small car. However, it has more precise knowledge of the kinds of slow car than either agent.

Having established *CL* we can of course use the results of Section 4 to manipulate the rough concepts. For instance, we can evaluate the validity of the idea that all cars known to the system are either fast or slow. That is we can find the rough measure $R(fast\_car(x) \vee slow\_car(x))$, which is:

$$R(fast\_car(x) \vee slow\_car(x)) = \Big[\{rover\_metro(x), reliant\_robin(x), lotus\_eclat(x), austin\_mini(x), vw\_golf(x)\}, \{rover\_metro(x), austin\_mini(x), lotus\_eclat(x), reliant\_robin(x), vw\_golf(x)\}\Big]$$

so that the proposition is completely true in that part of the universe known to the central system that does not include vans or family cars, since the rough description of $fast\_car(x) \vee slow\_car(x)$ does not overlap with the rough description of $family\_car(x) \vee van(x)$. However, there is an overlap between the rough descriptions of $fast\_car(x) \vee slow\_car(x)$ and $small\_car(x)$. Indeed, we can test the hypothesis that $slow\_car(x) \leftrightarrow small\_car(x)$, taking this to mean that:

$$slow\_car(x) \rightarrow small\_car(x) \wedge$$
$$small\_car(x) \rightarrow slow\_car(x)$$

Now,

$$(slow\_car(x) \rightarrow small\_car(x))^R = \Big[\{rover\_metro(x), austin\_mini(x), ford\_escort(x) \vee ford\_cortina(x), lotus\_eclat(x), vw\_golf(x), toyota\_corrolla(x), ford\_transit(x), vauxhall\_astra(x)\}, \{rover\_metro(x), austin\_mini(x), ford\_escort(x) \vee ford\_cortina(x), lotus\_eclat(x), vw\_golf(x), toyota\_corrolla(x), ford\_transit(x), vauxhall\_astra(x)\}\Big]$$

and:

$$(small\_car(x) \rightarrow slow\_car(x))^R = \Big[\{rover\_metro(x), austin\_mini(x), lotus\_eclat(x), reliant\_robin(x), vw\_golf(x), toyota\_corrolla(x), ford\_transit(x), vauxhall\_astra(x), nissan\_micra(x)\}, \{rover\_metro(x), austin\_mini(x), ford\_escort(x) \vee ford\_cortina(x), lotus\_eclat(x), reliant\_robin(x), vw\_golf(x), toyota\_corrolla(x), ford\_transit(x), vauxhall\_astra(x), nissan\_micra(x)\}\Big]$$

so that:

$$(slow\_car(x) \leftrightarrow small\_car(x))^R =$$
$$\Big[\{rover\_metro(x), austin\_mini(x),$$
$$lotus\_eclat(x), vw\_golf(x),$$
$$toyota\_corrolla(x), ford\_transit(x),$$
$$vauxhall\_astra(x)\},$$
$$\{rover\_metro(x), austin\_mini(x),$$
$$ford\_escort(x) \vee ford\_cortina(x),$$
$$lotus\_eclat(x), vw\_golf(x),$$
$$toyota\_corrolla(x), ford\_transit(x),$$
$$vauxhall\_astra(x)\}\Big]$$

which implies that the proposition is less than roughly true, and less true than the proposition that being a small car implies being a slow car.

6 Conclusion

The primary goal of this work was to develop the basis of a method of translating concepts and propositions from different languages used by a set of agents. This goal was achieved with the introduction of the notions of a dl-cut and a rough concept which allow the common language to be established as a dl-cut of all the different languages used by the various agents, and this dl-cut to be used to specify the rough concepts that may be used to express the concepts manipulated by the agents. Two simple algorithms have been provided that make it possible to establish dl-cuts and rough concepts. We also addressed the problem of reasoning with the dl-cuts once they were established, giving results describing how logical reasoning may be performed with the concepts.

Future research in this direction should concentrate on the heuristics needed for the development of more efficient algorithms constructing dl-cuts from description languages. The main issues are, in this respect, the need to minimize the amount of information lost in the process of translation and the complexity of the subsumption checks and tests for overlapping *wffs*.

References

[1] Brazdil, P. B. (1989) Transfer of Knowledge Between Systems: Use of Meta-knowledge in Debugging. In Kodratoff,Y. and Hutchinson, A. (ed.): *Machine and Human Learning*. Michael Horwood, London.

[2] Brazdil, P. B. and Torgo, L. (1990) Knowledge Acquisition via Knowledge Integration. In: Wielinga,B. et al. (eds.): *Current Trends in Knowledge Acquisition*, IOS Press, Amsterdam.

[3] Buchanan, B. G. and Shortliffe, E. H. (1984) *Rule-based expert systems: the* MYCIN *experiments of the Stanford heuristic programming project*, Addison-Wesley, Reading, Mass.

[4] Cockburn, D, Varga, L. Z. and Jennings, N. (1992) Cooperating INtelligent Systems for Electricity Distribution, *Proceedings of the 12th Annual Conference of the British Computer Society Specialist Group on Expert Systems*, Cambridge, UK.

[5] Fensel, D. and van Harmelen, F. (1994) A comparison of languages which operationalize and formalise KADS models of expertise, *The Knowledge Engineering Review*, **9**, (to appear).

[6] Ginsberg, M. L. (1991) Knowledge Interchange Format: The KIF of Death. *AI Magazine*, Fall.

[7] Giunchiglia, F. and Walsh, T. (1992) A Theory of Abstraction. *Artificial Intelligence* **57**, 323–389.

[8] Hobbs, J. (1985) Granularity. *Proceedings of the International Joint Conference on Artificial Intelligence*, Los Angeles, CA.

[9] Huhns, M. N., Jacobs, N., Ksiezyk, T., Shen, W-M, Singh, M. P, and Cannata, P. E. (1993) Integrating enterprise models in Carnot, *Proceedings of the International Conference on Intelligent and Cooperative Information Systems*, Rotterdam.

[10] Imielinski, T. (1987) Domain Abstraction and Limited Reasoning. *Proceedings of the International Joint Conference on Artificial Intelligence*, Milan, Italy, 997–1003.

[11] Jennings, N. R. Wittig, T. Archon, theory and practice, in *Distributed Artificial Intelligence; Theory and Praxis*, (N. M. Avouris and L. Gasser eds.), Kluwer Academic Press, (1992).

[12] de Kleer, J. and Williams, B. C. (1987) Diagnosing multiple faults, *Artificial Intelligence*, **32**, 97–130.

[13] Kubat, M. and Parsons, S. (1993) Reasoning with Roughly Described Concepts. Technical Report 356, Institutes for Information Processing, Graz.

[14] Lenat, D. B. and Guha, R. V. (1990) *Building Large Knowledge-based Systems: Representation and Reasoning in the Cyc Project*, Addison-Wesley, Reading, Mass.

[15] Mamdani, E. H. and Assilian, S. (1975) An experiment in linguistic synthesis with a fuzzy logic controller, *International Journal of Man-Machine Studies*, **7**, 1–13.

[16] McDermott, J. (1982) R1: a rule-based configurer of computer systems, *Artificial Intelligence*, **19**, 39–88.

[17] Neches, R. Fikes, R. Finin, T. Gruber, T. Patil, R. Senator, T. and Swartout, W. R. (1991) Enabling technology for knowledge sharing, *AI Magazine*, Fall.

[18] Parsons, S. and Kubat, M. (1994) A first order logic for reasoning under uncertainty using rough sets, *Journal of Intelligent Manufacturing* (to appear).

[19] Parsons, S. and Saffiotti, A. (1993) Integrating uncertainty handling formalisms in distributed artificial intelligence, *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, Granada.

[20] Parsons, S., Kubat, M., and Dohnal, M. (1994) A Rough Set Approach to Reasoning under Uncertainty. *Journal of Experimental and Theoretical Artificial Intelligence* (to appear)

[21] Pawlak, Z. (1982) Rough Sets. *International Journal of Computer and Information Sciences 11*, 341–356

[22] Schrieber, G., Wielinga, B., and Breuka, J. (1993) KADS*: A Principled Approach to Knowledge-Based System Development*, Knowledge-Based Systems, Volume 11, Academic Press, London.

[23] Smith, S. F., Fox, M. S. and Ow, P. S. (1986) Construction and maintaining detailed production plans: investigations into the development of knowledge-based factory scheduling systems, *AI Magazine*, **7**, 4.

[24] Torgo, L. and Kubat, M. (1991) Knowledge Integration and Forgetting. *Proceedings of the Czechoslovak Conference on Artificial Intelligence*, Prague, Czechoslovakia.

[25] Zhang, C. (1992) Cooperation under uncertainty in distributed expert systems, *Artificial Intelligence*, **56**, 21–69.

Appendix

Proof of Theorem 5.1

a) Modus Ponens

$R(q) \supseteq R(p \cap q) = R((\neg p \vee q) \wedge p) = [\alpha \cap \gamma, \beta \cap \delta]$, so the lower bound on the core is $\alpha \cap \gamma$. In addition, $[\alpha, \beta] = R(\neg p \vee q) \supseteq R(q)$, so the upper bound on the envelope is β. Hence $R(q) = [\alpha \cap \gamma, \beta]$. □

b) Modus Tollens

$R(\neg p) \supseteq R(\neg p \cap \neg q) = R((\neg p \vee q) \wedge \neg q) = [\alpha \cap \gamma, \beta \cap \delta]$, so the lower bound on the core is $\alpha \cap \gamma$. In addition, $[\alpha, \beta] = R(\neg p \vee q) \supseteq R(\neg p)$, so the upper bound on the envelope is β. Hence $R(\neg p) = [\alpha \cap \gamma, \beta]$. □

c) Resolution

We know that $R(q \vee r) = R((p \vee q \vee r) \wedge (\neg p \vee q \vee r)) = [((p \vee q \vee r)^C \cap (\neg p \vee q \vee r)^C), ((p \vee q \vee r)^E \cap (\neg p \vee q \vee r)^E)]$. Now, $(p \vee q \vee r)^C \supseteq (p \vee q)^C \cup r^C = \alpha \cup r^C$ and $(\neg p \vee q \vee r)^C \supseteq (\neg p \vee r)^C \cup q^C = \gamma \cup q^C$, so the lower limit on their intersection is $\alpha \cap \gamma$. Similarly, $(p \vee q \vee r)^E = (p \vee q)^E \cup r^E = \beta \cup r^E$ and $(\neg p \vee q \vee r)^E = (\neg p \vee r)^E \cup q^E = \gamma \cup q^E$. Now, the upper limits on r^E and q^E are δ and β, respectively, so the maximum size of the envelope is $\beta \cup \delta$. □

d) Syllogism

This follows immediately from the resolution result. We have $R(\neg p \vee q) = [\alpha, \beta]$ and $R(\neg q \vee r) = [\gamma, \delta]$. Resolving these together gives $R(\neg p \vee r) = [\alpha \cap \gamma, \beta \cup \delta]$ and the result follows. □

e) Universal Instantiation

$R(\forall x_i P(x_i)) = R(P(a)) \wedge R(P(b)) \wedge \ldots \wedge R(P(n)) = [P(a)^C \cap P(b)^C \cap \ldots \cap P(n)^C, P(a)^E \cap P(b)^E \cap \ldots P(n)^E]$. Thus $P(a)^C \supseteq (\forall x_i P(x_i))^C$ and $P(a)^E \supseteq (\forall x_i P(x_i))^E$, so $R(P(a)) = [\alpha, U]$. □