In: Proc. Third European Working Session on Learning (EWSL-88), pp37-44,
Ed: Derek Sleeman, UK: Pitman (Oct 1988)
http://www.cs.utexas.edu/users/pclark/papers

Representing Arguments as Background Knowledge for Constraining Generalisation

Peter Clark (pete@turing.ac.uk)
The Turing Institute
36 N.Hanover St
Glasgow, UK

Abstract

The use of statistical measures to constrain generalisation in learning systems has proved successful in many domains, but can only be applied where large numbers of examples exist. In domains where few training examples are available, other mechanisms for constraining generalisation are required. In this paper, we propose a representation of background knowledge based on *arguments* for and against a hypothesis rather than as statements in logic or probabilistic relations, and show how it can be used to constrain generalisation from single examples (sometimes referred to as 'case-based reasoning'). Examples are characterised by the set of arguments for and against a hypothesis of interest, and the resolution of conflicting arguments in a current problem is obtained by firstly locating an old example where the same or a similar conflict occurred, then secondly generalising the solution in the old example to also apply to the new problem. This allows learning to occur in domains where few training examples exist and background knowledge is available. We provide a description of this method in logical form, and analyse the assumptions under which it is valid, its limitations and possible future extensions.

1 Introduction

This paper presents a formalism for expressing background knowledge for a problem-solving task, in which knowledge is expressed not by a set of logical axioms but as *arguments* for and against a hypothesis being true. We show how, in domains where there are too few examples to allow statistical measures for constraining generalisation, such background knowledge can be used to guide the search for solutions to problems by generalising solutions to other, previously solved problems.

In the first part, we discuss the motivation for introducing background knowledge and the requirements we desire of our representation. We discuss existing representational schemes and their suitability and limitations for assisting learning. Following this discussion, we present a representation of background knowledge as arguments and illustrate their use in learning with an example. In part four, we present a formalisation of arguments using logic, and finally discuss the assumptions under which generalisations justified by arguments are valid, the limitations of this formalism and its possible future extensions.

2 The Problem

Much work in machine learning makes use of statistical measures to select between alternative inductive hypotheses drawn on the basis of examples (for example the systems ID3 [15] and AQ11 [12]). In order that the application of statistical measures is reliable, it is necessary to have a large number of training examples available. In many domains where the constraint for a large number of examples is met, rule induction methodology has proved successful ([18, 17, 11, 16]). However to extend this methodology to other domains, the ability to represent and exploit available domain knowledge in addition to that implicit in the example set is required. This is necessary in domains where statistical evidence alone is inadequate for reliably constraining the search for generalisations. The representation and use of such domain knowledge in the generalisation process is the subject of this paper.

In contrast to work on the inductive rule learning paradigm, research in the paradigm of explanation-based learning *has* concerned itself with the use of knowledge beyond a set of examples (eg. [7, 14, 10]). Explanation-based learning (EBL) makes use of logical implication as the basic relation for expressing domain knowledge. For example, a typical statement used by an EBL system might be $\texttt{Partof}(\texttt{x},\texttt{y}) \wedge \texttt{Isa}(\texttt{y},\texttt{Bottom}) \wedge \texttt{Is}(\texttt{y},\texttt{Flat}) \Rightarrow \texttt{Stable}(\texttt{x})$ (taken from [13]). Using such knowledge, EBL techniques have been demonstrated as a

useful means for improving the efficiency of problem-solving systems. However the requirement for background knowledge in the form of relations of logical implication does not allow representation of many types of uncertainty (for example, if the consequents only follow the antecedents with a certain probability, uncertainty in that probability, etc.). This is problematic, firstly as expert knowledge is rarely without such uncertainties and thus difficult or impossible to express in logical form, and secondly because, with background knowledge as a set of unchangeable logical axioms, the assumption is being made that much of learning has been already completed. An adaptive system can only adapt knowledge which it considers to be subject to uncertainty in some way, and thus representations of background knowledge incorporating uncertainty are required if flexibility of a learning system is to be obtained. Recent research of techniques to incorporate uncertainty in EBL has concentrated on allowing the assertion and retraction of statements in the 'domain theory' of logical implications. This allows a limited form of uncertainty to be represented, namely that the truth of the logical relations in the domain theory is subject to question (similar to the approach taken in truth maintenance systems [6, 5]). This allows more scope for learning (eg. [9, 19]), and is sometimes referred to as 'reasoning with an imperfect domain theory' [13]. However, this still does not allow representation of degrees of belief in the truth of a logical implication or of probabilistic implication; statements of the form 'x is usually true', although a common part of much expert knowledge, are still unrepresentable within this framework.

An alternative representation of background knowledge is to use some form of probabilistic representation. However, this too has its drawbacks which has led us to look at other representational schemes. Most importantly, although clearly defined updating procedures for probabilistic models of a domain exist, there is not as yet any clear theory about how to specify such models in the first place. Difficulties in formulating Bayesian models from expert knowledge have led us to look to other representational schemes not requiring precise probability measures to be supplied. A second problem is that many probabilistic representations are only 'proof functional' (ie. able to calculate a value for $\mathtt{A} \wedge \mathtt{B}$ given $\mathtt{A}$ and $\mathtt{B}$ [1]) by making assumptions of independence generally not justified in many domains. Cheeseman [2] and the subsequent debate provides an excellent discussion on the issue of Bayesian and other probabilistic representations for learning systems.

This has led us to propose a representation of background knowledge based on the use of 'arguments' for and against a hypothesis being true as a means of representing uncertain background knowledge which can be used for constraining generalisation from examples and avoids the use of numerical measures of probability. This latter aim is addressed by dealing only with (possibly partial) knowledge of the *relative* strengths of arguments, rather than any absolute measure of strength (as might be represented by a number) which would require total knowledge of the relative strengths of all arguments. In addition we are attempting to formalise the use of arguments by experts for and against hypotheses, and their frequent referral to previous similar cases, as part of their problem-solving technique. In this paper we define the notion of an argument formally and show how background knowledge in this form can be used to constrain generalisation in situations where there is insufficient statistical evidence to enforce adequate constraint.

3 A Representation of Arguments

3.1 Properties of Arguments

Here we define the basic properties of arguments for our representation. Following this, we give an example of how arguments can then be used to generalise a solution from an old example to also apply to a new problem, and then in the next section provide a formalisation of these properties using logic.

Arguments are considered as form of implication, which we denote by $\mathtt{A} \xrightarrow{\text{arg}} \mathtt{B}$. If $\mathtt{A}$ is true then we say that argument $\mathtt{A} \xrightarrow{\text{arg}} \mathtt{B}$ *supports* (is 'for') $\mathtt{B}$. This implication has different semantics from logical implication ($\mathtt{A} \Rightarrow \mathtt{B}$), as follows:

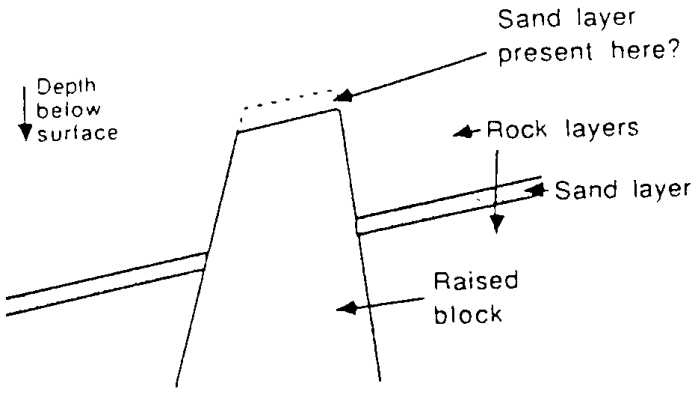
1. Where there are only arguments for $\mathtt{B}$, and none against $\mathtt{B}$ (ie. none for $\neg\mathtt{B}$), then $\mathtt{B}$ is true
2. Where there some arguments for $\mathtt{B}$ and some against $\mathtt{B}$, we have a *conflict*. Conflicts cannot be resolved (ie. result in $\mathtt{B}$ or $\neg\mathtt{B}$ being concluded) on the basis of arguments alone.
3. The same conflict will always resolve in the same way. Thus, if we have a previous case where the same conflict applied, and we know the outcome of that conflict (either $\mathtt{B}$ or $\neg\mathtt{B}$ found true), then we can infer the same outcome will also occur in the current case
4. Adding an argument in favour of a fact already known to be true cannot make it false (and vice versa)

In section 5, we discuss the underlying assumptions these properties are based on, and the conditions under which these assumptions are valid. First though, we provide an example of using arguments in reasoning.

3.2 Example

For the purposes of explanation we present an example from the domain of geology. Consider the seismic

Figure 1: A seismic cross-section showing a raised fault block



cross-section shown in figure 1, in which a geologist is interested in determining whether there is a layer of oil-bearing sand over the raised block. A geologist unfamiliar with the region may know the various factors which suggest or do not suggest that there is sand over the block, but does not know enough about the region to determine how these various 'arguments' for or against the hypothesis interact. Examples of such arguments might be, for example:

$$
\begin{array}{lcl}
\#1 : \texttt{Over\_block(x)} & \xrightarrow{\text{arg}} & \neg\texttt{Sand\_at(x)} \\
\#2 : \texttt{Sand\_nearby(x)} & \xrightarrow{\text{arg}} & \texttt{Sand\_at(x)} \\
\#3 : \texttt{Late\_fault(x)} & \xrightarrow{\text{arg}} & \texttt{Sand\_at(x)} \\
\#4 : \neg\texttt{Late\_fault(x)} & \xrightarrow{\text{arg}} & \neg\texttt{Sand\_at(x)} \\
\#5 : \texttt{Environment(x,d)} \wedge & & \\
\quad \texttt{Unfavourable(d)} & \xrightarrow{\text{arg}} & \neg\texttt{Sand\_at(x)} \\
\#6 : \texttt{Fault\_size(x,Small)} & \xrightarrow{\text{arg}} & \texttt{Late\_fault(x)}
\end{array}
$$

As a consequence, from known observations, the geologist can accumulate supporting arguments for and against a case, but as yet be unable to resolve it. In order to find a resolution to the problem, he or she will then search for *already drilled* wells similar to the one under consideration in the nearby region and observe how a similar conflict of arguments was resolved in those cases. Then, using this knowledge, the geologist hypothesises that their resolution will also be the same in the current case. In this way, we are using arguments to determine which previous solutions can be generalised to also apply to the current case. Our semantics of arguments state that only generalisation from previous cases where a similar argument conflict occurred can be made.

We can view this process of inferring a solution from an old case also applies to a new case as occurring in two stages:

1. Cases with known outcomes supply information about how to resolve conflicting sets of arguments
2. Information about how to resolve conflicting sets of arguments allow the prediction of outcomes in new cases

We introduce the relation $\texttt{Stronger}(\texttt{a}_\texttt{B}, \texttt{a}_{\neg\texttt{B}})$ to describe this, where $\texttt{a}_\texttt{B}$ is the set of all arguments *for* some fact B (ie. all $\texttt{A} \xrightarrow{\text{arg}} \texttt{B}$ where A is true) and $\texttt{a}_{\neg\texttt{B}}$ is the set of all arguments *against* B (ie. for $\neg$B). This predicate describes the relative strengths of different sets of arguments. Initially, this knowledge about argument sets may be partially or even fully unknown. However, as examples with known outcomes are encountered, such knowledge can be gradually learned and applied to known cases.

Consider, for example, the geologist knows the following facts for the to-be-drilled well (Well1 say):

$$
\begin{array}{l}
\texttt{Over\_block(Well1)} \\
\texttt{Sand\_nearby(Well1)} \\
\texttt{Environment(Well1,Paleo)} \\
\texttt{Unfavourable(Paleo)} \\
\texttt{Fault\_size(Well,Small)}
\end{array}
$$

As a consequence, the arguments *for* Sand_at(Well1) will be the set:

$$\{\#2, \#3\}$$

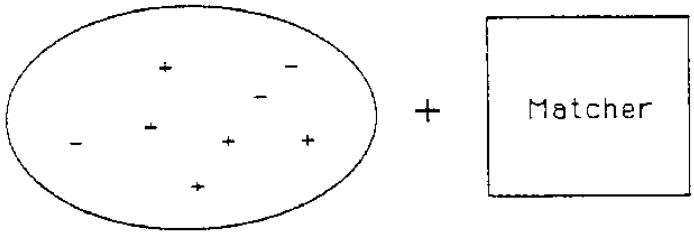
Similarly the arguments against Sand_at(Well1) are:

$$\{\#1, \#5\}$$

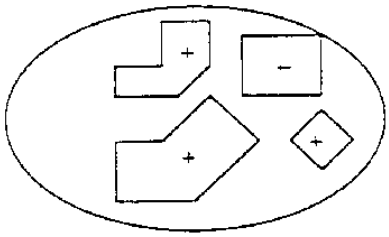
Now:

1. If we are *told* that, in fact, Sand_at(Well1) is true, then we can infer the set of arguments 'for' is stronger than the set of arguments 'against', ie. that
   $$\texttt{Stronger}(\{\#2, \#3\}, \{\#1, \#5\})$$
   was true.
2. If we wished to *find* whether Sand_at(Well1) was true, we would search for a previous case where similar conflicting argument sets *also* applied, and observe how they were resolved.
   A previous case where an identical conflict occurred can thus be used to resolve a current conflict. In addition, because arguments for a fact increase evidence for a fact (by definition), a previous case where a subset of the 'for' arguments was found stronger than a superset of the arguments 'against' (and vice versa) will also resolve the conflict. This property is an important part of the semantics of arguments, for example the following examples of known Stronger relation (possibly acquired from previous cases) are also sufficient to resolve the above conflict:
   $$\texttt{Stronger}(\{\#2\}, \{\#1, \#5\})$$
   $$\texttt{Stronger}(\{\#2, \#3\}, \{\#1, \#5, \#7, \#8\})$$

Figure 2: Case-Based and Rule-Based Concept Representation Methods



CBR : Examples + matching algorithm



Rules: concept boundaries explicitly delineated in example space

We can view this representation as characterising a case in our knowledge base not by the primitive features which it contains, but by the sets of arguments for and against the hypothesis of interest. As a consequence, we are allowing our domain knowledge to determine which previous solutions can also be applied to a new case. Our domain knowledge provides arguments, and past cases allow the incremental learning of how conflicting argument sets should be resolved.

3.3 Case-Based Reasoning

In section 3.1, we described the properties which we have assigned to the argument relation $\xrightarrow{\text{arg}}$. In particular, we declared that the same conflict betweeen sets of arguments would always resolve in the same way. Thus, given a conflict between argument sets which we wish to resolve, we can find how to resolve the conflict by observing the resolution of a similar conflict in a previous case, where the resolution is known.

This method of locating a similar previous case then transferring the solution to a current case is often referred to as a process of *case-based reasoning* (CBR). This contrasts with the 'rule induction' methodology of delineating concept boundaries explicitly before a run-time classification task is at hand. This is shown schematically in Figure ??. It is important to note the relation of case-based reasoning to rule induction as follows:

1. In CBR, generalisations are sought for at run-time when a particular classsification task is to be solved. Rule induction methods enumerate generalisations of examples during an induction phase, performed before run-time classification tasks.

2. CBR systems to not explicitly delineate the boundaries (in example space) of generalisations they form; they merely ensure generalisations cover a previous and current case. Rule induction methods do explicitly delineate concept boundaries, and as such have a 'bias' towards those concepts where the concept representation language permits concise description of those boundaries.

3. There are corresponding trade-offs between CBR and rule induction concerning memory usage and efficiency.

For a more detailed discussion of these relationships, see [3]. It should be noted that, instead of performing 'case-based reasoning' to use arguments, we could alternatively form an explicit representation of a concept boundary given a previous case and a set of arguments and use this for classifying new cases rather than use a matching algorithm. However, such a representation is likely to be cumbersome (eg. using a concept representation language such as that of AQ11) and computationally expensive to form. In addition, should the concept boundary need to be recalculated due to the addition of new examples, the computational cost in calculating the original boundary will have been wasted.

4 Representation and Semantics

4.1 Preliminaries

We now provide a description of the semantics of arguments using logic. In the following, we use the following notation:

$\mathtt{x}, \mathtt{y}, \ldots$	variables
$\underline{\mathtt{x}}, \underline{\mathtt{y}}, \ldots$	sets of variables
$\mathtt{C}, \mathtt{D}, \ldots$	constants
$\underline{\mathtt{C}}, \underline{\mathtt{D}}, \ldots$	sets of constants

When we discuss arguments for and against some fact's truth, we are actually making a meta-level statement about that fact. When referring to a fact itself (rather than its truth value), we follow the convention of Genesereth and Nilsson of writing the fact in quotes [8]. This convention is as follows:

`G(C)`	the value of predicate `G(C)` (true/false)
`"G(C)"`	the predicate `G(C)` itself
`"G(<x>)"`	the predicate `G(x)` with `x`'s *value* substituted in (<, > denote unquoting). (eg. if `x` = `Fred`, then `"G(<x>)"="G(Fred)"`).

4.2 Notation

The user expresses domain knowledge about arguments in using predicate *schema*[1] of the form below. To translate the semantics of such a schema into logic, we introduce the relation $\texttt{Arg}(\texttt{N},\texttt{G})$ meaning "argument N applies in support of (ie. 'for') G":

$$\begin{array}{c}\texttt{N} : \texttt{F}(\underline{\texttt{x}},\underline{\texttt{y}}) \xrightarrow{\text{arg}} \texttt{G}(\underline{\texttt{x}}) \\ \text{is a notation for} \\ \forall \underline{\texttt{x}}\,(\ (\exists \underline{\texttt{y}}\ \texttt{F}(\underline{\texttt{x}},\underline{\texttt{y}})) \Leftrightarrow \texttt{Arg}(\texttt{N},\texttt{"G(<}\underline{\texttt{x}}\texttt{>)"})\ ) \\ \text{loosely translated as:} \\ \text{"}\textbf{if}\text{ the condition F is true} \\ \textbf{then}\text{ argument N supports (is 'for') G"}\end{array} \quad (1)$$

In the argument schema, $\texttt{G}(\underline{\texttt{x}})$ denotes a single (possibly negated) n-ary relation and $\underline{\texttt{x}}$ the set of free variables it contains, $\texttt{F}(\underline{\texttt{x}},\underline{\texttt{y}})$ denotes a boolean combination of relations with free variables $\underline{\texttt{x}}$ (shared with G) and $\underline{\texttt{y}}$ (unshared), and N is a label (eg. a number) for the relation. This is interpreted as meaning that, for all $\texttt{G}(\underline{\texttt{x}})$ where the condition $\texttt{F}(\underline{\texttt{x}},\underline{\texttt{y}})$ is true, argument N supports (is 'for') the conclusion $\texttt{G}(\underline{\texttt{x}})$. For example the argument

$$\#1 : \texttt{Owns\_car}(\texttt{x},\texttt{c}) \wedge \texttt{Big}(\texttt{c}) \xrightarrow{\text{arg}} \texttt{Rich}(\texttt{x})$$

expresses that if someone owns a big car then this is a supporting argument for their being rich. If we knew that, say, $\texttt{Owns\_car}(\texttt{Fred},\texttt{Lotus})$ and $\texttt{Big}(\texttt{Lotus})$ then from equation 1 we conclude

$$\texttt{Arg(\#1,"Rich(Fred)")}$$

Similarly, we can now collect the set of *all* arguments which support some fact $\texttt{G}(\underline{\texttt{C}})$. We denote this set using the predicate Args as follows:

$$\begin{array}{c}\forall \texttt{a}_\texttt{G}, \underline{\texttt{x}}\ \texttt{Args}(\texttt{a}_\texttt{G},\texttt{"G(<}\underline{\texttt{x}}\texttt{>)"}) \\ \textit{iff}\ \ \texttt{a}_\texttt{G} = \{\texttt{n} | \texttt{Arg}(\texttt{n},\texttt{"G(<}\underline{\texttt{x}}\texttt{>)"})\}\end{array}$$

4.3 Semantics of Arguments

We now consider the semantics of arguments, based on reasoning with the sets of arguments $\texttt{a}_\texttt{G}$ 'for' and $\texttt{a}_{\neg\texttt{G}}$ 'against' some fact $\texttt{G}(\underline{\texttt{C}})$.

Firstly, we say that sets of arguments for and against a conclusion will completely *determine* the truth of that conclusion. In other words, conflicts involving the same sets of arguments will always result in the same conclusion. This can be expressed in logic as:

$$\begin{array}{r}\forall \texttt{a}_\texttt{G}, \texttt{a}_{\neg\texttt{G}}((\forall \underline{\texttt{x}}\ \texttt{Args}(\texttt{a}_\texttt{G},\texttt{"G(<}\underline{\texttt{x}}\texttt{>)"}) \wedge \texttt{Args}(\texttt{a}_{\neg\texttt{G}},\texttt{"}\neg\texttt{G(<}\underline{\texttt{x}}\texttt{>)"}) \Rightarrow\ \texttt{G}(\underline{\texttt{x}})) \\ \vee (\forall \underline{\texttt{x}}\ \texttt{Args}(\texttt{a}_\texttt{G},\texttt{"G(<}\underline{\texttt{x}}\texttt{>)"}) \wedge \texttt{Args}(\texttt{a}_{\neg\texttt{G}},\texttt{"}\neg\texttt{G(<}\underline{\texttt{x}}\texttt{>)"}) \Rightarrow \neg\texttt{G}(\underline{\texttt{x}})))\end{array}$$

loosely translated as:
"For all cases with the same arguments for and against G,
either G will always be true
or G will always be false"[2]

If all the $\texttt{G}(\underline{\texttt{x}})$s are true, we say that the set of arguments $\texttt{a}_\texttt{G}$ is *stronger* than the set $\texttt{a}_{\neg\texttt{G}}$, and vice versa if all the $\texttt{G}(\underline{\texttt{x}})$s are false. In order to aid comprehensibility and ease bookkeeping operations within the system, we introduce a relation Stronger to express this fact:

for all $\texttt{G}(\underline{\texttt{C}})$ with sets of arguments $\texttt{a}_\texttt{G}$ 'for' and $\texttt{a}_{\neg\texttt{G}}$ 'against':

$$\left.\begin{array}{rcl}\texttt{G}(\underline{\texttt{C}}) & \Leftrightarrow & \texttt{Stronger}(\texttt{a}_\texttt{G},\texttt{a}_{\neg\texttt{G}}) \\ \neg\texttt{G}(\underline{\texttt{C}}) & \Leftrightarrow & \texttt{Stronger}(\texttt{a}_{\neg\texttt{G}},\texttt{a}_\texttt{G})\end{array}\right\} \quad (2)$$

Thus, new cases provide new knowledge of the Stronger relation between argument sets, and this can conversely be used to resolve sets of conflicting arguments for new cases.

With this definition, we can observe how to resolve conflicts between argument sets by locating cases where an identical conflict has occurred in the past. We now express a second semantic property of arguments, namely that adding an argument in favour of some fact $\texttt{G}(\underline{\texttt{x}})$ can only 'increase the strength' of evidence for $\texttt{G}(\underline{\texttt{x}})$ (ie. cannot change its truth value). We can express this simply as follows:

$$\begin{array}{c}\forall \texttt{a}_\texttt{G},\texttt{a}_{\neg\texttt{G}},\texttt{a}'_\texttt{G},\texttt{a}'_{\neg\texttt{G}}\ \ \texttt{Stronger}(\texttt{a}_\texttt{G},\texttt{a}_{\neg\texttt{G}}) \Rightarrow \texttt{Stronger}(\texttt{a}'_\texttt{G},\texttt{a}'_{\neg\texttt{G}}) \\ \texttt{where}\ \texttt{a}'_\texttt{G} \supseteq \texttt{a}_\texttt{G}, \texttt{a}'_{\neg\texttt{G}} \subseteq \texttt{a}_{\neg\texttt{G}}\end{array} \quad (3)$$

It should be noted that the Stronger relation is not assumed to be transitive[3] and thus does not define an ordering on sets of arguments. This is because we are trying to model an expert's reasoning, where sometimes transitivity of arguments is not observed. Instead, we only compare cases with either identical arguments, or argument pairs based on the subset-superset relation above (eqn 3).

Finally we state that any set of arguments is at least stronger than no arguments:

$$\forall \texttt{a}_\texttt{G}\ \ \texttt{a}_\texttt{G} \neq \{\} \Rightarrow \texttt{Stronger}(\texttt{a}_\texttt{G},\{\}) \quad (4)$$

and that the relation is antisymmetric:

$$\forall \texttt{a}_\texttt{G}, \texttt{a}_{\neg\texttt{G}}\ \ \texttt{Stronger}(\texttt{a}_\texttt{G},\texttt{a}_{\neg\texttt{G}}) \Leftrightarrow \neg\texttt{Stronger}(\texttt{a}_{\neg\texttt{G}},\texttt{a}_\texttt{G})$$

[1] A predicate schema is obtained from a sentence (closed formula) by optionally replacing object constants with variables and optionally removing quantification from variables. As a consequence, a schema may contain free variables

[2] This can be expressed using Davies and Russell's notation for determinations [4] as:

$$\texttt{Args}(\texttt{a}_\texttt{G},\texttt{"G(<}\underline{\texttt{x}}\texttt{>)"}) \wedge \texttt{Args}(\texttt{a}_{\neg\texttt{G}},\texttt{"}\neg\texttt{G(<}\underline{\texttt{x}}\texttt{>)"}) \succ \texttt{i G}(\underline{\texttt{x}})$$

i being a *polar variable* representing the truth value of $\texttt{G}(\underline{\texttt{x}})$.

[3] ie. $\texttt{Stronger}(\texttt{A},\texttt{B}) \wedge \texttt{Stronger}(\texttt{B},\texttt{C}) \not\Rightarrow \texttt{Stronger}(\texttt{A},\texttt{C})$

4.4 Computation with Arguments

The semantics of arguments are defined in terms of the sets $\mathtt{a}_{\mathtt{G}}$ 'for' and $\mathtt{a}_{\neg\mathtt{G}}$ 'against' some fact $\mathtt{G}(\underline{\mathtt{C}})$. However, in a practical situation, there will in general be the problem that we may not be able to determine these sets from our existing theory. This will occur if neither the truth nor falsity of an argument's condition can be determined, either due to limited computational resources or incompleteness of the theory. In order to cope with this problem, we show how the semantics can be re-written in terms of sets of arguments which *are* findable by the reasoning system.

Consider we have the following four arguments

$$\begin{array}{c}\#1 : \mathtt{A}(\mathtt{x}) \xrightarrow{\text{arg}} \mathtt{G}(\mathtt{x}) \\ \#2 : \mathtt{B}(\mathtt{x}) \xrightarrow{\text{arg}} \mathtt{G}(\mathtt{x}) \\ \#3 : \mathtt{C}(\mathtt{x}) \xrightarrow{\text{arg}} \mathtt{G}(\mathtt{x}) \\ \#4 : \mathtt{D}(\mathtt{x}) \xrightarrow{\text{arg}} \neg\mathtt{G}(\mathtt{x})\end{array}$$

and also know the facts:

$$\begin{array}{c}\mathtt{A}(\mathtt{Fred}) \\ \neg\mathtt{C}(\mathtt{Fred})\end{array}$$

Here we have insufficient information to tell whether #2 is an argument for $\mathtt{G}(\mathtt{Fred})$ or not, as we are unable to tell whether the condition $\mathtt{B}(\mathtt{Fred})$ is true. Consequently, we cannot determine the set $\mathtt{a}_{\mathtt{G}}$ of arguments 'for' $\mathtt{G}(\mathtt{Fred})$. However, we *can* determine lower and upper bounds on this set $\mathtt{a}_{\mathtt{G}}$ by finding those sets of arguments which can be shown to apply and can't be shown to not apply respectively. For example, we can easily conclude from the above that:

$$\{\#1\} \subseteq \mathtt{a}_{\mathtt{G}} \subseteq \{\#1, \#2\}$$

Consequently, we will now show how the above semantics for arguments can be simply rewritten in terms of lower and upper bounds. Following from this, we show that the tighter these bounds, the more likely we are to be able to draw conclusions about new cases from old ones. Thus we show there is a trade-off between expending resources and the ability to reach conclusions.

4.4.1 Reasoning with Bounds on Arguments

Consider we have established lower and upper bounds on the set of arguments $\mathtt{a}_{\mathtt{G}}$ for some fact $\mathtt{G}(\underline{\mathtt{C}})$. We denote these bounds $\mathtt{a}^-_{\mathtt{G}}$ and $\mathtt{a}^+_{\mathtt{G}}$ respectively, ie:

$$\mathtt{a}^-_{\mathtt{G}} \subseteq \mathtt{a}_{\mathtt{G}} \subseteq \mathtt{a}^+_{\mathtt{G}}$$

Using equation 3, we can rewrite equation 2 as follows. Firstly, if we wish to know the truth of $\mathtt{G}(\underline{\mathtt{C}})$, then the following `Stronger` relations will allow us to deduce it:

$$\left.\begin{array}{rcl}\mathtt{Stronger}(\mathtt{a}^-_{\mathtt{G}}, \mathtt{a}^+_{\neg\mathtt{G}}) & \Rightarrow & \mathtt{G}(\underline{\mathtt{C}}) \\ \mathtt{Stronger}(\mathtt{a}^-_{\neg\mathtt{G}}, \mathtt{a}^+_{\mathtt{G}}) & \Rightarrow & \neg\mathtt{G}(\underline{\mathtt{C}})\end{array}\right\} \quad (5)$$

This follows straightforwardly from eqn 3 as

$$\mathtt{Stronger}(\mathtt{a}^-_{\mathtt{G}}, \mathtt{a}^+_{\neg\mathtt{G}}) \Rightarrow \mathtt{Stronger}(\mathtt{a}_{\mathtt{G}}, \mathtt{a}_{\neg\mathtt{G}})$$

Conversely, if we already *know* the truth of a particular case $\mathtt{G}(\underline{\mathtt{C}})$, we can infer knowledge about the `Stronger` relations:

$$\left.\begin{array}{rcl}\mathtt{G}(\underline{\mathtt{C}}) & \Rightarrow & \mathtt{Stronger}(\mathtt{a}^+_{\mathtt{G}}, \mathtt{a}^-_{\neg\mathtt{G}}) \\ \neg\mathtt{G}(\underline{\mathtt{C}}) & \Rightarrow & \mathtt{Stronger}(\mathtt{a}^+_{\neg\mathtt{G}}, \mathtt{a}^-_{\mathtt{G}})\end{array}\right\} \quad (6)$$

These equations can be intuitively understood as corresponding to the most conservative estimates we can make of the arguments $\mathtt{a}_{\mathtt{G}}$ and $\mathtt{a}_{\neg\mathtt{G}}$ for and against a fact. For example, if at least all the arguments definitely shown to be in favour of a fact ($\mathtt{a}^-_{\mathtt{G}}$) are stronger than all the arguments that could possibly be against ($\mathtt{a}^+_{\neg\mathtt{G}}$), it must be the case that the fact is true; the actual set of arguments 'for' ($\mathtt{a}_{\mathtt{G}}$) can only be larger and the set 'against' ($\mathtt{a}_{\neg\mathtt{G}}$) only smaller, which only strengthens the case for $\mathtt{G}(\underline{\mathtt{C}})$ being true (eqn 3).

4.4.2 Computational Trade-Off

Given lower and upper bounds on the set of arguments for a fact, we can use the equations 5 to find whether the fact is indeed true. The tighter the bounds the reasoning system can draw, the more likely it is that a previous case can be found which will offer a solution to the conflict of argument sets in the current case. This can be seen from equations 5 and 3, as the larger $\mathtt{a}^-_{\mathtt{G}}$ and the smaller $\mathtt{a}^+_{\mathtt{G}}$ the more likely it is that the `Stronger` relation can be inferred from known `Stronger` relations (eqn 3). Thus, where the set of arguments which apply to a case cannot be exactly determined, the reasoning system should expend effort to bound this set. For many applications, exhaustive enumeration of known and possible arguments would be the most practical approach to take.

5 Assumptions and their Validity

We have presented a representation of background knowledge and described how it can be used for constraining generalisation from single examples. Here we discuss the assumptions under which such generalisations are valid, and following this we discuss future extensions required to this work.

The basic assumptions the formalism makes are that:

- Argument conflicts will always resolve the same way
- Adding arguments for a conclusion always strengthens the case for it.

Underlying these two assumptions are several strong, basic assumptions:

1. We assume that the data we have about a given case is sufficient to *completely determine* the values of any unknown features we are interested in

2. We assume that the set of arguments provided by the user is *sufficiently complete* to avoid the same arguments being resolved in different ways for different cases. If there is some additional argument which may apply in a conflict and is significant enough to change its outcome, but the system is unaware of it, then the assumption that conflicts known to the system will always be resolved the same way will be violated
3. We assume that the arguments provided are *correct*, such that adding an argument in favour of a case will only strengthen the case

These assumptions constrain the applications for which such reasoning is valid to those where extensive weak background knowledge and detailed information about cases are available. The domain of legal reasoning is one which is particularly suitable.

6 Conclusion and Future Work

In the absence of adequate statistical information to constrain generalisation, other methods of constraint must be employed. In this paper, we have described a representation of background knowledge in the form of *arguments* for and against a case, and a method by which such arguments can be used to justify generalising old solutions to new cases. Our aim has been to provide a representation of uncertain background knowledge which can be used to constrain generalisation in the absence of adequate statistical evidence from examples, and to avoid the requirement for explicit probabilities to be specified for the arguments comprising background knowledge.

There are several limitations to the method presented here suggesting future work which we now highlight. Most importantly, we have assumed that the same conflict between sets of arguments will always resolve itself in the same way, and it is on the basis of this assumption that we are able to justify the generalisation of old solutions to apply also to new cases. Thus we are assuming that a previous example completely removes the uncertainty presented by a conflict between sets of arguments. This assumption is often violated in real world domains for various reasons including imperfect background knowledge and limited knowledge of the particular problem at hand, and to extend the theory to relax this assumption would be an important future step. Secondly, in the theory presented we generalise on the basis of background knowledge and a single example, and thus statistical methods play no role. The theory needs to be extended to allow statistical methods to also be included. For example if the same conflict between sets arguments for a hypothesis has occurred four times before, resolve three times 'for' and once 'against', we would like to resolve the current case 'for' (or at least reason about the different outcomes in previous cases) rather than merely saying this is inconsistent with our background knowledge. Thirdly, it may be the case that there is no previous example where an argument conflict similar to a current conflict occurred. In this situation, we would like to nevertheless make some prediction for the current problem but the representation presented will not offer any method for making such a prediction. Finally, we have not considered ways in which the background knowledge itself can be extended or corrected in the light of new evidence from examples, again a feature which would be a useful extension.

Acknowledgements

This research arose out of work with Enterprise Oil plc, and I particularly thank Dave Rhodes and Tony Scurr for their involvement. Thanks also to Tim Niblett, Simon Grant and Robin Boswell for valuable discussions on earlier versions of this paper.

References

[1] A. Bundy. Probability, truth and logic: reply to cheeseman. *Computational Intelligence*, 4(1):69–70, Feb 1988.

[2] P. Cheeseman. An inquiry into computer understanding. *Computational Intelligence*, 4(1), Feb 1988. (followed by 23 commentaries).

[3] P. Clark. A comparison of exemplar-based and rule-based learning systems. In *International workshop on Machine Learning, Meta-reasoning and Logics*, pages 69–81, February 1988. (Also available as TIMLG-15, Turing Inst., Glasgow, UK).

[4] T. R. Davies and S. J. Russell. A logical approach to reasoning by analogy. In *IJCAI-87*, pages 264–270, 1987.

[5] J. DeKleer. Choices without backtracking. In *AAAI–84*, 1984.

[6] J. Doyle. A truth maintenance system. *Artificial Intelligence*, 12:231–272, 1979.

[7] R. Fikes, P. Hart, and N. Nilsson. Learning and executing generalized robot plans. In B. Webber and N. Nilsson, editors, *Readings in AI*, pages 231–249, Tioga, Palo Alto, CA, 1981.

[8] M. R. Genesereth and N. J. Nilsson. *Logical Foundations for Artificial Intelligence*. Kaufmann, Ca, 1987.

[9] R. Hall. *Learning by Failing to Explain.* Technical Report 906 (AI-TR-906), MIT AI Laboratory, 1986.

[10] J. E. Laird, P. S. Rosenbloom, and A. Newell. Chunking in soar: the anatomy of a general learning mechanism. *Machine Learning*, 1(1):11–46, 1986.

[11] R. S. Michalski and R. Chilausky. Learning by being told and learning from examples: an experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean diagnosis. *Policy Analysis and Information Systems*, 4(2):125–160, 1980.

[12] R. S. Michalski and J. Larson. *Incremental generation of VL_1 hypotheses: the underlying methodology and the description of program AQ11.* Technical Report ISG 83-5, The University of of Illinois at Urbana-Champaign, Department of Computer Science, Urbana, 1983.

[13] T. M. Mitchell, R. M. Keller, and S. T. Kedar-Cabelli. Explanation-based generalization: a unifying view. *Machine Learning Journal*, 1(1):47–80, 1986.

[14] J. Mostow and N. Bhatnagar. Failsafe – a floor planner that uses ebg to learn from its failures. In J. McDermott, editor, *IJCAI-87*, pages 249–255, Kaufmann, Ca, 1987.

[15] J. R. Quinlan. Learning efficient classification procedures and their application to chess endgames. In J. G. Carbonell, R. S. Michalski, and T. M. Mitchell, editors, *Machine Learning, vol. 1*, Tioga, Palo Alto, Ca, 1983.

[16] J. R. Quinlan, P. J. Compton, K. A. Horn, and L. Lazarus. Inductive knowledge acquisition: a case study. In *Proceedings of the second Australian Conference on the Applications of Expert Systems*, pages 183–204, New South Wales Institute of Technology, Sydney, 1986.

[17] A. Shapiro and T. Niblett. *Automatic induction of classification rules for a chess endgame*, pages 73–91. Volume 3, Pergamon, Oxford, 1982.

[18] B. Shepherd. An appraisal of a decsion tree approach to image classification. In *IJCAI-83*, Kaufmann, Ca, 1983.

[19] K. VanLehn. Learning a domain theory by completing explanations. In T. M. Mitchell, J. G. Carbonell, and R. S. Michalski, editors, *Machine Learning: A Guide to Current Research*, Kluwer, Lancaster, UK, 1986.