Knowledge Base Refinement as Improving an Incorrect and Incomplete Domain Theory

by

David C. Wilkins

April 1990
The ODYSSEUS program automates knowledge base refinement by improving a domain theory. This paper describes the techniques used by ODYSSEUS to address three types of domain theory pathologies: incorrectness, inconsistency, and incompleteness.

In ODYSSEUS, an incomplete domain theory is extended by the Metarule Chain Completion Method. This method exploits the use of an explicit meta-level representation of the strategy knowledge for a generic problem class (e.g., heuristic classification) that is separate from the domain theory (e.g., medicine) to be improved. Our work implements and compares the extension of an incomplete domain theory using case-based inductive learning and explanation-based apprenticeship learning; in the latter, learning occurs by completing failed explanations of observed human problem-solving actions. Extending an incomplete domain theory and correcting an incorrect domain theory both use the Confirmation Decision Procedure Method, which validates arbitrary instantiated tuples of the knowledge base by the use of an underlying domain theory. Lastly, the consistency of the knowledge base is improved by use of the Sociopathic Reduction Algorithm.
Knowledge Base Refinement As Improving An Incorrect and Incomplete Domain Theory

David C. Wilkins
Knowledge-Based Systems Group
Department of Computer Science
University of Illinois
405 North Mathews Ave
Urbana, IL 61801

April 1990

Submitted to:
Artificial Intelligence Journal

Earlier versions of this paper have appeared as
Knowledge Base Refinement as Improving an Incorrect and Incomplete Domain Theory

David C. Wilkins

Department of Computer Science
University of Illinois
405 North Mathews Ave
Urbana, IL 61801

Abstract

The ODYSSEUS program automates knowledge-base refinement by improving a domain theory. This paper describes the techniques used by ODYSSEUS to address three types of domain theory pathologies: incorrectness, inconsistency, and incompleteness.

In ODYSSEUS, an incomplete domain theory is extended by the metarule chain completion method. This method exploits the use of an explicit metalevel representation of the strategy knowledge for a generic problem class (e.g., heuristic classification) that is separate from the domain theory (e.g., medicine) to be improved. Our work implements and compares the extension of an incomplete domain theory using case-based inductive learning and explanation-based apprenticeship learning; in the latter, learning occurs by completing failed explanations of observed human problem-solving actions. Extending an incomplete domain theory and correcting an incorrect domain theory both use the confirmation decision procedure method, which validates arbitrary instantiated tuples of the knowledge base by the use of an underlying domain theory. Lastly, the consistency of the knowledge base is improved by use of the sociopathic reduction algorithm.
# Contents

1 Introduction 3

2 MINERVA Classification and Design Shell 3

2.1 The Evolution from Heracles to Minerva 5

3 Odysseus's Method for Extending an Incomplete Domain Theory 7

3.1 Detection of Knowledge Base Deficiency 9

3.2 Suggesting a Knowledge Base Repair 10

3.3 Validation of Knowledge Base Repair 12

4 Odysseus's Method for Improving an Incorrect Domain Theory 12

5 Odysseus's Method for Improving an Inconsistent Domain Theory 14

6 Related Research 15

6.1 Odysseus and Explanation-Based Learning 15

6.2 Case-Based versus Apprenticeship Learning 15

7 Experimental Results 16

7.1 Improving an Incorrect and Inconsistent Domain Theory 17

7.2 Extending Incomplete Domain Theory via Case-Based Reasoning 18

7.3 Extending Incomplete Domain Theory via Apprenticeship Learning 18

8 Conclusions 19

9 Acknowledgements 21
1 Introduction

A central problem of expert systems is knowledge-base refinement (Buchanan and Shortliffe, 1984). Numerous research efforts have addressed the problem of improving an expert system that solves heuristic classification problems. The major research projects that have directly confronted this problem include the interactive semi-automatic approaches of TEIRESIAS (Davis, 1982), AQUINAS (Boose, 1984), and MORE (Kahn et al., 1985). They also include the automatic case-based inductive methods of INDUCE (Michalski et al., 1983), ID3 (Quinlan, 1983), SEEK2 (Ginsberg et al., 1985), and RL (Fu and Buchanan, 1985), which perform empirical induction over a library of test cases. This chapter describes a new approach to the refinement problem that involves a combination of failure-driven explanation-based learning and the use of underlying domain theories. Our approach is embodied in the ODYSSEUS learning program; ODYSSEUS contains specific (and separate) methods to address automatically three types of knowledge base pathologies: incorrectness, inconsistency, and incompleteness (Wilkins, 1987).

The remainder of this paper is organized as follows: Section 1.2 describes the MINERVA expert system shell that was specifically designed to facilitate failure-driven explanation-based learning. Our experience has shown that a sophisticated expert system architecture can provide an enormous amount of leverage to a learning program. Section 1.3 describes the apprenticeship learning methods used by ODYSSEUS to extend an incomplete domain theory; the key idea used to extend an incomplete domain theory is called the metarule chain completion method. Section 1.4 describes the methods used by ODYSSEUS to correct an incorrect domain theory; our approach to dealing with an incorrect domain theory is called the confirmation decision procedure method. Section 1.5 discusses the method used to remove certain types of inconsistencies from a correct but inconsistent domain theory; this method is called the sociopathic reduction algorithm. Section 1.6 presents results of a wide range of evaluation experiments that have been carried out, and Section 1.7 describes related research.

2 MINERVA Classification and Design Shell

The ODYSSEUS learning program can improve any knowledge base crafted for the MINERVA expert system shell; its overall organization is shown in Figure 1 (Park et al., 1989). MINERVA is a refinement of HERACLES, based on the experience gained in creating the ODYSSEUS apprenticeship learning program for HERACLES (Wilkins, 1987). HERACLES is itself
a refinement of EMYCIN, based on the experience gained in creating the GUIDON case-based tutoring program for EMYCIN (Clancey, 1986). These shells use a problem-solving method called *heuristic classification*, which is the process of selecting a solution out of a preenumerated solution set, using heuristic techniques (Clancey, 1985). The primary application knowledge base for MINERVA and HERACLES is the NEOMYCIN medical knowledge base for diagnosis of meningitis and similar neurological disorders (Clancey, 1984). This section describes the types of knowledge encoded in MINERVA and HERACLES, and how MINERVA differs from HERACLES.

---

**Figure 1:** MINERVA System Architecture.

*Domain knowledge* consists of MYCIN-like rules and simple frame knowledge for an application domain such as medicine or geology. An example of rule knowledge in Horn clause format is

\[
\text{conclude(migraine-headache, yes, .5) := finding(photophobia, yes),}
\]

meaning 'to conclude the patient has a migraine headache with a certainty .5, determine if the patient has photophobia.' An example of frame knowledge is

\[
\text{subsumed-by(viral-meningitis, meningitis),}
\]

meaning 'hypothesis viral meningitis is subsumed by the hypothesis meningitis.'
Problem-state knowledge is generated during execution of the expert system. Examples of problem-state knowledge are rule-applied(rule163), which says that rule 163 has been applied during this consultation, and differential(migraine-headache, tension-headache), which says that the expert system's active hypotheses are migraine headache and tension headache.

Strategy knowledge is contained in the shell, and it approximates a cognitive model of problem solving. For heuristic classification problems, this model is often referred to as hypothesis-directed reasoning (Elstein et al., 1978). The different problem-solving strategies that can be employed during problem solving are explicitly represented, which facilitates use of the model to follow the line of reasoning of a human problem solver. The strategy knowledge determines what domain knowledge is relevant at any given time, and what additional information is needed to solve the problem. The problem-state and domain knowledge, including rules, are represented as tuples; and strategy metarules are quantified over these tuples.

The strategy knowledge needs to access the domain and problem-state knowledge. To achieve this, the domain and problem-state knowledge is represented as tuples. Even rules are translated into tuples. For example, if rule 160 is conclude(hemorrhage yes .5) :- finding(diplopia yes) ∧ finding(aphasia yes), it would be translated into the following four tuples: evidence.for(diplopia hemorrhage rule160 .5), evidence.for(aphasia hemorrhage rule160 .5), antecedent(diplopia rule160), antecedent(aphasia, rule160). Strategy metarules are quantified over the tuples. Figure 4 presents four strategy metarules in Horn clause form; the tuples in the body of the clause quantify over the domain and problem-state knowledge. The rightmost metarule in Figure 4 encodes the strategy to find out about a symptom by finding out about a symptom that subsumes it. The metarule applies when the goal is to find out symptom P1, and there is a symptom P2 that is subsumed by P1, and P2 takes Boolean values, and it is currently unknown, and P2 should be asked about instead of being derived from first principles. This is one of eight strategies in HERACLES that is also used in MINERVA for finding out the value of a symptom; this particular strategy of asking a more general question has the advantage of cognitive economy: a 'no' answer provides the answer to a potentially large number of questions, including the subsumed question.

2.1 The Evolution from Heracles to Minerva

MINERVA is a reworking of HERACLES, similar to the way that HERACLES is a reworking of EMYCIN. The ultimate objective in both these efforts has been a more declarative and modular representation of knowledge. This facilitates construction of a learning program to
examine and reason about the knowledge structures of the metalevel strategy in the expert system, to interpret better a user's strategy in terms of the metalevel strategy knowledge in the expert system, and to allow the same shell to encode strategy knowledge for the generic problem tasks of analysis (e.g., heuristic classification) and synthesis (e.g., VLSI circuit design).

There are four principal differences between MINERVA and HERACLES at the strategy level. In determining which task to perform next, HERACLES uses a fixed order goal tree; by contrast MINERVA employs an opportunistic blackboard scheduler. This facilitates interpreting a user's strategy in terms of the expert system's strategies, and better integrates top-down and bottom-up strategic reasoning. Second, in controlling metalevel reasoning, HERACLES uses dynamic control flags and variables, such as task end conditions. In MINERVA, a pure functional programming style and a deliberation-action loop have been used; this eliminates all flags and variables at the strategy level. So in MINERVA, the system state is completely determined by domain-level static and dynamic knowledge. Third, in HERACLES, strategy metarule premises sometimes change the state of the system, invoke subgoals, and use procedural attachment to LISP code; and HERACLES strategy metarule actions can invoke several goals. In contrast, MINERVA metarules do not follow any of these practices, which allows a pure deliberation-action cycle for strategic reasoning. The MINERVA style of metarules reduces side effects, thus making it easier for the learning program to reason about the strategy knowledge. Fourth, in MINERVA, more of the meta-level code in the expert system, such as the rule interpreter code, has been encoded in strategy metarules.

Other changes are as follows: The MINERVA system is completely implemented in PROLOG; by contrast, HERACLES uses a combination of PROLOG-like clauses with procedural attachment to LISP for each of the PROLOG clause predicates in metarules. The more uniform representation in MINERVA moves us toward our long-term goal of allowing a learning program to reason about all knowledge structures in the expert system shell. MINERVA incorporates an ATMS to maintain consistency of the knowledge base, uses a logic metainterpreter, and supports both certainty factors and Pearl's method to represent rule certainty and for propagation of information in a hierarchy of diagnostic hypotheses (Pearl, 1986b; Pearl, 1986a). As can be seen, all of the changes that mentioned have resulted in a more declarative and functional knowledge representation.
3 Odysseus’s Method for Extending an Incomplete Domain Theory

We have developed two methods for extending an incomplete domain theory: an apprentice-ship learning approach and a case-based reasoning approach. This section will only describe the former approach. Table 1 shows the major refinement steps and the method of achieving them for apprenticeship and case-based learning. The techniques will be elaborated below.

The solution approach of the ODYSSEUS apprenticeship program for extending an incomplete domain theory in a learning-by-watching scenario is illustrated in Figure 2. As Figure 2 shows, the learning process involves three distinct steps: detect domain theory deficiency, suggest domain theory repair, and validate domain theory repair. This section defines the concept of an explanation and then describes the three learning steps.

The main observable problem-solving activity in a diagnostic session is finding out values of features of the artifact to be diagnosed—we refer to this activity as asking findout questions. An explanation in ODYSSEUS is a proof that demonstrates how an expert’s find-out question is a logical consequence of the current problem state, the domain and strategy knowledge, and one of the current high-level strategy goals. An explanation is created by backchaining the metalevel strategy metarules; Figure 4 provides examples of these metarules represented in Horn clause form. The backchaining starts with the findout metarule and continues until a metarule is reached whose head represents a high-level problem-solving

<table>
<thead>
<tr>
<th>Learning Method</th>
<th>Case-Based Learning (similarity-based)</th>
<th>Apprenticeship Learning (explanation-based)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>Heuristic rules.</td>
<td>Heuristic rules. 4 types of frame knowledge.</td>
</tr>
<tr>
<td>Detect KB deficiency</td>
<td>Select and run a case. Deficiency exists if case is misdiagnosed.</td>
<td>Observe expert solving a case. Deficiency exists if action of expert cannot be explained.</td>
</tr>
<tr>
<td>Suggest KB repair</td>
<td>Generalize or specialize rules. Induce new rules.</td>
<td>Find tuples that allow explanations to be completed under single fault assumption.</td>
</tr>
<tr>
<td>Validate KB repair</td>
<td>Use underlying domain theory to validate repairs.</td>
<td>Use underlying domain theory to validate repairs.</td>
</tr>
</tbody>
</table>

Table 1: Comparison of case-based and apprenticeship learning method for extending an incomplete domain theory.
goal. To backchain a metarule requires unification of the body of the Horn clause with domain and problem-state knowledge. Examples of high-level goals are: to test a hypothesis, to differentiate between several plausible hypotheses, to ask a clarifying question, and to ask a general question.

Apprenticeship learning is a form of learning by watching, in which learning occurs as a by-product of building explanations of human problem-solving actions. An apprenticeship is the most powerful method that human experts use to refine and debug their expertise in knowledge-intensive domains such as medicine. The major accomplishment of our method of apprenticeship learning is a demonstration of how an explicit representation of the strategy
knowledge for a general problem class, such as diagnosis, can provide a basis for learning the knowledge that is specific to a particular domain, such as medicine.

3.1 Detection of Knowledge Base Deficiency

The first stage of learning involves the detection of a knowledge base deficiency. An expert's problem solving is observed and explanations are constructed for each of the observed problem-solving actions. An example will be used to illustrate our description of the three stages of learning, based on the NEOMYCIN knowledge base for diagnosing neurology problems. The input to ODYSSEUS is the problem-solving behavior of a physician, John Sotos, as shown in Figure 3. In our terminology, Dr. Sotos asks findout questions and concludes with a final diagnosis. For each of his actions, ODYSSEUS generates one or more explanations of his behavior.

When ODYSSEUS observes the expert asking a findout question, such as asking if the patient has visual problems, it finds all explanations for this action. When none can be

<table>
<thead>
<tr>
<th>Patient's Complaint and Volunteered Information:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Alice Ecila, a 41 year old black female.</td>
</tr>
<tr>
<td>2. Chief complaint is a headache.</td>
</tr>
<tr>
<td>Physician's Data Requests:</td>
</tr>
<tr>
<td>3. Headache duration?</td>
</tr>
<tr>
<td>focus=tension headache. 7 days.</td>
</tr>
<tr>
<td>4. Headache episodic?</td>
</tr>
<tr>
<td>focus=tension headache. No.</td>
</tr>
<tr>
<td>5. Headache severity?</td>
</tr>
<tr>
<td>focus=tension headache. 4 on 0-4 scale.</td>
</tr>
<tr>
<td>6. Visual problems?</td>
</tr>
<tr>
<td>focus=subarachnoid hemorrhage. Yes.</td>
</tr>
<tr>
<td>7. Double vision?</td>
</tr>
<tr>
<td>focus=subarachnoid hemorrhage, tumor. Yes.</td>
</tr>
<tr>
<td>8. Temperature?</td>
</tr>
<tr>
<td>focus=infectious process. 98.7 Fahrenheit.</td>
</tr>
</tbody>
</table>

... ...

<table>
<thead>
<tr>
<th>Physician's Final Diagnosis:</th>
</tr>
</thead>
</table>

Figure 3: An example of what the Odysseus apprentice learner sees. The data requests in this problem-solving protocol were made by John Sotos, M.D. The physician also provides information on the focus of the data requests. The answers to the data requests were obtained from an actual patient file from the Stanford University Hospital, extracted by Edward Herskovits, M.D.
found, an explanation failure occurs. This failure suggests that there is a difference between the knowledge of the expert and the expert system and it provides a learning opportunity. The knowledge difference may lie in any of the three types of knowledge that we have described: strategy knowledge, domain knowledge, or problem state knowledge. Currently, ODYSSEUS assumes that the cause of the explanation failure is that the domain knowledge is deficient. In the current example, no explanation can be found for findout question number 7 in figure 3 (asking about visual problems), and an explanation failure occurs.

3.2 Suggesting a Knowledge Base Repair

The second step of apprenticeship learning is to conjecture a knowledge base repair. A confirmation theory (which will be described in the discussion of the third stage of learning) can judge whether an arbitrary tuple of domain knowledge is erroneous, independent of the other knowledge in the knowledge base.

The search for the missing knowledge begins with the single fault assumption. It should be noted that the missing knowledge is described conceptually as a single fault, but because of the way the knowledge is encoded, we can learn more than one tuple when we

<table>
<thead>
<tr>
<th>Group Hypotheses</th>
<th>Test Hypothesis</th>
<th>Applyrule</th>
<th>Findout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy Metarule</td>
<td>Strategy Metarule</td>
<td>Strategy Metarule</td>
<td>Strategy Metarule</td>
</tr>
<tr>
<td>goal(group-hyp(H1,H2)) :- differential(H1), taxonomic(H1), parent(H2,H1), not pursued(H2), closest-common-ancestor(H2,H1), not(root(H2)), goal(test-hyp(H2)).</td>
<td>goal(test-hyp(H2)) :- concluded-by(H1,R1), not(pursued(R1)), inpremise(P1 R1), goal(applyrule(R1)).</td>
<td>goal(applyrule(R1)) :- not(rule-applied(R1)), inpremise(P1 R1), evid-for(P1,H2,R1 S1), soft-datum(P1), not(concluded(P1)), goal(findout(P1)), applyrule-forward(R1).</td>
<td>goal(findout(P1)) :- subsumes(P2,P1), not(concluded(P1)), boolean(P2), not(concluded(P2)), ask-user(P1).</td>
</tr>
</tbody>
</table>

Figure 4: Learning by completing failed explanations. The illustrated strategy-level Horn clause metarules can chain together to form an explanation of how the the findout action of ask-user(P1) relates to the high-level goal of group-hypoth(H1,H2). In this particular case, all the tuples in the chain cannot be instantiated with domain knowledge. Odysseus' attempts to complete this and other failed explanation chains by adding domain knowledge to the knowledge base so that all the tuples unify.
learn rule knowledge. For ease of presentation, this feature is not shown in the following examples.

Conceptually, the missing knowledge could be eventually identified by adding a random domain knowledge tuple to the knowledge base and seeing whether an explanation of the expert's findout request can be constructed. How can a promising piece of such knowledge be effectively found? Our approach is to apply backward chaining to the findout question metarule, trying to construct a proof that explains why it was asked. When the proof fails, it is because a tuple of domain or problem-state knowledge needed for the proof is not in the knowledge base. If the proof fails because of problem-state knowledge, we look for a different proof of the findout question. If the proof fails because of a missing piece of domain knowledge, we temporarily add this tuple to the domain knowledge base. If the proof then goes through, the temporary piece of knowledge is our conjecture of how to refine the knowledge base.

Figure 4 illustrates one member of the set of failed explanations that ODYSSEUS examines in connection with the unexplained action, ask-user(visual problems), that is contained in the tail of the rightmost metarule. These strategy metarules create a chain between the high-level goal in the head of the leftmost metarule, group-hypotheses(Hypothesis1, Hypothesis2) and the low-level observable action in the tail of the rightmost metarule ask-user(visual problems). Note that this chain is but one path is a large explanation graph that connects the observable action of asking about visual problems to all high-level goals. Each path in the graph is a potential explanation, and each node in a path is a strategy metarule. The failed explanation that ODYSSEUS is examining consists of the four metarules shown in Figure 4: Group Hypotheses, Test Hypothesis, Applyrule, and Findout. For a metarule to be used in a proof, its variables must be instantiated with domain or problem state tuples that are present in the knowledge base. In this example, the evidence.for tuple is responsible for the highlighted chain not forming a proof. It forms an acceptable proof if the tuple evidence.for(photophobia acute.meningitis $rule $cf) or evidence.for(diplopia acute.meningitis $rule $scf) is added to the knowledge base. During the step that generates repairs, neither the form of the left-hand side of the rule (e.g., number of conjuncts) or the strength is known. In the step to evaluate repairs, the exact form of the rule is produced in the process of evaluation of the worth of the tuple.
3.3 Validation of Knowledge Base Repair

The task of the third step of apprenticeship learning is to evaluate the proposed repair. To do this, we use the confirmation decision procedure (CDP) method. CDPs are constructed for each type of tuple in the domain theory, and can determine if the tuple is an acceptable tuple. Of the 19 different types of tuples in the Neomycin knowledge base, we have implemented CDPs for three of them: evidence.for, clarifying.question, and ask.general.question tuples. In addition to their use for validating knowledge base repairs, CDPs are also used to modify or delete incorrect parts of the initial domain theory; they are described in greater detail in Section 4.

Evidence.for tuples were generated in the visual problems example. In order to confirm the first candidate tuple, ODYSSEUS uses an empirical induction system that generates and evaluates rules that have photophobia in their premise and acute meningitis in their conclusion. A rule is found that passes the rule 'goodness' measures, and it is automatically added to the object-level knowledge base. All the tuples that are associated with the rule are also added to the knowledge base. This completes our example.

The CDP method also validates frame-like knowledge. An example of how this is accomplished will be described for clarify question tuples, such as clarify.questions(headache-duration headache). This tuple means that if the physician discovers that the patient has a headache, she should always ask how long the headache has lasted. The confirmation theory must determine whether headache duration is a good clarifying question for the 'headache' symptom. To achieve this, ODYSSEUS first checks to see if the question to be clarified is related to many hypotheses (the ODYSSEUS explanation generator allows it to determine this), and then tests whether the clarifying question can potentially eliminate a high percentage of these hypotheses. If these two criteria are met, then the clarify questions tuple is accepted.

4 Odysseus's Method for Improving an Incorrect Domain Theory

The main focus of this chapter is on extending an incomplete domain theory via apprenticeship learning. However, it is clearly helpful if we are extending a domain theory that is correct and consistent. This section describes the methods that we have developed to improve the correctness of the domain theory. These methods are applied to the domain
theory prior to the use of apprenticeship learning.

The key to addressing the problem of incorrect knowledge is the use of the confirmation decision procedure (CDP) method, which connects tuples in the domain theory to underlying theories of the domain that are capable of judging their correctness. In this approach, a CDP is created for each type of domain theory tuple in the knowledge base. Given an arbitrary instantiated tuple, the CDP calculates whether the tuple is true or false. In some cases the CDP can suggest how the tuple can be modified so as to make it true.

Of the 19 different types of domain theory tuples in the NEOMYCIN domain theory, we have created CDP's for three types of tuples. Theses tuples comprise approximately 70% of all tuples in the domain theory. For example, a CDP has been implemented for evidence.for tuples. These tuples are derived from the heuristic domain rules provided by a user that relate evidence to hypotheses. Validating evidence.for tuples therefore consists of validating the heuristic associational rules in the knowledge base.

The CDP for evidence.for consists of an induction system, a set of rule biases, and a representative case library for the application domain. It accepts or rejects heuristic rules, whether they are rules in the initial knowledge base or rules conjectured during apprenticeship learning. In addition to accepting or rejecting rules, the CDP for evidence.for can modify a given rule to make it correct; it does this by adding conjuncts or modifying the rule strength. A rule can be modified to be “correct” by using probability and decision theory and representative sets of cases to to determine its correct weight or strength (in contrast to trusting the weight provided by the user). If a rule lacks sufficient strength, the CDP will try to add conjuncts to the rule to increase its specificity.

When given an evidence.for tuple, its corresponding heuristic associational rule, which is indicated by the third argument of the evidence.for relation, is tested in five ways by the evidence.for CDP. A test for rule simplicity ensures that the number of antecedent conditions of the rules are less than the specified number. The test for strength accepts rules whose certainty factors (CF) are greater than a threshold value. The third bias is to test the generality of the rules. It succeeds only if the rules cover a certain percentage of the cases in a representative case library. The test for colinearity ensures that the proposed rules are not similar to any existing rules in performing classification of the induction set of cases. Finally the bias for uniqueness will check that the rules fire on a training case and there exist no rules in the current domain rule set that also succeed for that case. Good rules are those recommendations that pass the verification process. This rule may then be added into the system.
It is often difficult to create CDPs for some types of tuples in the domain theory. For example, consider the tuple type askfirst(Parm). This tuple says that a particular feature of the system being diagnosed should be obtained from a user instead of derived from first-principles. It is difficult to imagine how to do this for an arbitrary feature, although eventually a way must be found if knowledge acquisition is to be completely automated.

Note that most knowledge bases are much more heterogeneous than LEAP, a learning apprentice for acquiring a domain theory that consists of VLSI circuit implementation rules. In this system, the domain theory only contains implementation rules (in our parlance, only contains one type of domain tuple). LEAP can verify the implementation rules using Kirchhoff’s laws as its underlying domain theory. The challenge of using this idea for knowledge-base systems is that most domain theories contain many different types of domain knowledge, not just one type as in LEAP.

The CDPs were originally constructed to validate repairs during apprenticeship learning. However, they nicely allow the initial knowledge base to be validated prior to apprenticeship learning. As will be reported in Section 1.6, about half of the existing knowledge base is modified during the processing stage that focuses on ensuring that the domain theory contains correct knowledge.

5 Odysseus’s Method for Improving an Inconsistent Domain Theory

A processing stage prior to apprenticeship learning also removes a form of inconsistent knowledge from the domain theory, which is responsible for deterioration of the performance of the system due to sociopathic interactions between elements of the domain theory. A domain theory is sociopathic if and only if (1) all the rules in the knowledge base individually meet some “goodness” criteria; and (2) a subset of the knowledge base gives better performance than the original knowledge base. The five biases described in Section 1.4 provide an example of goodness criteria for heuristic rules in the domain theory.

The significance of the phenomena of sociopathy is as follows. First, most extant expert systems have sociopathic knowledge bases. Second, traditional methods to correct missing and wrong rules, e.g., the general TEIRESIAS approach (Davis, 1982), cannot handle the problem. Third, sociopathy imposes a limit on the quality of knowledge base performance. And last, it implies that some kind of global refinement for the acquired knowledge
is essential for machine learning systems.

The phenomena of sociopathicity is addressed at length in another paper, wherein we show that the best method for dealing with this form of inconsistency is to find a subset of the original domain theory that is not sociopathic (which must exist by our definition of sociopathicity). A summary of our results are as follows: The process of finding an optimal subset of a sociopathic knowledge base is modeled as a bipartite graph minimization problem and shown to be NP-hard. A heuristic method, the sociopathic reduction algorithm, has been developed to find a suboptimal solution for sociopathic domain theories. The heuristic method has been experimentally shown to give good results.

6 Related Research

6.1 Odysseus and Explanation-Based Learning

The ODYSSEUS apprenticeship learning method involves the construction of explanations, but it is different from explanation-based learning as formulated in EBG (Mitchell et al., 1986) and EBL (DeJong, 1986); it is also different from explanation-based learning in LEAP (Mitchell et al., 1989), even though LEAP also focuses on the problem of improving a knowledge-based expert system. In EBG, EBL, and LEAP, the domain theory is capable of explaining a training instance, and learning occurs by generalizing an explanation of the training instance. In contrast, in our apprenticeship research, a learning opportunity occurs when the domain theory, which is the domain knowledge base, is incapable of producing an explanation of a training instance. The domain theory is incomplete or erroneous, and all learning occurs by making an improvement to this domain theory.

6.2 Case-Based versus Apprenticeship Learning

In empirical induction from cases, a training instance consists of an unordered set of feature-value pairs for an entire diagnostic session and the correct diagnosis. In contrast, a training instance in apprenticeship learning is a single feature-value pair given within the context of a problem-solving session. This training instance is therefore more fine-grained, can exploit the information implicit in the order in which the diagnostician collects information, and allows obtaining many training instances from a single diagnostic session. Our apprenticeship learning program attempts to construct an explanation of each training instance; an
explanation failure occurs if none is found. The apprenticeship program then conjectures and tests modifications to the knowledge base that allow an explanation to be constructed. If an acceptable modification is found, the knowledge base is altered accordingly. This is a form of learning by completing failed explanations.

The case-based learning approach currently modifies or adds heuristic rules to the knowledge base. It runs all the cases in the library and locates those that are misdiagnosed. Given a misdiagnosed case, the local credit assignment problem is solved as follows: The premises of the rules that concluded the wrong final diagnosis are weakened by specialization, and the premises of the rules that concluded the correct diagnosis are strengthened. If this does not solve the problem, new rules will be induced from the patient case library that apply to the misdiagnosed case and that conclude the correct final diagnosis. The verification procedure used to test all knowledge base modifications is identical to that described for apprenticeship learning.

7 Experimental Results

Our knowledge-acquisition experiments centered on improving the ProHCD shell containing the NEOMYCIN knowledge base for diagnosing neurology problems. The initial NEOMYCIN knowledge base was constructed manually over a 7 year period; the first test of this system on a representative suite of test cases was performed in conjunction with the ODYSSEUS system. The NEOMYCIN vocabulary includes 60 diseases; our physician, Dr. John Sotos, determined that the existing data request vocabulary of 350 manifestations only allowed diagnosis of 10 of these diseases. Another physician, Dr. Edward Herskovits, constructed a case library of 115 cases for these 10 diseases from actual patient cases from the Stanford Medical Hospital, to be used for testing ODYSSEUS. The test set consisted of 112 of these cases.

Let us begin our performance analysis by considering the baseline system performance prior to any ODYSSEUS knowledge base refinement. The expected diagnostic performance that would be obtained by randomly guessing diagnoses is 10%, and the performance expected by always choosing the most common disease is 18%. Version 2.3 of HERACLES with the NEOMYCIN knowledge base initially diagnosed 31% of the cases correctly, which is 3.44 standard deviations better than always selecting the disease that is a priori the most likely. On a student t-test, this is significant at a $t = .001$ level of significance. Thus we can conclude that NEOMYCIN's initial diagnostic performance is significantly better than guessing. Version 3.1 of ProHCD, with the manually constructed NEOMYCIN knowledge
base, gave almost identical performance results; it initially diagnosed 32 of the 112 cases correctly (28.5% accuracy).

Table 1.2 shows the various diseases and their sample sizes in the evaluation set. The results of each test suite are described along three dimensions. TP (true-positive) refers to the number of cases that the expert system correctly diagnosed as present, FN (false-negative) to the number of times a disease was not diagnosed as present but was indeed present, and FP (false-positive) to the number of times a disease was incorrectly diagnosed as present.

<table>
<thead>
<tr>
<th>Disease</th>
<th>Number Cases</th>
<th>KB1</th>
<th>KB2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bacterial Meningitis</td>
<td>16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Brain Abscess</td>
<td>7</td>
<td>14</td>
<td>2</td>
</tr>
<tr>
<td>Cluster Headache</td>
<td>10</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>Fungal Meningitis</td>
<td>8</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>Migraine</td>
<td>10</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Myco-TB Meningitis</td>
<td>4</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Primary Brain Tumor</td>
<td>16</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>Subarach Hemorrhage</td>
<td>21</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>Tension Headache</td>
<td>9</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>Viral Meningitis</td>
<td>11</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>None</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Totals</td>
<td>112</td>
<td>32</td>
<td>80</td>
</tr>
</tbody>
</table>

Table 2: Summary of MINERVA experiments. The KB1 column is the performance using the manually constructed domain theory. KB2 shows performance after use of methods that correct an incorrect domain theory.

7.1 Improving an Incorrect and Inconsistent Domain Theory

The first stage of improvement involves locating and modifying incorrect domain knowledge tuples. Our method modified 48% of the heuristic rules in the knowledge base. The improvement obtained using the refined knowledge base is shown in column KB2 of Table 1.2; ProHCD diagnosed 62 cases correctly (55.3% accuracy), showing an improvement of about 27%. The second stage of improvement involves correcting inconsistent domain knowledge. No experimental results are reported here, although our methods have been previously shown to lead to significant improvement (Wilkins and Ma, 1989).
Table 3: Summary of MINERVA experiments. KB3 and KB4 show the performance after using case-based learning and apprenticeship learning, respectively, to extend the incomplete domain theory.

7.2 Extending Incomplete Domain Theory via Case-Based Reasoning

The third stage of improvement involves extending a correct but incomplete domain knowledge base. Two experiments were conducted. The first used case-based learning; all the cases were run, and two misdiagnosed cases in areas where the knowledge base was weak were selected. The case-based learning approach was applied to these two cases. This refinement, shown in column KB3 of Table 1.2, enabled the system to diagnose 68 cases correctly (60.7% accuracy), showing an aggregate improvement of 32%.

7.3 Extending Incomplete Domain Theory via Apprenticeship Learning

The second experiment used apprenticeship learning. For use as a training set, problem-solving protocols were collected by having Dr. Sotos solve two cases, consisting of approximately 30 questions each. ODYSSEUS discovered 10 pieces of knowledge by watching these two cases being solved; eight of these were domain rule knowledge. These eight pieces of information were added to the NEOMYCYN knowledge base of 152 rules, along with two pieces of frame knowledge that classified two symptoms as ‘general questions’; these are questions that should be asked of every patient. This refinement, shown in column KB4
of Table 1.2, enabled the system to diagnose 73 cases correctly (65.2% accuracy), an aggregate improvement of about 37%. Compared to NEOMYCIN's original performance, the performance of NEOMYCIN after improvement by ODYSSEUS is 2.86 standard deviations better. On a student t-test, this is significant for \( t = .006 \). One would expect the improved NEOMYCIN to perform better than the original NEOMYCIN in better than 99 out of 100 sample sets.

It is important to note that the improvement occurred despite the physician's only diagnosing one of the two cases correctly. The physician correctly diagnosed a cluster headache case and misdiagnosed a bacterial meningitis case. As is evident from examining Tables 1.1 and 1.2, the improvement was over a wide range of cases, and the accuracy of diagnosing bacterial meningitis cases actually decreased. These counterintuitive results confirm our hypothesis that the power of our learning method derives from following the line of reasoning of a physician on individual findout questions and is not sensitive to the final diagnosis as is the case in learning by empirical induction from examples.

All of this new knowledge learned by apprenticeship learning was judged by Dr. Sotos as plausible medical knowledge, except for a domain rule linking aphasia to brain abscess. Importantly, the new knowledge was judged by our physician to be of much higher quality than when straight empirical induction was used to expand the knowledge base, without the use of explanation-based learning.

More experimental work remains. Our previous experiments with ODYSSEUS suggest that the apprenticeship learning approach is better than a case-based approach for producing a user-independent knowledge base to support multiple problem-solving goals such as learning, teaching, problem-solving, and explanation generation.

8 Conclusions

In this chapter, we presented the three distinct methods used by ODYSSEUS to improve a domain theory.

Our method of extending an incomplete domain theory is a form of failure-driven explanation-based learning, which we refer to as apprenticeship learning. Apprenticeship is the most effective means for human problem solvers to learn domain-specific problem-solving knowledge in knowledge-intensive domains. This observation provides motivation to give apprenticeship learning abilities to knowledge-based expert systems. The paradigmatic
example of an apprenticeship period is medical training, in which we have performed our investigations.

With respect to the incomplete theory problem, the research described illustrates how an explicit representation of the strategy knowledge for a general problem class, such as diagnosis, provides a basis for learning the domain-level knowledge that is specific to a particular domain, such as medicine, in an apprenticeship setting. Our approach uses a given body of strategy knowledge that is assumed to be complete and correct with the goal of learning domain-specific knowledge. This contrasts with learning programs such as LEX and LP where the domain-specific knowledge (e.g., integration formulas) is completely given at the start, and the goal of learning strategy knowledge (e.g., preconditions of operators) (Mitchell et al., 1983). Two sources of power of the ODYSSEUS approach are the method of completing failed explanations, called the metarule chain completion method, and the use of a underlying domain theories to evaluate domain-knowledge changes via the confirmation decision procedure method. Our approach complements the traditional method of empirical induction from examples for refining a knowledge base for an expert system for heuristic classification problems. With respect to learning certain types of heuristic rule knowledge, empirical induction from examples plays a significant role in our work. In these cases, an apprenticeship approach can be viewed as a new method of biasing selection of which knowledge is learned by empirical induction.

An apprenticeship learning approach, such as described in this chapter, is perhaps the best possible bias for automatic creation of a large 'user-independent' knowledge bases for expert systems. We desire to create knowledge bases that will support the multifaceted dimensions of expertise exhibited by some human experts, dimensions such as diagnosis, design, teaching, learning, explanation, and critiquing the behavior of another expert.

The long-term objectives of this research are the creation of learning methods that can harness an explicit representation of generic shell knowledge and that can lead to the creation of a user-independent knowledge base that rests on deep underlying domain models. Within this framework, this paper described specialized methods that address three major types of knowledge base pathologies: incorrect, inconsistent, and incomplete domain knowledge.
9 Acknowledgements

Many people have greatly contributed to the evolution of the ideas presented in this chapter. We would especially like to thank Bruce Buchanan, Bill Clancey, Tom Dietterich, Haym Hirsh, John Holland, John Laird, Pat Langley, Bob Lindsay, John McDermott, Ryszard Michalski, Roy Rada, Tom Mitchell, Paul Rosenbloom, Ted Shortliffe, Paul Scott, Devika Subramanian, Marianne Winslett, the members of the Grail learning group, and the Guidon tutoring group. This work would not have been possible without the help of physicians Eddy Herskovits, Kurt Kapsner, and John Sotos.

We would also like to express our deep gratitude to Lawrence Chachere, Ziad Najem, Young-Tack Park, and Kok-Wah Tan, and other members of the Knowledge-based Systems group at University of Illinois for their major role in the design and implementation of the MINERVA shell and for many fruitful discussions. This work was principally supported by NSF grant MCS-83-12148, and ONR grants N00014-79C-0302 and N00014-88K0124.
References


<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
<th>City</th>
<th>State</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dr. Charles Perzy</td>
<td>Department of Computer Science</td>
<td>Pittsburgh, PA</td>
<td>15213</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Mark Pex</td>
<td>Carnegie Mellon University</td>
<td>Pittsburgh, PA</td>
<td>15213</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Carl R. Pedersen</td>
<td>Department of Educational Psychology</td>
<td>Calgary, Alberta</td>
<td>60628</td>
<td>Canada</td>
</tr>
<tr>
<td>Dr. Art Greenser</td>
<td>Dept. of Psychology</td>
<td>Memphis, TN</td>
<td>38115</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Wayne Gray</td>
<td>Artificial Intelligence Laboratory</td>
<td>White Plains, NY</td>
<td>20904</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Dr. John R. Pedersen</td>
<td>BBN Laboratories</td>
<td>Cambridge, MA</td>
<td>02138</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Allen Ginsberg</td>
<td>AMT Bell Laboratories</td>
<td>Holmdel, NJ</td>
<td>08731</td>
<td>USA</td>
</tr>
<tr>
<td>Mr. Lee Gladwin</td>
<td>AT&amp;T Bell Laboratories</td>
<td>Leesburg, VA</td>
<td>20176</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Robert Glass</td>
<td>Learning Research &amp; Development Center</td>
<td>University of Pittsburgh</td>
<td>15260</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Marva D. Clark</td>
<td>111 Homestead Terrace</td>
<td>Ithaca, NY</td>
<td>14850</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Dr. Michael Friendly</td>
<td>Psychology Department</td>
<td>York University</td>
<td></td>
<td>South Africa</td>
</tr>
<tr>
<td>Dr. Dr. Michael Friendly</td>
<td>York University</td>
<td>Toronto ON</td>
<td></td>
<td>Canada</td>
</tr>
<tr>
<td>Col. Dr. Ernst Frese</td>
<td>Herzophysiologischer Dienst</td>
<td>1126 Wien</td>
<td></td>
<td>Austria</td>
</tr>
<tr>
<td>Dr. Dr. Robert M. Gagne</td>
<td>Psychology Department</td>
<td>Tallahassee, FL</td>
<td>32302</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Brian R. Gaines</td>
<td>Knowledge Sciences Institute</td>
<td>Calgary, Alberta</td>
<td></td>
<td>Canada</td>
</tr>
<tr>
<td>Dr. Dr. Mark Pex</td>
<td>Carnegie Mellon University</td>
<td>Pittsburgh, PA</td>
<td>15213</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Dr. Karl R. Pedersen</td>
<td>Department of Educational Psychology</td>
<td>Calgary, Alberta</td>
<td>60628</td>
<td>Canada</td>
</tr>
<tr>
<td>Dr. Dr. Art Greenser</td>
<td>Dept. of Psychology</td>
<td>Memphis, TN</td>
<td>38115</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Wayne Gray</td>
<td>Artificial Intelligence Laboratory</td>
<td>White Plains, NY</td>
<td>20904</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Dr. John R. Pedersen</td>
<td>BBN Laboratories</td>
<td>Cambridge, MA</td>
<td>02138</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Allen Ginsberg</td>
<td>AMT Bell Laboratories</td>
<td>Holmdel, NJ</td>
<td>08731</td>
<td>USA</td>
</tr>
<tr>
<td>Mr. Lee Gladwin</td>
<td>AT&amp;T Bell Laboratories</td>
<td>Leesburg, VA</td>
<td>20176</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Robert Glass</td>
<td>Learning Research &amp; Development Center</td>
<td>University of Pittsburgh</td>
<td>15260</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Marva D. Clark</td>
<td>111 Homestead Terrace</td>
<td>Ithaca, NY</td>
<td>14850</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Dr. Michael Friendly</td>
<td>Psychology Department</td>
<td>York University</td>
<td></td>
<td>South Africa</td>
</tr>
<tr>
<td>Dr. Dr. Michael Friendly</td>
<td>York University</td>
<td>Toronto ON</td>
<td></td>
<td>Canada</td>
</tr>
<tr>
<td>Col. Dr. Ernst Frese</td>
<td>Herzophysiologischer Dienst</td>
<td>1126 Wien</td>
<td></td>
<td>Austria</td>
</tr>
<tr>
<td>Dr. Dr. Robert M. Gagne</td>
<td>Psychology Department</td>
<td>Tallahassee, FL</td>
<td>32302</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Brian R. Gaines</td>
<td>Knowledge Sciences Institute</td>
<td>Calgary, Alberta</td>
<td></td>
<td>Canada</td>
</tr>
<tr>
<td>Dr. Dr. Karl R. Pedersen</td>
<td>Department of Educational Psychology</td>
<td>Calgary, Alberta</td>
<td>60628</td>
<td>Canada</td>
</tr>
<tr>
<td>Dr. Dr. Art Greenser</td>
<td>Dept. of Psychology</td>
<td>Memphis, TN</td>
<td>38115</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Wayne Gray</td>
<td>Artificial Intelligence Laboratory</td>
<td>White Plains, NY</td>
<td>20904</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Dr. John R. Pedersen</td>
<td>BBN Laboratories</td>
<td>Cambridge, MA</td>
<td>02138</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Allen Ginsberg</td>
<td>AMT Bell Laboratories</td>
<td>Holmdel, NJ</td>
<td>08731</td>
<td>USA</td>
</tr>
<tr>
<td>Mr. Lee Gladwin</td>
<td>AT&amp;T Bell Laboratories</td>
<td>Leesburg, VA</td>
<td>20176</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Robert Glass</td>
<td>Learning Research &amp; Development Center</td>
<td>University of Pittsburgh</td>
<td>15260</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Marva D. Clark</td>
<td>111 Homestead Terrace</td>
<td>Ithaca, NY</td>
<td>14850</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Dr. Michael Friendly</td>
<td>Psychology Department</td>
<td>York University</td>
<td></td>
<td>South Africa</td>
</tr>
<tr>
<td>Dr. Dr. Michael Friendly</td>
<td>York University</td>
<td>Toronto ON</td>
<td></td>
<td>Canada</td>
</tr>
<tr>
<td>Col. Dr. Ernst Frese</td>
<td>Herzophysiologischer Dienst</td>
<td>1126 Wien</td>
<td></td>
<td>Austria</td>
</tr>
<tr>
<td>Dr. Dr. Robert M. Gagne</td>
<td>Psychology Department</td>
<td>Tallahassee, FL</td>
<td>32302</td>
<td>USA</td>
</tr>
<tr>
<td>Dr. Brian R. Gaines</td>
<td>Knowledge Sciences Institute</td>
<td>Calgary, Alberta</td>
<td></td>
<td>Canada</td>
</tr>
</tbody>
</table>

**OAR DISTRIBUTION LIST (ILLINOIS/WILKINS)**
<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ross-n, VA 22021 Meno Park, San Diego, ORSAY, FRANCE Pen..acola, FL Dr. lHoneton, TX</td>
<td></td>
</tr>
<tr>
<td>Dept of Computer Science</td>
<td>Rutgers University New Brunswick, NJ 08902</td>
</tr>
<tr>
<td>Vess M. Moler</td>
<td>NPRDC, Code C4 San Diego, CA 92152-4800</td>
</tr>
<tr>
<td>Dr. Janet Maine</td>
<td>Mail Code EPS NASA Johnson Space Center Houston, TX 77058</td>
</tr>
<tr>
<td>Dr. William L. Maley</td>
<td>NESTPMAA Pensacola, FL 32509-5000</td>
</tr>
<tr>
<td>Dr. Michel Manage</td>
<td>Intellia 36 rue Georges Clemenceau 01400, ORSAY, FRANCE</td>
</tr>
<tr>
<td>Dr. William Mark</td>
<td>Lockheed AI Center Building 314 2718 Sand Hill Rd. Department 9006 Melv Park, CA 90025</td>
</tr>
<tr>
<td>Dr. Sandra P. Marshall</td>
<td>Dept of Psychology San Diego State University San Diego, CA 92182</td>
</tr>
<tr>
<td>Dr. Maxton M. Matthews</td>
<td>Department of Computer Science University of South Carolina Columbia, SC 29008</td>
</tr>
<tr>
<td>Dr. John Meyer</td>
<td>University of Michigan 701 Church Street Ann Arbor, MI 48104</td>
</tr>
<tr>
<td>Dr. John McDermott</td>
<td>DEE DINE 36 E Atlas 280 Donald Lynch Blvd M 100 Frederick 5575</td>
</tr>
<tr>
<td>Prof. David D. McDonald</td>
<td>Department of Computer &amp; Information Sciences University of Massachusetts Amherst, MA 01003</td>
</tr>
<tr>
<td>Dr. Joseph C. McLachlan</td>
<td>Code 43 Navy Personnel R &amp; D Center San Diego, CA 92152-8600</td>
</tr>
<tr>
<td>Dr. Barbara Menas</td>
<td>SRI International 332 Revsveden Avenue Menlo Park, CA 94025</td>
</tr>
<tr>
<td>Dr. Douglas L. Medin</td>
<td>Psychology Department Perry Building University of Michigan 329 Packard Rd. Ann Arbor, MI 48104</td>
</tr>
<tr>
<td>Dr. Theodore Menzler</td>
<td>Department of the Navy Office of the Chief of Naval Research ARLINGTON, VA 22217-5000</td>
</tr>
<tr>
<td>Dr. Alan L. Meyersmith</td>
<td>Office of Naval Research, Code 1430 800 N. Quincy Rd. ARLINGTON, VA 22217</td>
</tr>
<tr>
<td>Dr. Ryvyan S. Michalski</td>
<td>Department of Computer Science George Mason University 4400 University Drives FAIRFAX, VA 22030</td>
</tr>
<tr>
<td>Prof. John Morten</td>
<td>MCC Cognitive Development Unit 17 Garden Street LONDON WC1H 0AH UNITED KINGDOM</td>
</tr>
<tr>
<td>Prof. Jerry M. Miller</td>
<td>CNR-istituto Tecnologia Didattica Via All'Opera Pia 11 GENOVA-ITALIA 16146</td>
</tr>
<tr>
<td>Prof. James R. Miller</td>
<td>MCC 2550 W. Balcony Center Dr. Austin, TX 78716</td>
</tr>
<tr>
<td>Prof. Perry L. Miller</td>
<td>Dept of Anthropology Yale University School of Medicine 330 Cedar St. P. O. Box 2232 New Haven, CT 06520</td>
</tr>
<tr>
<td>Dr. Christine M. Mitchell</td>
<td>School of Indus. and Sys. Eng. Center for Man-Machine Systems Research Georgia Institute of Technology Atlanta, GA 30322-0105</td>
</tr>
<tr>
<td>Dr. Kenneth S. Murray</td>
<td>Dept of Computer Science University of Texas at Austin Taylor Hall 2.124 Austin, TX 78712</td>
</tr>
<tr>
<td>Prof. Melvyn E. Nagae</td>
<td>Dept of Electrical Engineering Kyoto University Yashida-Honmachi Sakyu-Ku Kyoto JAPAN</td>
</tr>
<tr>
<td>Dr. T. Nikketa</td>
<td>The Turing Institute George Street 28 North Hanover Street Glasgow G1 7AD UNITED KINGDOM</td>
</tr>
<tr>
<td>Dr. W. A. N. Nissen</td>
<td>Dept of Mathematics &amp; Computer Science New Campus Ben Geren University of Navy P. O. Box 543 94103 Belf-Sheva ISRAEL</td>
</tr>
<tr>
<td>Dr. A. F. Nerius</td>
<td>Code 5530 Naval Research Laboratory Washington, DC 20375-5000</td>
</tr>
<tr>
<td>Dr. D. A. Norman</td>
<td>C-015 Institute for Cognitive Science University of California La Jolla, CA 92037-0829</td>
</tr>
<tr>
<td>Dr. Harold F. O'Neil, Jr.</td>
<td>School of Education - WPH 801 Department of Educational Psychology &amp; Technology University of Southern California Los Angeles, CA 90009-0001</td>
</tr>
<tr>
<td>Prof. Paul O'Reilly</td>
<td>Department of Information and Computer Science University of California Irvine, CA 92717</td>
</tr>
<tr>
<td>Prof. Debra O'Leary</td>
<td>Learning &amp; Development Center University of Pittsburgh Pittsburgh, PA 15260</td>
</tr>
<tr>
<td>Dr. James R. Murray</td>
<td>Dept of Computer Science University of Texas at Austin Taylor Hall 2.124 Austin, TX 78712</td>
</tr>
<tr>
<td>Dr. James B. Olsen</td>
<td>WICAT Systems 1475 South State Street Orem, UT 84058</td>
</tr>
</tbody>
</table>
ONR DISTRIBUTION LIST [ILLINOIS/WILKINS]

Dr. Judith Reitman Olsen
Graduate School of Business
University of Michigan
731 Tappan
Ann Arbor, MI 48109-1294

Adviser Piper
PM TRADE
ATTN: MCMPC-TMO-BT
1234 Research Parkway
Orlando, FL 32826

Dr. Stephen Reder
ONR
111 SW Main, Suite 500
Portland, OR 97204

Arlington, VA 22207-1000

Office of Naval Research
Code 11473C
870 N. Quincy Street
Arlington, VA 22203

Dr. Piersielli
Graduate School of Education
EMST Division
6322 Telman Hall
University of California, Berkeley
Berkeley, CA 94702

Dr. James A. Regin
University of Maryland
School of Medicine
Department of Neurology
21 South Greene Street
Baltimore, MD 21201

Dr. J. Wesley Regan
APHR/LD1
Brea AFB, TX 78226

Dr. A. A. Revick
Bethesda Laboratories
369 Mountain Avenue
Murray Hill, NJ 07974

Dr. Judith Orsen
Basic Research Office
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Dept. of Administrative Sciences
Code 64
Naval Postgraduate School
Monterey, CA 93943-0228

Dr. Brian Reiner
Cognitive Science Lab
Princeton University
221 Nassau Street
Princeton, NJ 08544

Dr. Stuart J. Russell
Computer Science Division
University of California
Berkeley, CA 94720

Prof. Tim O'Brien
Institute for Education Policy
The Open University
Walton Hall
Milton Keynes MK7 6AA
Buckinghamshire, U.K.

Dr. Tomass Peggie
Massachusetts Institute of Technology E55-101
Center for Biological Information Processing
Cambridge, MA 02139

Dr. Lauren Reznick
Learning R & D Center
University of Pittsburgh
2000 O'Hara Street
Pittsburgh, PA 15213

Dr. Roger C. Schank
Northwestern University
Infor. Sci. and Learning Sciences
1930 Maple
Evanston, IL 60201

Dr. Jesse Ohrncky
Institute for Defense Analyses
1101 N. descendants St.
Alexandria, VA 22211

Dr. Peter Palle
University of Colorado
Department of Psychology
Boulder, CO 80304-0245

Dr. J. Jeffrey Richardson
Center for Applied AI
College of Business
University of Colorado
Boulder, CO 80309-0118

Prof. Christopher K. Riesbeck
Department of Computer Science
Yale University
P. O. Box 2114
New Haven, CT 06520-2118

Dr. Jeffrey C. Schissler
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Dr. Ernest P. Paik
MIT
Laboratory for Computer Science
Room 418
45 Technology Square
Cambridge, MA 02139

Dr. Joseph P. Perk
ATTN: MCMPC-TMO-BT
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22332-1000

Prof. David C. Raia
Dept. of Computer Science
University of Illinois
1404 E. Gregory Drive
Panoas, IL 61803

Dr. Janet W. Schield
818 LDRD Building
University of Pennsylvania
2030 O'Hara Street
Pittsburgh, PA 15213

Dr. R. M. Pasen
Department of Computer Science
UCLA
Irvine, CA 92717

Dr. J. Ross Quinlan
School of Computer Science
N.S.W. Institute of Technology
Broadway
N.S.W. AUSTRALIA 2007

Dr. Edwin L. Risland
Dept. of Computer and Information Science
University of Massachusetts
Amherst, MA 01003

Dr. Paul D. Scott
University of Iowa
Dept. of Computer Science
Winnepasg Park
Columbia, MO 65205

Dr. Ray F. Rees
Institute for Research on Learning
2550 Rowser Street
Palo Alto, CA 94306

Dr. John R. Rees
California Institute of Technology
Pasadena, CA 91125

Dr. Shaker A. Rahwan
Computer Science Department
University of Southern California
Los Angeles, CA 90030

Dr. Linda G. Roberts
Science, Education, and Transportation Program
Office of Technology Assessment
Congress of the United States
Washington, DC 20510

Dr. Albert Segre
Carroll University
Computer Science Department
Upton Hall
Ithaca, NY 14850-2701

Dr. Ray S. Rees
ARD (PERI-I)
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. Shaker A. Rahwan
Computer Science Department
University of Southern California
Los Angeles, CA 90030

Mr. James P. Rez
Cuba, FL 228-18
Naval Surface Weapons Center
White Oak Laboratory
Silver Spring, MD 20902

LT Col. Michael N. Rodgers
Canadian Forces Personnel Applied Research Unit
4030 Yonge Street, Suite 600
Willowdale, Ontario M3B 3E7

Dr. Caleb M. Seiffert
Dept. of Psychology
University of Michigan
355 Packard Rd.
Ann Arbor, MI 48104

Dr. C. Petrie, Chair
Dept. of Psychology
Morgan State University
Cold Spring Ln. - Hillen Rd.
Baltimore, MD 21230

Mr. Margaret Recker
Graduate Group in Education
EMST Division
6322 Telman Hall
University of California, Berkeley
Berkeley, CA 94720

Dr. Oliver C. Selfridge
CTE Labs
Waltham, MA 02214
ONR DISTRIBUTION LIST [ILLINOIS/WILKINS]

Dr. Paul T. Wahlg
Army Research Institute
5001 Eisenhower Ave.
ATTN: PERS-RL
Alexandria, VA 2233-1400

Dr. Joseph Wahl
Alphatech, Inc.
1 Burlington Executive Center
131 Middlesex Turnpike
Burlington, MA 01803

Dr. Beverly P. Woolf
Dept. of Computer and Information Sciences
University of Massachusetts
Amherst, MA 01003

Dr. Ronald R. Yager
Machine Intelligence Institute
Iona College
New Rochelle, NY 10801

Dr. Masoud Yazdani
Dept. of Computer Science
University of Essex
Prince of Wales Road
Essex, ENGLAND

Dr. Joseph L. Young
National Science Foundation
Room 523
1800 G Street, N.W.
Washington, DC 20550

Dr. Maria Zemankova
National Science Foundation
1800 G Street, N.W.
Washington, DC 20550

Dr. Uri Zemik
GS - CRD
P. O. Box 6
Schenectady, NY 12301