Learning to Relate Terms in a Multiple Agent Environment

P. Brazdil (*) S. Muggleton (**)

(*) LIACC-CIUP, Rua Campo Alegre, 823, 4100 Porto, Portugal
E-mail: pbrazdil@nccup.ctt.pt ; Later: pbrazdil@liacc.up.pt

(**) The Turing Institute, 36 North Hanover Street, Glasgow G1 2AD, UK
E-mail: steve@turing.ac.uk

Abstract

In the first part of the paper we describe how different agents can arrive at different (but overlapping) views of reality. Although the agents can cooperate when answering queries, it is often desirable to construct an integrated theory that explains 'best' a given reality. The technique of knowledge integration based on an earlier work is briefly reviewed and some shortcomings of this technique are pointed out. One of the assumptions underlying the earlier work was that all agents must use the same predicate vocabulary. Here we are concerned with the problems that can arise if this assumption does not hold. We also show how these problems can be overcome. It is shown that standard machine learning techniques can be used to acquire the meaning of other agent's concepts. The experiments described in this paper employ INTEG.3, a knowledge integration system, and GOLEM, an inductive system based on relative least general generalization.

Keywords: knowledge integration, language differences, learning concept definitions, learning unknown concepts, predicate vocabulary, learning in distributed systems.

1. Introduction

Many AI applications are inherently distributed. Consider, for example, the problem of developing a large expert system in some domain (e.g. medicine). This process usually involves consultations with various experts. The knowledge of these experts may often be quite complementary. In each consultation one could cover a somewhat different part of the domain. Facilities are needed for elicitation and analysis of knowledge from multiple experts (Boose et al., 1989). Moreover, it is necessary to develop cooperative mechanisms that enable multiple systems to work together to solve common problems. We would like also the system to improve the overall problem solving performance. As Durfee et al. (1989, p.103) have pointed out, improvements of performance may often be achieved by transferring certain knowledge (operators) from one system to another.

Whenever knowledge is generated and transferred from one subsystem to another, conflicting information can easily be generated. So far relatively little work has been done in the area of how one potential conflicts could be resolved.

The objective of our earlier work (Brazdil and Torgo, 1990) was to partly cover this gap. In our scenario we admitted the existence of several different agents, each of whom was capable of constructing theories on the basis of given data. The objective of system INTEG.3 was to construct an integrated theory from the individual theories. Our experiments have shown that the integrated theory had in general better performance that the original theories. In that work we have, however, assumed that all agents use the same predicate vocabulary. The purpose of this paper is to show what we can do if this assumption does not hold. That is, we will show how agents can overcome certain language differences automatically, without human intervention. As we shall see standard machine learning techniques can be used to acquire the meaning of other agents' concepts.

The purpose of the method described here could be stated as follows: Given two or more theories (which are assumed to belong to different agents), construct a *new theory* that includes the essential parts of the original theories, while trying to minimize possible inconsistencies and redundancies on the basis of experimental tests.

Other people have adopted a somewhat different stand. Gams (1989), for example, maintains all apparently redundant rules (or theories) within the system. He has shown that when redundant rules are taken into account, this can lead to improvements of performance. Theories that contain many redundant rules have, however, certain disadvantages over those that have

been trimmed down. They are more difficult to understand and consequently they are also more difficult to modify.

Organization of the Paper

The rest of this paper is organized as follows. First we discuss some other related work in this area. Section 2 is devoted to the description of how different agents can arrive at different (but overlapping) views of reality.

Section 3 discusses the functioning of a distributed system. It shows that whenever different agents have overlapping but non-identical knowledge, it is advantageous for the agents to cooperate. One disadvantage of the distributed solution, however, is that all agents must be ready to step in and act. The functioning of the distributed system will be impaired if one agent is non-operational.

An alternative, centralized solution is described in Section 4. This section reviews the method of knowledge integration (more details are in (Brazdil and Torgo, 1990)) and identifies some problems that can arise whenever agents use different predicate vocabulary. Section 5 describes how these problems can be overcome.

Relation to Other Work

The problem of language differences typically arises when knowledge is elicited from several (human) experts. As Shaw and Gaines (1989) have pointed out, the *same* term can have *different* meaning in different systems. They call this situation a *conflict*. *Different* terms may, however, have *identical* meanings. Shaw and Gaines call this situation a *correspondence*.

Our system INTEG.3 (Brazdil and Torgo, 1990) deals with the problem of *conflict*, but not of correspondence. The system uses empirical evidence to decide whose definition of a given concepts is 'best'. In this paper we are concerned with the problem of correspondence. Although, the meanings of *parent* and *father* are not identical, the system will establish how they are related.

Murray and Porter (1989) have developed a system PROTOKI (a prototype of a larger system KI). The system is intended to provide support when new piece of information is integrated in the existing knowledge base. The process of integration involves three main steps: recognition, elaboration and adaptation. The third step is concerned with resolution of anomalies. These

arise, for example, when conflicting solutions can be reached using different chains of reasoning. The prototype system inspects the explanation that lead to the anomaly and tries to determine the weakest premise. This premise is then modified.

On one hand Murray and Porter's work goes further than ours. So far we have not tried to integrate knowledge consisting of rules that are chained. On the other hand PROTOKI requires a great deal of domain dependent knowledge, and besides, does not attempt to deal with the problem of *correspondence*. It does not try to establish relationships between similar concepts that may already exist within the system.

2. World of Agents

Here we will be involved with a distributed system, similar to the ones discussed by Durfee et al., (1989). We assume that the system consists of a number of separate agents that can communicate. We will assume *loose coupling* between agents in the sense defined by Durfee et al., implying that the communication costs between agents are significant. This assumption has the following important implication for us: It is worth trying to integrate related pieces of knowledge within one system.

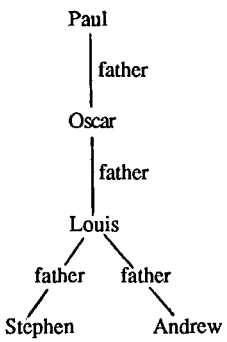
Here we will assume that each agent has certain perceptive, communicative and reasoning capabilities, but also that it is capable of inductive reasoning and integrating pieces of knowledge into one theory. That is, the agents will be capable of:

- perceiving a portion of the given world (or simulated world),
- accepting facts and rules from another agent,
- formulating queries and supplying them to another agent,
- responding to queries formulated by another agent,
- interpreting answers provided by another agent,
- inducing rules on the basis of facts,
- integrating knowledge.

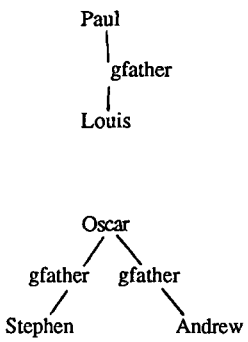
The last two capabilities differentiate our agents from those discussed by Durfee at al. Here we shall adopt a particular form of knowledge representation. All agents' knowledge will be represented in the form of facts and/or rules. This, however, does not mean that the agents must represent their knowledge in this way. Our simulation merely places certain constraints on what the agents can or cannot do.

2.1 Our Scenario

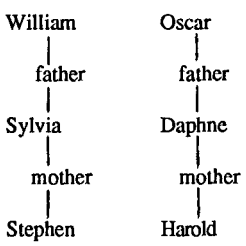
The experiments described in later sections involve two agents (A, B) and the user. The user can be viewed as another agent. Our scenario also includes a micro-world of family relations. Actually, we will distinguish various views of this micro-world. All of these views are shown in Fig.1. Note that this figure shows how *we see* the domain. The agents´ representation of this domain will be shown later.



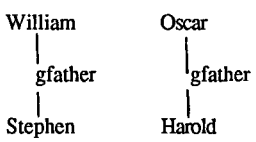
(a) Family-relations-D1



(b) Family-relations-G1



(c) Family-relations-D2



(d) Family-relations-G2

Oscar is not a grandfather of Sylvia.
Stephen is not a grandfather Oscar.
Stephen is not a grandfather Stephen.
Sylvia is not a grandfather Stephen.
Louis is not a grandfather Sylvia.
Paul is not a grandfather Andrew.

(e) Family-relations-N1 (negative facts).

Fig.1 The given micro-world of family relations and its views

The interpretation of this figure is rather obvious. The link between *Paul* and *Oscar*, for example, with the label *"father"* represents the fact that *"Paul is a father of Oscar"*. The term *gfather* is an abbreviation of *grandfather*. The microworld *Family-relations-N2* is assumed to be identical to *Family-relations-N1*.

2.2 Agent´s Differing Views of Reality

Perceptive actions provide the agents with a *description* of the relevant part of the world. This description consists of a set of predicates that are part of A´s vocabulary. Here we will assume that A´s predicate vocabulary includes the following predicates:

father (X,Y), mother (X,Y), gfather (X,Y).

After agent A has executed a perceptive action, its knowledge base will include instances of the three predicates mentioned. If A´s perceptive action has been directed towards the micro-world of *Family-relations-D1*, its knowledge base will contain the assertions:

```
father ( "Paul",   "Oscar").      father ( "Louis", "Stephen").      (D1A)
father ( "Oscar",  "Louis").      father ( "Louis",  "Andrew").
```

Similarly, perception of *Family-relations-G1* will give:

```
gfather ( "Oscar",  "Stephen").    gfather ( "Paul",   "Louis").      (G1A)
gfather ( "Oscar",  "Andrew").
```

The set G1A represents "conclusions" that we would like the agent to derive on the basis of its rules and D1A. Agent A´s view of the negative facts available *(Family-relations-N1)* will be identified by N1A.

In our scenario agent B has a somewhat different view of reality. First, let us assume that B´s predicate vocabulary includes:

parent (X,Y), male (X), female (X), gfather (X,Y),

but not *father(X,Y)* or *mother(X,Y)*. Second, let us assume that B´s attention is directed towards *Family-relations-D2*. B´s representation of the corresponding facts is:

```
parent ( "William", "Sylvia").     parent ( "Oscar", "Daphne").      (D2B)
parent ( "Sylvia", "Stephen").     parent ( "Daphne", "Harold").
male ( "William").                 male ( "Oscar").
male ( "Stephen").                 male ( "Harold" ).
female ( "Sylvia").                female ( "Daphne").
```

Agent B´s representation of *Family-relations-G2* will similarly be:

```
gfather ( "William", "Stephen").   gfather ( "Oscar", "Harold").      (G2B)
```

Agent B´s view of the negative facts *(Family-relations-N2)* will give assertions called N2B.

2.3 Construction of Different Theories of Reality

As we have mentioned earlier our agents are capable of inducing theories on the basis of given (observed) facts. The outcome of inductive process depends on a number of factors, such as:

- inductive method employed,
- the facts available,
- background knowledge,
- predicate vocabulary adopted,
- other "inductive biases".

Here we will assume that both agents use the same inductive method called GOLEM. The method is based on *Relative Least General Generalization* and is described in detail in (Muggleton & Feng, 1990).

We would like to point out that the assumption concerning the agents´ inductive method simplifies somewhat the points we want to make here. However, the methods described here are in no way based on (or limited by) this assumption.

The facts contained in the set (D1A) and the correct answer set (G1A) and the negative facts (N1A) enable agent A to generate the following definition of *´gfather´*:

```
gfather(X,Y) :- father(X,Z), father(Z,Y).                    (R1A)
```

Rule (R1A) covers all A´s positive examples of *'gfather´*, and so the facts

```
gfather ( "Oscar", "Stephen").    gfather ( "Paul", "Louis").    (G1A)
gfather ( "Oscar", "Andrew").
```

are no longer necessary for answering queries.

As B uses a different set of initial facts, and also, different predicate vocabulary, the rule generated by B is different from A´s:

```
gfather(X,Y) :- parent(X,Z), male(X), parent(Z,Y), female(Z).    (R2B)
```

This rule covers B´s positive examples of *'gfather'* and so the facts

```
gfather ( "William", "Stephen").    gfather ( "Oscar", "Harold").    (G2B)
```

are no longer necessary for answering queries.

We notice that A´s concept of *gfather(..)* is defined in terms of A´s vocabulary, that is, using the concept of *father(..)*. Similarly B´s concept is defined in terms of B´s vocabulary. In our

case this vocabulary includes the concepts *parent(..)*, *male(..)* and *female(..)*. This is the reason why agent A fails to prove, for example, the following fact (this fact appears in G2B):

gfather ("William", "Stephen")

using

gfather(X,Y) :- father(X,Z), father(Z,Y) (R1A)

and

parent ("William", "Sylvia").
parent ("Sylvia", "Stephen").
male ("William").

that is certain relevant facts from (D2B). A´s definition of *gfather(..)* contains the reference to *father(..)*, but B´s data uses the concepts *parent(..)*, *male(..)* and *female(..)*. As agents A and B use a somewhat different vocabulary, A´s definition cannot be applied to B´s data. In the next section we will see how this problem can be overcome.

3. Distributed Problem Solving

As we have seen in the previous section both agents A and B were capable of answering certain queries concerning *gfather(..)* after certain training. However, neither A nor B could answer correctly all the queries that A and B were presented with during the training phase. This suggests that something could be gained by letting both agents try to work together. In this section we will analyze this mode of operation in more detail.

In order to be able to function together as one system the agents need to be able communicate. Communication between agents is achieved using a special communication layer obeying certain rules. Its purpose is to determine when the given problem (query) should be resolved by the recipient agent (Ai) (i.e. the agent that received the query), and when it should be sent to someone else. For simplicity of exposition the user will be regarded as another agent.

Let us adopt the following rather simple rules to control the communication between agents:

(Com1) Agent Ai should attempt to solve problem P before sending it to another agent.

(Com2) If Ai succeeded in solving P, the answer should be returned to the agent that issued P.

(Com3) If Ai failed to solve P and if there is some agent Aj that has not yet attempted to solve P, P should be sent to Aj. If Aj succeeded in solving P the answer should be sent to the agent that issued P.

(Com4) If Ai failed to solve P and if all agents available have attempted to solve P without success, failure should be returned to the agent that issued P.

(Com5) Each agent should use his own theories and data when solving P (it is not allowed to interchange of theories or data at run-time).

The rules shown above assure that the given problem is not passed around unnecessarily from one agent to another. The rules have another purpose, however. They determine how a particular agent should act when trying to resolve the given problem.

The rules shown above can be used to control the behavior of the agents A and B. In the following we will use the symbols "A+B" to identify a *distributed system* with agents A and B obeying the rules (Com1)-(Com5). Let us see how this system handles one query. Suppose the user has asked agent A to determine whether

gfather ("William", "Stephen")

is true. Agent A by himself cannot answer this query, but thanks to the fact that A and B can communicate, correct answer can be given. The query is simply sent to agent B and then a positive answer is sent back to A. This answer is then returned to the user. It is easy to show that, here, the problem solving performance of the distributed system A+B exceeds the performance of individual agents.

The rules (Com1)-(Com5) do not take into account relative success rates of individual agents or the quality of individual rules. Whenever one agent has succeeded in answering the given query, its answer is returned (Com2). No attempt is made to check which answer the other agents would give. Different agents could however give conflicting answers and so some strategy is needed to decide which answer should be chosen. Some people have proposed and used a kind of voting scheme (Gams, 1989). Our method (Brazdil and Torgo, 1990) attempts to select a subset of rules which appear to be most reliable.

Distributed systems have other disadvantages, however. If we cannot a priory determine to which agent particular queries should be attributed to, all agents must be operational and ready to step in. This affects the costs involved in obtaining the answer(s). This suggests that all the rules acquired individually could be collected by one of the agents and then this agent could act alone to answer the user´s queries. In the next section we will discuss this process in detail, and also, analyze some problems that can arise whenever agents use different vocabulary.

4. Theory Integration

The aim of knowledge integration could be defined as follows. Consider all theories available and construct an integrated theory taking into account performance criteria of various constituents. The details of this algorithm are in (Brazdil and Torgo, 1990).

Suppose agent A has been designated to construct an integrated theory on the basis of the individual theories acquired earlier by agents A and B. As each theory consists of only one rule, the candidate rules set will contain the following two rules:

```
gfather(X,Y) :- father(X,Z), father(Z,Y).                    (R1A)
gfather(X,Y) :- parent(X,Z), male(X), parent(Z,Y), female(Z).  (R2B)
```

The integration algorithm needs to be supplied with available data (D1A,D2B) and required answers (G1A,G2B). Assuming that both agents have kept all the initial data, agent A needs to get only B´s data to proceed. The integration algorithm will then test the given rules on available data. In our case there are just two rules. As each rule covers only a part of the data without covering any negative examples, both rules will appear in the integrated theory.

The integrated theory enables the agent to answer correctly any query selected from set G1A or G2B. For example, the query

```
gfather ( "William",  "Stephen")
```

will succeed. Despite this apparent success, one important issue has remained unresolved. Agent A is unable to perceive B´s micro-world and act correctly.

Let us assume that agent A has the rules R1A and R2B. Let us further assume that all data is obtained using direct observations (perceptive actions). The perceptive action directed towards the micro-world *Family-relations-D2* will update A´s knowledge base with:

```
father ( "William", "Sylvia").     father ( "Oscar", "Daphne").     (D2A)
father ( "Sylvia", "Stephen").     father ( "Daphne", "Harold").
```

It can be verified that the goal

```
gfather ( "William",  "Stephen")
```

will fail. The failure is due to the fact that A´s rules and his data use *different vocabulary*. Rule R2B, for example, contains the predicates

```
parent(X,Z), male(X), parent(Z,Y), female(Z)
```

all of which are unknown concepts (for A). They do not figure among A´s primitive concepts, and moreover, agent A has no definition showing what these concepts mean. In the next section we will show how standard ML techniques can be used to acquire the definitions of such concepts.

5. Acquiring the Meaning of Unknown Concepts

In the previous section we have shown that when theories integrate knowledge of different agents, care must be taken to avoid failures. Failures are likely when agents use a somewhat different vocabulary. The rules acquired by communication can contain concepts that are simply unknown to the recipient agent.

In this section we will describe a technique that can be used to overcome these problems. We will show how agents can acquire the required definitions of unknown concepts. As we shall see standard machine learning techniques can be used to construct such definitions. One of the crucial steps in this process will be the following. The agents will attempt to describe the same "situation" so as to be able to formulate the relationships between different concepts.

The technique used here seems to be quite common in human learning. Consider, for example, mother and a child. The mother will often describe the situation the child can see. The child is able to learn the meaning of her mother´s concepts by relating the mother´s description to its own experience.

Let us apply this strategy to our scenario. Here we will use the micro-world of *Family-relations-D2* to try to define the unknown concepts. As we have seen agent A´s view of this micro-world is:

father ("William", "Sylvia").	father ("Oscar", "Daphne").	(D2A)
mother ("Sylvia", "Stephen").	mother ("Daphne", "Harold").	

B´s view of this micro-world is:

parent ("William", "Sylvia").	parent ("Oscar", "Daphne").	(D2B)
parent ("Sylvia", "Stephen").	parent ("Daphne", "Harold").	
male ("William").	male ("Oscar").	
male ("Stephen").	male (" Harold").	
female ("Sylvia").	female ("Daphne").	

The concepts that are unknown to A and whose meaning is to be defined are:

parent(X,Z), male(X), female(Z).

These facts will enable the agent A to generate the required definitions. Given these facts the inductive system will generate[1]:

```
parent (X,Z)   :- father (X,Z).                    (RI1)
parent (X,Z)   :- mother (X,Z).                    (RI2)
male (X)       :- father (X,_).                    (RI3)
female (Z)     :- mother (Z,_).                    (RI4)
```

The rules shown above relate B's concepts (e.g. *parent(..)*) to A's own concepts (i.e. *father(..)*). The new definitions enable the agent A to overcome the problem mentioned earlier. It can be verified that the goal

gfather ("William", "Stephen")

no longer fails. The solution of this goal requires rule R2B. Application of this rule generates the subgoals

parent("William",Z), male("William"), parent(Z,"Stephen"), female(Z),

all of which can be interpreted correctly thanks to the new definitions (RI1-RI4). The application of rule RI1 to the subgoal *parent("William,Z)*, for example, will produce the subgoal *father("William",Z)*. In other words, B's concept of *parent(..)* will be substituted by the concept of *father(..)*, which is one of A's primitive concepts.

5.1 Role of Interface Theory

In this paper we have been concerned with the problem of composing theories from different constituents generated by different agents. Let us come back to our scenario and examine these constituents in more detail. In particular let us look at A's theory which was completed in the last section. It consists of the following rules:

```
gfather(X,Y)   :- father(X,Z), father(Z,Y).                    (R1A)
gfather(X,Y)   :- parent(X,Z), male(X), parent(Z,Y), female(Z).  (R2B)

parent (X,Z)   :- father (X,Z).                    (RI1)
parent (X,Z)   :- mother (X,Z).                    (RI2)
male (X)       :- father (X,_).                    (RI3)
female (Z)     :- mother (Z,_).                    (RI4)
```

Rule R1A was generated by agent A's inductive subsystem on the basis of its data. Rule R2B was obtained from agent B using a process of communication.

[1] The present version of GOLEM is unable to generate the definitions of *male(X)* and *female(Z)* shown here. The point is that the system cannot learn clauses with arbitrary existential quantification, such as $\exists y \forall x$ male(x) <-- father(x,y). It is expected that a solution to this problem can, however, be found.

Rule R1A was initially generated by B, and then B has simply supplied this rule to A. Sometimes this type of learning is called "learning by being told". Agent A was spared the effort of generating the rule.

Rules (RI1-RI4) represent a kind of *interface theory* that establishes relationships between concepts. In our case, these rules provide the definition of some unknown concepts that appeared in the rule supplied by B. These definitions enable agent A to interpret correctly B's theory.

6. Representation of Agent's Beliefs

In all experiments described here the agents' beliefs have been represented using assertions of the form *bel(Ag1, P1)* where *Ag1* represents an agent and *P1* some predicate. Rules of the form

bel(Ag1, P1) :- bel(Ag2, P2)

relate beliefs of one agent to beliefs of another agent. Note that if *Ag1* and *Ag2* are the same, we get a special case of this rule relating different agent's beliefs. Here we will call this representation *meta-level representation.*

For example, A's belief set D1A is represented using the following meta-level assertions:

```
bel("A", father("Paul", "Oscar"))                    (D1A*)
bel("A", father("Oscar", "Louis"))
etc.
```

All other facts are represented in a similar manner. This representation affects the form of the rules generated by GOLEM. The system did not generate rule R1A shown earlier, but rule R1A* which has the following form:

```
bel("A", gfather(X,Y) ) :- bel("A", father(X,Z )) , bel("A", father(Z,Y ))    (R1A*)
```

As this represention is somewhat less readable, we have decided to the simpler form throughout in this paper, without the meta-predicate bel(..). Notice that there is no loss of information as long as we deal with one particular agent only. That is, all rules Ri belonging to the simplified representation system can be automatically transformed into more complex rules Ri* (and vice versa) if we assume that all beliefs belong to one agent only (e.g. A).

As our scenario involves two agents, the issue concerning representation of beliefs deserves more attention. Let us analyse the final theory generated by the system:

```
bel("A", gfather(X,Y))   :- bel("A", father(X,Z)), bel("A",father(Z,Y)).   (R1A*)
bel("A", P)              :- bel("B", P).                                    (RI0*)

bel("B", gfather(X,Y))   :- bel("B", parent(X,Z)),  bel("B", male(X)
                            bel("B", parent(Z,Y)),  bel("B", female(Z)).    (R2B*)

bel("B", parent (X,Z))   :- bel("A", father (X,Z)).                         (RI1*)
bel("B", parent (X,Z))   :- bel("A", mother (X,Z)).                         (RI2*)
bel("B", male (X))       :- bel("A", father (X,_)).                         (RI3*)
bel("B", female (Z))     :- bel("A", mother (Z,_)).                         (RI4*)
```

Rule RI0 is a kind of interface rule that has been added manually to the rules shown above. When interpreted in the Prolog style, we can say that it transforms the goal *bel("A",P)* into *bel("B",P)*. Looking at it from a different angle, this rule enables agent A to interpret goal P using rules of B. Rule R2B* is the rule generated by agent B. The remaining rules (RI1*-RI4*) represent the interface theory. Rule RI1*, for example, enables the system to transform the goal *bel("B", parent (X,Z))* into *bel("A", father (X,Z))*.

7. Discussion

Most earlier work in ML has been concerned with the problem of constructing (and improving) a theory on the basis of examples. The problem of integrating two or more theories has been largely ignored. Perhaps it has reminded people of "learning by being told" which has usually been dismissed as "too easy" and of no real interest to ML community.

The purpose of our earlier work was to partly fill in this gap. The system described in (Brazdil and Torgo, 1990) consisted of several learning agents, and it tried to minimize possible inconsistencies and redundancies on the basis of experimental tests. The experiments have shown that this strategy can lead to performance gains. In this paper we have shown how agents can overcome certain language differences that can arise in communication between them. As has been demonstrated, standard machine learning techniques can be used to acquire the meaning of other agents' concepts. The theory consisting of concept definitions represents a kind of "interface theory". It relates concepts of different agents, providing them with meaning.

Some people argue that although different theories are useful, these should be retained as separate entities. Gams (1989), for example, maintains all apparently redundant rules (theories) within the system. Improvements of performance can be achieved by taking these redundant

rules into account. Theories containing redundant rules are, however, more difficult to understand and consequently also more difficult to modify.

Knowledge integration is concerned with issues that are related to those in *incremental learning* systems. There are some important differences between the two approaches, however. When employing some incremental version of a given learning algorithm only *one* system is constructing theories. Knowledge integration, on the other hand, involves *several* systems all of which try to construct their own theories on the basis of their own experience. Knowledge integration tends to capitalize on the results obtained by different systems.

In our scenario both agent A and B attempt to generate the definition of *grandfather*. As they fail to cover some cases (for example A's definition fails on B's data), agent A attempts to complete his knowledge using the method of knowledge integration. As we have seen this process may involve additional learning (i.e. generation of new concept definitions). If we were not interested to exploit the results obtained by B (that is B's original definitions) we could have simply supplied agent A with B's data. Agent A could have revised his definitions on the basis of the additional data. We believe that agent A would have been able to generate the correct definitions this way (i.e. by incremental learning). However, does not try to capitalize on the results obtained by agent B earlier.

More work could be done to evaluate the two approaches. Obviously communication and generation of the interface theory has certain costs, too. It would be interesting to quantify the effort associated with each alternative. Moreover, it would be useful to define a set of rules (or heuristics) that would enable us to decide which approach would be most appropriate in a new domain.

Acknowledgments

This work was supported by Esprit 2 Project Ecoles (3059). The authors wish to thank Commission of European Communities for their support.

The authors wish to thank also Thomas Hoppe for carrying out some experiments with *ancestors* and *parents* at The Turing Institute in Glasgow and also for various helpful comments. Thanks also to anonymous referees for their helpful remarks. These have taken into account when preparing the final version of this paper.

References

Boose J., Bradshaw J., Kitto C. and Sherma D. (1989): "From ETS to Acquinas: Six Years of Knowledge Acquisition Tool Development," in *Proceedings of Third European Workshop on Knowledge Acquisition for Knowledge-Based Systems*, J. Boose, B. Gaines and J.G. Ganascia (eds.), Paris, July 1989.

Brazdil P. and Torgo L. (1990): "Knowledge Acquisition via Knowledge Integration", in Current Trends in Artificial Intelligence, B. Wielinga et al. (eds.), IOS Press, Amsterdam, 1990.

Durfee E., Lesser V.R. and Corkill D.D. (1989): "Cooperative Distributed Problem Solving", in *The Handbook of Artificial Intelligence, Volume IV,* Barr A., Cohen P.R. and Feigenbaum E.A. (eds.), Addison Wesley, 1989.

Gams M. (1989): "The Measurement Highlight the Importance of Redundant Knowledge", in *Proceedings of 4th European Working Session on Machine Learning (EWSL-89)*, K. Morik (ed.), pp. 71-80, Pitman - Morgan Kaufmann.

Muggleton S. and Feng C. (1990): "Efficient Induction of Logic Programs", in Proceedings of the First Conference on Algorithmic Learning, Tokyo, Japan, October 1990, Ohmsha Publ., Tokyo.

Murray K.S. and Porter B.W. (1989): "Controlling Search for the Consequences of New Information During Knowledge Integration", in *Proceedings of 6th International Workshop on Machine Learning*, A.M. Segre (ed.), Ithaca, New York, Morgan Kaufmann Inc.

Shaw M. and Gaines B. (1989): Knowledge Acquisition: Some Foundations, manual Methods and Future Trends, in *Proceedings of Third European Workshop on Knowledge Acquisition for Knowledge-Based Systems*, J. Boose, B. Gaines and J.G. Ganascia (eds.), Paris, July 1989.