Multistrategy Learning and Theory Revision

LORENZA SAITTA
MARCO BOTTA
FILIPPO NERI

SAITTA@DI.UNITO.IT
BOTTA@DI.UNITO.IT
NERI@DI.UNITO.IT

Dipartimento di Informatica, Università di Torino, Corso Svizzera 185, 10149—Torino, Italy

Abstract. This article presents the system WHY, which learns and updates a diagnostic knowledge base using domain knowledge and a set of examples. The a priori knowledge consists of a causal model of the domain that states the relationships among basic phenomena, and a body of phenomenological theory that describes the links between abstract concepts and their possible manifestations in the world. The phenomenological knowledge is used deductively, the causal model is used abductively, and the examples are used inductively. The problems of imperfection and intractability of the theory are handled by allowing the system to make assumptions during its reasoning. In this way, robust knowledge can be learned with limited complexity and a small number of examples. The system works in a first-order logic environment and has been applied in a real domain.

Keywords: Multistrategy learning, causal models, abduction, diagnostic expert systems

1. Introduction

Several authors have advocated the necessity of using deep models of the structure and behavior of the entities involved in a given domain, for instance, in diagnostic systems (Davis, 1984; Genesereth, 1984; Reiter, 1984; de Kleer & Seely Brown, 1986). On the other hand, everyday experience suggests that, in order to achieve high performance in reasonable time, the integrated use of expertise and reasoning from underlying principles should provide the best results (Chandrasekaran & Mittal, 1983; Kahn, 1984).

A shift of attention toward methodologies more focused in a deeper understanding of the world is even more appropriate in learning. And, in fact, this shift is actually going on. Early systems relied on purely inductive techniques (Mitchell, 1982; Michalski, 1983; Quinlan, 1986). Induction can detect regularities in large amount of data, but it can neither explain why these regularities occur nor guarantee their meaningfulness. Nevertheless, inductive systems have been proved viable tools for solving some real-world problems (Michalski & Chilausky, 1980; Quinlan, 1986; Cestnik, Kononenko, & Bratko, 1987; Bergadano, Giordana, & Saitta, 1988, 1990).

The papers by Mitchell, Keller, and Kedar-Cabelli (1986) and DeJong and Mooney (1986) proposed a new, deductive approach to learning that had the great merit of focusing attention on the fundamental role a priori knowledge can play in this process. Deductive learning offers grounds for the knowledge it derives, at the cost of supplying a complete and consistent theory of the domain. Since this last requirement cannot be met in real applications, attempts have been made to take the best of the inductive and deductive approaches by integrating them into a single framework (Lebowitz, 1986; Bergadano & Giordana, 1988; Pazzani, 1988; Danyluk, 1991).

On the other hand, other authors proposed to use deep models of the domain also in learning and to integrate different reasoning schemes (Morris & O'Rourke, 1990; Michalski,

1991 (see also this issue); Tecuci, 1991 (see also this issue); Saitta et al., 1991; Kodratoff, 1991; Mooney & Ourston, 1991; Matwin & Plante, 1991; De Raedt & Bruynooghe, 1991). In this article we are concerned with a particular type of deep models, namely, *causal* models. The notion of causal relation has been widely debated, and different definitions have emerged in various disciplines. AI scientists have also contributed to the discussion (de Kleer & Seely Brown, 1986; Cox & Pietrzykowski, 1987; Poole, 1988; Torasso & Console, 1989; Console et al., 1990). We shall rely, here, on the following intuitive interpretation of causal relation: A is the cause of B if a physical mechanism can be specified that shows how the occurrence of A brings about B. In order to exploit the information embedded in the causal model, we make use of an *abductive* reasoning scheme. Abduction is meant as the process of regressing, through chains of cause-effect relations, from observations to a set of axioms, the first *causes*, which are the ultimate hypotheses we are disposed to accept. Obviously, the definition of the first causes is task dependent and can be suggested by an expert in the domain on pragmatic grounds.

In order to reduce the computational complexity of this type of reasoning, *assumptions* about the state of the world can be made. In this way, the causal model focuses the search for a problem solution toward fundamental phenomena, producing a “deep skeleton” in which “surface” details can be added later. A more extensive analysis of the motivations for using causal models in learning can be found in Saitta et al. (1991).

A causal model $\mathcal{C}$ of a domain does not exhaust all the information needed for learning. In fact, $\mathcal{C}$ might be too abstract to allow practical rules to be learned on its basis alone. Therefore, we shall consider, besides $\mathcal{C}$, a *phenomenological* theory $\mathcal{P}$, stating how abstract concepts can be instantiated in the world. $\mathcal{P}$ contains structural information about objects and definitions of relevant categories and their taxonomic organization; but, more importantly, $\mathcal{P}$ contains a set of rules aimed at describing the manifestations of abstractly defined concepts, i.e., re-expressing them in terms of “operational” predicates (Keller, 1988). The knowledge $\mathcal{P}$, even if necessary in order to fill the gap between abstract entities and their possible manifestations, can only explain *how* things happen, but not *why*. Then, pragmatically, a phenomenon will be considered *justified* only if it is explained by $\mathcal{C}$, i.e., if it is “covered” by a first cause. The distinction between $\mathcal{C}$ and $\mathcal{P}$ is also reflected in the different representation formalisms and in the associated reasoning mechanisms (abductive for $\mathcal{C}$ and deductive for $\mathcal{P}$). In support of the usefulness of keeping $\mathcal{P}$ and $\mathcal{C}$ apart, we notice that in diagnostic problems, for instance, the same malfunction (recognizable from the symptoms specified by $\mathcal{P}$) may often have different causes (specified by $\mathcal{C}$) and that only the knowledge of the true cause allows an effective repair.

We shall now describe how the above-mentioned ideas have been implemented in the system WHY, developed as an extension of the learning system ML-SMART (Bergadano, Giordana, & Saitta, 1988), in which inductive and deductive techniques have already been integrated (Bergadano & Giordana, 1988). WHY has been applied to a real-world problem of mechanical troubleshooting (Saitta et al., 1991). However, a simpler example of a heat transfer problem will be used in this article, for the sake of comprehensibility.

Several aspects differentiate the system WHY from others devoted to similar learning tasks. First of all, WHY is able to deal with first-order logic languages, representing both the domain theory and the target knowledge, by keeping track of the (possibly multiple) bindings between variables in the formulas and components of the instances. Second, the

system can be used for acquiring and refining a target knowledge base as well as for revising the a priori knowledge supplied to it. In this last task, the system works in a semi-automated way, allowing a direct interaction with the expert, who can suggest modifications to the background or to the target knowledge; the expert will receive an immediate feedback to his/her suggestions, because WHY evaluates the effects of the modifications and presents them back to the expert for validation. In order to ease this interactive aspect, WHY is provided with a friendly interface toward the user, based on a frame system supporting a menu-like interaction protocol. Another innovative aspect of WHY is the integration of the performance element into the learner, so that they both use the same knowledge structures and the same reasoning process can serve both classification and learning.

This article is organized as follows. Section 2 describes the representation formalisms used for the various types of knowledge used by WHY. Section 3 describes the basic reasoning mechanisms in isolation, whereas section 4 illustrates how these reasoning schemes can be combined to build up justifications. In section 5, one-step learning and the refinement of the target knowledge are described, whereas section 6 is devoted to the interactive theory revision phase. Section 7 contains some conclusions.

2. Knowledge representation

WHY handles four bodies of knowledge: the causal model, $\mathcal{C}$, the phenomenological knowledge, $\mathcal{P}$, the target knowledge base, $\mathcal{KB}$, and the control knowledge.

The representation schemes for all the kinds of knowledge share a first-order logic language $\mathcal{L}$ in Horn clause form. In the following, a double arrow $\Rightarrow$ will denote material implication, whereas the simple arrow $\rightarrow$ a causal link. Moreover, lower-case letter predicates are operational (i.e., directly observable on the data), whereas capital letter predicates are non-operational. These two types of predicates belong to the sets $P^{(o)}$ and $P^{(n)}$, respectively.

2.1. Target knowledge base

As a first approximation, the target knowledge base can be viewed as an unordered set of decision rules:

$$r \equiv \forall e [\exists x \mid \varphi(e, x) \Rightarrow h(e)] \quad (1)$$

where x denotes a set of variables, e an event to be interpreted, h a concept belonging to a set H_0 , and φ a formula of $\mathcal{L}$. Actually, the target knowledge base has a more complex representation, as will be explained in section 5.

The set $S(h)$ of rules, whose conclusion is h , is an *intensional* description of the concept h . During the learning process, each $\varphi(e, x)$ has also an *extensional* representation, consisting of all the training examples satisfying φ ; the extension of φ is stored as a relation in a relational data base (Bergadano, Giordana, & Saitta, 1988). More precisely, let F be a

set of (structured) events and $\varphi^{(o)}(x_1, x_2, \dots, x_k)$ an operational formula. Given an event $e \in F$, every possible binding, satisfying $\varphi^{(o)}$, between k -tuples of components of e and the k -tuple of variables $x_1, x_2, \dots, x_k$ is represented in a $(k + 2)$ -ary relation $\varphi^* = \langle F, H, X_1, X_2, \dots, X_k \rangle$. The field F contains the identifier of the example and H its correct classification (the concepts, which e is an instance of); moreover, $X_1, X_2, \dots, X_k$ are the names of the constants (components of e) bound to the variables $x_1, x_2, \dots, x_k$. Notice that several $(k + 2)$ -tuples, belonging to the same event, may occur in φ^* .

2.2. Causal model

The logical representation of the causal model $\mathcal{C}$ consists of an acyclic graph. In figure 1 a very simple model of heat transfer phenomena is reported. Nodes in the network are either *primary* (ellipses) or *accessory* (rectangles and clouds). Primary nodes correspond to processes or system states, and a subset of these nodes contains the *first causes* (shaded nodes), which do not have entering edges. Effects of the same cause are AND-ed; causes of the same effect are OR-ed.

Accessory nodes represent two types of additional information: *constraints* (rectangles) and *contexts* (clouds). Constraints are attached to edges and correspond to conditions that are to be necessarily verified in order to be able to instantiate the corresponding cause-effect relation. Contexts are attached to primary nodes and represent the environmental conditions allowing the instantiation of the related phenomenon. One of the differences between constraints and contexts is the following: constraints describe physical properties of the involved bodies or structural information, i.e., properties that do not depend on time.

Figure 1. Simplified model of heat transfer phenomena.

Contexts state conditions that are possibly only temporarily true, and usually involve parts of the considered system larger than the one currently under analysis; for example, conditions may refer to system components different from the one for which the causal network has currently been activated. In figure 1, the explanation of why a body x is hot may involve the presence of a hot body y , adjacent to it, from which heat can flow toward x through heat conduction.

Primary nodes of the network may or may not be observables, i.e., evaluable either directly on the data (if their description involves operational predicates only) or indirectly, using the phenomenological theory $\mathcal{P}$. Constraints and contexts must be observable; as an exception, contexts may also be described in terms of primary causal nodes, introducing recursion in the net. This is the case of $\text{HOT}(y)$ in the contexts of $\text{IRRADIATION}(y, x)$ and $\text{CONDUCTION}(y, x)$ in figure 1.

Finally, a triangle associated to a node contains that part of the phenomenological theory $\mathcal{P}$ that describes the manifestations associated to the node. As an example, triangle P_1 contains the following set of rules, allowing the predicate $\text{CIRCUIT}(y, r)$ to be evaluated:

$$\begin{aligned} &\text{TERMINALS}(y) \wedge \text{RESISTANCE}(r) \wedge \text{connected-to}(y, r) \Rightarrow \text{CIRCUIT}(y, r) \\ &\text{plug}(y) \Rightarrow \text{TERMINALS}(y) \\ &\text{terminals}(y) \Rightarrow \text{TERMINALS}(y) \\ &\text{ELT-CABLE}(x, y) \Rightarrow \text{TERMINALS}(y) \\ &\text{cord}(z) \wedge \text{plug}(y) \wedge \text{connected-to}(y, z) \Rightarrow \text{ELT-CABLE}(z, y) \\ &\text{resistance}(r) \Rightarrow \text{RESISTANCE}(r) \end{aligned}$$

Operationalizing predicates occurring in nodes of $\mathcal{C}$ is not the unique link between $\mathcal{P}$ and $\mathcal{C}$; in fact, the name of a node of $\mathcal{C}$ can occur in the body of some clauses defining non-operational predicates in $\mathcal{P}$. This is also the way the links between $\mathcal{C}$ and the concepts in H_0 are established; as a matter of fact, concepts are non-operational predicates defined in $\mathcal{P}$. Suppose, for instance, that we want to use the network of figure 1 to characterize objects that are dangerous to be touched because they burn. We define the class $\text{UNSAFE-TO-TOUCH}(x)$ as follows (names of nodes occurring in $\mathcal{C}$ are underlined in $\mathcal{P}$):

$$\begin{aligned} &\underline{\text{HOT}}(x) \Rightarrow \text{UNSAFE-TO-TOUCH}(x) \\ &\underline{\text{HOT}}(y) \wedge \text{obj}(x) \wedge \text{part-of}(y, x) \Rightarrow \text{UNSAFE-TO-TOUCH}(x) \end{aligned}$$

2.3. Phenomenological theory

The phenomenological theory $\mathcal{P}$ consists of a set of Horn clauses describing taxonomies, structural information, general knowledge, and links between abstract concepts and manifestations. For instance:

$$\begin{aligned} &\text{table}(x) \Rightarrow \text{FURNITURE}(x) \\ &\quad (\text{taxonomic knowledge}) \\ &\text{NEAR}(z, y) \wedge \text{part-of}(x, z) \Rightarrow \text{NEAR}(x, y) \\ &\quad (\text{general knowledge}) \end{aligned}$$

$$\begin{array}{ll}
\text{cord}(x) \wedge \text{plug}(y) \wedge \text{connected-to}(y, x) \Rightarrow \text{ELT-CABLE}(x, y) & \text{knowledge)} \\
(\text{structural} & \\
\text{bright}(x) \Rightarrow \text{INCANDESCENT}(x) & \\
(\text{manifestation}) &
\end{array}$$

Another example of $\mathcal{P}$ is the rule set P_1 reported before. Notice that the causal network $\mathcal{C}$ subdivides $\mathcal{P}$ into subsets, allowing the reasoner to consider only limited parts of $\mathcal{P}$ at any given time.

2.4. Theory incompleteness

When encoding a domain theory, incompleteness may affect almost any part of it (DeJong, 1990), and we may not be aware of where and what knowledge is missing. On the other hand, there are cases in which either we know exactly where some information is missing, but we do not know what information, or we want to hide information on purpose (for instance, for tractability reasons). In these cases we would like to warn the system about this lack of information. To this aim, the special predicate Ω (unknown) is introduced. For example, the rule

$$\text{elt-plate}(x) \wedge \Omega \Rightarrow \text{HEAT-SOURCE}(x) \quad (2)$$

means that an electric plate *may* be a heat source, if some other unspecified condition is verified (for instance, if it is plugged in). As an extreme case, Ω may coincide with the whole body of a clause, as in the following:

$$\Omega \Rightarrow Q(x) \quad (3)$$

Rule (3) denotes that we are unable (or do not want) to specify how the predicate $Q(x)$ can be operationalized. Sometimes, we are in an intermediate situation: the expert can tell that the predicate $Q(x)$ is expressible in terms of a combination of other predicates $\{R_1, R_2, \dots, R_n\}$, either operational or non-operational, but he/she does not know exactly which ones. Then, we associate the set $\{R_1, R_2, \dots, R_n\}$ to Ω in a *dependency* rule (Bergadano, Giordana, & Ponsero, 1989):

$$\{R_1, R_2, \dots, R_n\} \Rightarrow \Omega \Rightarrow Q(x) \quad (4)$$

which can be used by the inductive reasoner as a reduced search space.

2.5. Control knowledge

The system WHY, as well as its parent ML-SMART, is a knowledge-intensive learner, which has a body of declarative knowledge to control its behavior. The control acts at two main levels: the high-level control chooses among the available reasoning schemes (abduction,

induction, deduction) the most suitable one for each learning step. The low-level control determines the behavior of a single learning strategy. The control knowledge is implemented as a set of frames, containing the description of a situation and the action(s) suitable to handle that situation.

3. Basic reasoning mechanisms

Three basic reasoning schemes are integrated in the system WHY: induction, deduction, and abduction. Each of these is briefly illustrated in the following.

3.1. Induction

Induction is used to fill the possible gaps in the domain theory and is performed by invoking the inductive module of the ML-SMART system. This module searches, using a general-to-specific strategy, in a space of first-order logical formulas and outputs a network of decision rules, whose left-hand sides are the selected formulas. A body of control knowledge, in declarative form, is used to guide the search toward the most promising paths and to evaluate candidate formulas for inclusion in the final rule set. Criteria for evaluating formulas include (but are not limited to) their completeness and consistency.

ML-SMART is also responsible for maintaining the relational data base, in which the examples and the extensions of the rules are stored. Multiple bindings between variables in the formulas and parts in the examples are allowed: this feature proved to be very useful during the justification process, as explained in the next sections. An extensive description of ML-SMART can be found in Bergadano, Giordana, and Saitta (1988).

3.2. Deduction

Deduction is also performed using ML-SMART's deductive module, as described in Bergadano and Giordana (1988). The module handles a first-order theory expressed in Horn clause form and is based on Robinson and Siebert's (1982) LOGLISP. The deductive mechanism deals with many examples at the same time, exploiting the existing data base in ML-SMART. A novelty introduced by WHY is the possibility of also performing a forward deduction, i.e., of building the operationalization tree starting from the operational predicates true of the considered examples.

3.3. Abduction

Abductive reasoning is needed to exploit the causal model. In figure 2 a typical part of the causal network is reported, for the sake of illustration of the basic reasoning steps.

Reasoning with the causal model can be performed by moving from effects to causes (search for an explanation via abduction) or from causes to effects (prediction via deduction) in order to assert the truth status of each node:

Figure 2. Scheme of the relations among nodes in the causal network. Two basic reasoning steps are defined: deduction from causes to effects and abduction from effects to causes.

- Deduction from causes to effects:

$$N_1 \wedge CTX_1 \wedge \gamma \rightarrow N_2$$

If node N_1 is true in the context CTX_1 and the constraints γ are verified, then node N_2 deterministically (if Ω does not occur in γ) or plausibly (if Ω occurs in γ) follows.

- Abduction from effects to causes:

$$N_2 \wedge \gamma \rightarrow N_1 \wedge CTX_1$$

If effect N_2 has been observed (or hypothesized) and the constraints γ are verified, then we can hypothesize that N_2 was originated by N_1 in the context CTX_1 . In this abductive step, Ω does not have any influence.

Notice that the truth status of a node can also be determined by its observability, i.e., by its evaluation on the data (rules P_1 , for instance), independently of the presence of its causes and/or effects. This fact constitutes another connection between phenomenological and causal reasoning.

The above reasoning steps, as simple as they appear, are quite complex to be implemented in a first-order logic environment, because of the variable bindings. How to maintain these bindings and the truth status of each node (possibly different for each binding of each instance) will be described in section 4.

The “context” nodes play a particular role in the causal reasoning. Suppose that we are trying to explain why a given object x is hot. The causal network tells us that there are a number of causes justifying this phenomenon; however, some of these are acceptable only if the environment supplies the conditions to support them. For instance, heat conduction requires that there is an object y , different from x , which is, in turn, hot and adjacent to x . When the causal reasoning is invoked on a given part of the system (in the example, the object x), contexts act as defaults and are always assumed true. Then, each of the first causes, hypothesized to explain the phenomenon, has its context hypothesized at the same time. It is up to the general control strategy to decide whether accepting the assumed context by default or to verify it, possibly re-entering the causal network. This strategy closely matches the one often used by human experts and has also the advantage of breaking recursion in the causal reasoning.

3.4. Handling assumptions

Making assumptions is a fundamental aspect of the reasoning process in uncertain environments and, in particular, when only an incomplete theory is available. In WHY, assumptions can be made during both the phenomenological and the causal reasoning, but the ways they are handled in the two cases differ.

Let us consider a set of clauses in $\mathcal{P}$, sharing the same head:

$$\varphi_k(\mathbf{x}) \Rightarrow P(\mathbf{x}) \quad (1 \leq k \leq n)$$

According to Clark's completion (Clark, 1978), we can assume that

$$P(\mathbf{x}) \Leftrightarrow \varphi_1(\mathbf{x}) \vee \dots \vee \varphi_n(\mathbf{x}) \quad (5)$$

If the predicate Ω does not occur in any of the $\varphi_k(\mathbf{x})$'s, then using rule (5), predicate $P(\mathbf{x})$ can be proved false if and only if all the $\varphi_k(\mathbf{x})$ ($1 \leq k \leq n$) are false. Things are different if Ω does occur. If the predicate occurs in the body of a clause, as, for instance, in (2), the effect of Ω is that $P(\mathbf{x})$ will never be true, but can only be *assumed* true if the other predicates in the body of the rule are true. If a clause like (3) occurs, then Clark's completion of P will contain it as a disjunct:

$$P(\mathbf{x}) \Leftrightarrow \varphi_1(\mathbf{x}) \vee \dots \vee \varphi_n(\mathbf{x}) \vee \Omega \quad (6)$$

According to (6), $P(\mathbf{x})$ is never false, because Ω is, by default, assumed true. The number of assumptions made during reasoning is an important strategic parameter.

In the causal network, the symbol Ω may only occur in constraints and contexts. Predicates occurring in constraints and different from Ω are *necessary*, in the sense that they are assumed false if they cannot be proved true. The presence of Ω in a constraint denotes a weakening in the causal relation associated with the corresponding edge: in this case, the cause *may* produce the effect, but not necessarily. For example, in figure 1, an object x can become hot upon heating only if a sufficient time has elapsed or if it is not melting. Effects generated by a cause in a possibilistic way have a status of assumed true even if the cause is true.

4. Integrating multiple strategies

An abstract description of the overall learning process can be given by outlining its main phases:

ELICITATION:

Elicit from the expert a body of background knowledge, partitioned into a causal network and a phenomenological theory $\mathcal{J} = \mathcal{C} \cup \mathcal{P}$, using the USER INTERFACE. Moreover, let F_0 be a set of training examples.

JUSTIFICATION:

- Phase 1** — Build up an AND/OR forest $\mathcal{G}$, called the *justification forest*, by applying the forward deduction algorithm FORW-DED to the phenomenological theory $\mathcal{P}$. Insert in the data base the extension, evaluated on the set F_0 , of each node in $\mathcal{G}$. Activate some nodes of the causal network $\mathcal{C}$.
- Phase 2** — Apply the abductive algorithm CAUSAL to find paths in the causal network, regressing from the activated nodes toward first causes. Insert in the data base the extension, evaluated on the set F_0 , of each node in each causal path activated in $\mathcal{C}$.
- Phase 3** — Verify hypothesized contexts, possibly recursively re-entering the causal network.

ONE-STEP LEARNING:

- Generate decision rules for the concepts in H_0 , by collecting the predicates which are true (or assumed true) along the activated causal paths, and insert the rules in $\mathcal{KB}$.
- If** The rules have an “acceptable” degree of completeness and consistency
 Then Stop
 Else Go to KNOWLEDGE REFINEMENT
- If** Some contradiction arises
 Then Go to THEORY REVISION

KNOWLEDGE REFINEMENT:

Given an unknown example f
 Classify f using $\mathcal{KB}$.

- If** The classification is correct
 Then Update the data base.
 Update the justification forest.
- Else** Locate on $\mathcal{G}$ the node(s) responsible for the error.
- If** There is an omission error
 Then Generalize $\mathcal{KB}$, possibly by invoking induction and/or interacting with the expert.
- If** There is a commission error
 Then Specialize $\mathcal{KB}$, possibly by invoking induction and/or interacting with the expert.
- Update the data base.
 Update the justification forest.

THEORY REVISION:

- If** A justification should have been found and was not
 Then Invoke induction for filling a gap in $\mathcal{P}$ or ask the expert to generalize $\mathcal{P}$ and/or extend $\mathcal{C}$.
- If** A justification was found and should not have been found
 Then Ask the expert to specialize $\mathcal{P}$ and/or modify $\mathcal{C}$.
- Evaluate the consequences of the suggested modifications.
 Ask the expert to choose his/her preferred modifications and add them to $\mathcal{J}$.
 Update all data structures.

In the following, the justification process will be described, as well as the relations between the justification forest and the target knowledge base. For the sake of exemplification, we shall use the task, mentioned in section 3, of characterizing possibly UNSAFE-TO-TOUCH objects, given a set of 14 examples and 18 counterexamples of this concept, taken from everyday life (irons, pots, lamps, knives, etc.).

4.1. Construction of the justification forest

In order to describe how the system works, some definitions have to be introduced first. Let $\mathfrak{I} = \mathcal{C} \cup \mathcal{P}$ be the domain theory. Let, moreover, Σ be the set of primary nodes in the causal network $\mathcal{C}$ and Σ_0 the subset of nodes corresponding to the first causes. Two kinds of justifications will be considered: the first one (analogous to the one introduced in EBL) refers to *instances* and is an explanation, in terms of cause/effect relations, of why an example f is an instance of a concept h .

Definition 1 — Given a theory $\mathfrak{I} = \mathcal{C} \cup \mathcal{P}$ and an instance f of a concept h (denoted by $h(f)$), we call *explanation* $E(h, f)$ of f w.r.t. h the pair $\{\sigma_0(f), A\}$ such that:

$$\mathfrak{I} \cup \sigma_0(f) \cup A \vdash h(f)$$

where $\sigma_0(f)$ is a first cause, true of f , and A is a set of assumptions, i.e., a set of predicates belonging to $\mathcal{P}$, which have to be assumed true in order to derive $h(f)$. $\square$

The second type of justification refers to *rules* and *instances* and explains why a given rule is correctly verified by an instance f of a concept h .

Definition 2 — Given a rule $r \equiv (\varphi(z, x) \Rightarrow h(z)) \in \mathcal{KB}$, and an instance f of h , let $\varphi^{(o)}(f)$ be the operationalization of φ true of f . Then, a *justification* $J(r, f)$ of r w.r.t. f is a 4-tuple $\{\sigma_0(f), A, \psi^{(o)}, \rho\}$ such that:

- a) $\mathfrak{I} \cup \sigma_0(f) \cup A \vdash (\psi^{(o)}(f) \Rightarrow h(f))$
- b) $\psi^{(o)} = \rho \wedge \xi_1$ and $\varphi^{(o)} = \rho \wedge \xi_2$ (i.e., $\psi^{(o)}$ and $\varphi^{(o)}$ have a subformula ρ in common)
- c) $\sigma_0(f)$ is a first cause, true of f , and A is a set of assumptions $\square$

We distinguish between *total justification*, in which $\rho \equiv \varphi^{(o)}$ and *partial justification*, in which $\xi_2 \neq \emptyset$. Definitions 1 and 2 can be generalized to the case of a set of examples F , obtaining $E(h, F)$ and $J(r, F)$, respectively.

The justification forest $\mathcal{G}$ is an extension of the classical explanation tree used in EBL to the case of simultaneous operationalization of several predicates on many examples. The construction of $\mathcal{G}$ is performed by the algorithm *FORWDED*, which starts from the set of operational predicates $P^{(o)}$ and applies forward the rules in $\mathcal{P}$ until either no more rules are applicable or some primary node of the causal network has been reached.

Algorithm FORW-DED

Let Stimuli be a set of activated rules in $\mathcal{O}$ and let $P^{(o)}$ be the set of operational predicates. (A rule is *activated* when at least one conjunct in its left-hand side is true of at least one example).

$\mathcal{G} = \emptyset$.

Initialize Stimuli with the rules activated by the predicates in $P^{(o)}$.

Add all the predicates in $P^{(o)}$, true on some example, to $\mathcal{G}$ as leaves.

Create in the data base the relation associated to each leaf.

Repeat

(1) Select a rule $r \equiv \varphi \rightarrow P$ from Stimuli.

Operationalize P through r by completely unfolding φ .

If $P \in \mathcal{G}$ **Then** Add to $\mathcal{G}$ the new operationalization as an OR-son of P .

Update relation P^* by merging the old with the new one.

Else Create a new node P in $\mathcal{G}$ and associate to it P^* .

If $P \in \Sigma$ (primary nodes in $\mathcal{O}$)

Then Remove r from Stimuli and Go To (1).

Remove r from Stimuli.

Add to Stimuli all the rules in whose body P occurs.

Until Stimuli = $\emptyset$.

In figure 3, a part of the justification forest built up for the 32 examples of unsafe and safe objects is reported.

The set of numbers associated with each node in figure 3 represents the set of examples satisfying the formula associated to the node. Notice that, for each example, more than one binding is possible. A major source of complexity in the construction of $\mathcal{G}$ is the need

Figure 3. Part of the justification forests for the considered set of 14 unsafe-to-touch and 18 safe-to-touch objects.

to maintain identity constraints among variables in AND nodes. In fact, the bindings between variables and components in the examples are performed independently from one another in the son nodes (being the forest built up bottom-up). As an example, all the pairs of objects connected each other are stored in the extension of the node $\text{conn-to}(u, v)$; however, only those pairs of connected objects that are a cord and a plug, respectively, are to be selected to form an electric cable. Therefore, it may happen that an AND node is false, but their son nodes are true, in isolation. This detailed information about bindings is fundamental for the tasks of knowledge refinement and theory revision, because it allows the location where an explanation fails to be exactly individuated.

Additional pieces of information are also added to each node: one is its possible connection with primary or secondary nodes of $\mathcal{C}$, and another is a *truth status*, which can assume one among the set $\{\text{True}, \text{Assumed True}, \text{Assumed False}, \text{False}\}$ of values. An “Assumed True” status may derive from the presence of an Ω within an operationalization or from the assumption of some predicate that was not specified in the data, but could not be proved false. Obviously, the status is not a global property of the node, because different assumptions might have been made for different examples. Then, the status is evaluated for every binding of each example and stored, as an additional field in each tuple, in the relation associated to the node.

4.2. Finding causal paths

During the preceding phase, a subset of the nodes in Σ have been activated. Now, we want to use abduction to hypothesize first causes, moving back from the activated nodes. The abductive process is performed by the following algorithm *ABDUCTION*. For the sake of simplicity, the parts of the algorithm handling the extensional representation of the causal nodes and the assumptions have been omitted.

Algorithm ABDUCTION

Let σ be an activated node from which a causal path starts, and let CAUSES be the set of immediate causes of σ .

Let SUCCESS be the set of paths successfully arrived to a first cause from σ and FAIL the set of failed paths.

SUCCESS = FAIL = $\emptyset$

CAUSES = {Immediate causes of σ }

For each node $\sigma_i \in \text{CAUSES}$ do

Let path_i be a sequence of causal nodes, starting from σ

Let γ_i be the (possibly empty) constraint on the edge connecting σ_i to σ in $\mathcal{C}$

Evaluate the formula $\text{path}_i = \sigma \wedge \gamma_i \wedge \sigma_i$

If No example verifies path_i or (σ_i has a subset P_i of $\mathcal{P}$ associated to it and No example verifies P_i)

Then FAIL = FAIL $\cup \{\text{path}_i\}$

Else Hypothesize σ_i with its associated context CTX_i (if any)

SUCCESS = SUCCESS $\cup \{\text{path}_i\}$

Remove σ_i from CAUSES.

The above algorithm describes how a single step of abduction is performed. Sequences of abduction steps belonging to the same path are kept together in a single global description. When an exhaustive search for causal paths is too expensive, paths that currently have fewer assumptions are followed first. In the example of figure 3, the following causal paths have been successfully completed (among others):

Path₁ = $\langle \underline{HOT}(x), \underline{HEATING}(x), \text{GOOD-HEAT-CONDUCTOR}(x), \underline{IN-THERMAL-FLUX}(x), \underline{CONDUCTION}(y, x), \text{HOT}(y), \text{ADJACENT}(x, y) \rangle$ for the set of examples $\{1, 2, 4, 6, 16, 17, 19, 23\}$

Path₂ = $\langle \underline{HOT}(x), \underline{HEATING}(x), \text{GOOD-HEAT-CONDUCTOR}(x), \text{INSIDE}(r, x), \underline{ELT-POWER-DISSIPATION}(r), \underline{ELT-CURRENT}(r), \text{CIRCUIT}(y, r), \underline{ELT-VOLTAGE}(y) \rangle$ for the set of examples $\{1, 2, 4, 6\}$

In the paths, underlined predicates are primary nodes in $\mathcal{C}$, whereas italic predicates belong to contexts. The examples involved are four electric irons $\{1, 2, 4, 6\}$, which are hot because of the Joule effect produced by the electric current flow, and different types of pots $\{16, 17, 19, 23\}$, which are hot because of the heat conduction from some heat source. During causal justification, the context $\text{HOT}(y) \wedge \text{ADJACENT}(x, y)$, associated with the first cause $\text{CONDUCTION}(y, x)$ is hypothesized at the same time as the first cause. Then, $\text{CONDUCTION}(y, x)$ is a potential explanation also for the four irons. Due to this ambiguity, the hypothesized context is checked, and turns out to be false for $\{1, 2, 4, 6\}$ and true for the remaining objects. Notice that, during verification of the context, the predicate $\text{ADJACENT}(y, x)$ is tested first, because it does not involve recursion. In this way, for the three irons $\{1, 2, 4\}$ this verification fails immediately and the re-activation of the causal reasoning on $\text{HOT}(y)$ is necessary only for $\{6\}$. However, this also fails, because the iron is adjacent to an object that is cold. Then, the first cause $\text{CONDUCTION}(y, x)$ only covers examples $\{16, 17, 19, 23\}$, whereas $\underline{ELT-VOLTAGE}(y)$ only covers $\{1, 2, 4, 6\}$.

5. Acquisition and refinement of the target knowledge

After the process of building up, for the training examples, the justification forest and the set of causal paths, learning can start. In particular, an initial $\mathcal{KB}$ is extracted and afterwards can be refined, if it is the case.

5.1. One-step learning

From the causal paths, an initial set of decision rules is generated, one rule for each set of explained examples. For the sake of exemplification, let us consider the two sets of examples $\{1, 2, 4, 6\}$ and $\{16, 17, 19, 23\}$ for which the final causal explanations are the following:

Cause₁ = $\langle \underline{HOT}(x), \underline{HEATING}(x), \text{GOOD-HEAT-CONDUCTOR}(x), \underline{IN-THERMAL-FLUX}(x), \underline{CONDUCTION}(y, x), \text{HOT}(y), \text{ADJACENT}(x, y) \rangle$ for the set of examples $\{16, 17, 19, 23\}$

$\text{Cause}_2 = \langle \underline{\text{HOT}}(x), \underline{\text{HEATING}}(x), \text{GOOD-HEAT-CONDUCTOR}(x), \text{INSIDE}(r, x), \underline{\text{ELT-POWER-DISSIPATION}}(r), \underline{\text{ELT-CURRENT}}(r), \text{CIRCUIT}(y, r), \underline{\text{ELT-VOLTAGE}}(y) \rangle$ for the set of examples $\{1, 2, 4, 6\}$

In order to complete a causal path, several predicates, not denoting primary causal nodes, have been verified and associated with the path. In particular, let ∂ be the conjunction of the necessary conditions (constraints) and ψ the conjunction of the remaining ones. For Cause_1 the two sets are

$$\partial\{16, 17, 19, 23\} = \text{GOOD-HEAT-CONDUCTOR}(x)$$

$$\psi\{16, 17, 19, 23\} = \text{HOT}(y) \wedge \text{ADJACENT}(x, y),$$

whereas, for Cause_2 , we obtain (by operationalizing $\underline{\text{ELT-VOLTAGE}}(y)$ by means of $\text{PLUGGED-IN}(y)$):

$$\partial\{1, 2, 4, 6\} = \text{GOOD-HEAT-CONDUCTOR}(x) \wedge \text{INSIDE}(r, x) \wedge \text{CIRCUIT}(y, r)$$

$$\psi\{1, 2, 4, 6\} = \text{PLUGGED-IN}(y)$$

As a first approximation, we shall consider the following heuristic decisions rules:

$$\begin{aligned} r_1 &\equiv \text{GOOD-HEAT-CONDUCTOR}(x) \wedge \text{HOT}(y) \wedge \text{ADJACENT}(x, y) \Rightarrow \\ &\Rightarrow \text{UNSAFE-TO-TOUCH}(x) \end{aligned} \quad (7)$$

$$\begin{aligned} r_2 &\equiv \text{GOOD-HEAT-CONDUCTOR}(x) \wedge \text{INSIDE}(r, x) \wedge \text{CIRCUIT}(y, r) \wedge \\ &\text{PLUGGED-IN}(y) \Rightarrow \text{UNSAFE-TO-TOUCH}(x) \end{aligned} \quad (8)$$

The distinction between the necessary conditions (in *italics*) and the others is kept as such in the heuristic rules, because we want the performance system to manage them in different ways: necessary conditions are assumed false if they cannot be proved true, whereas the other conditions are assumed true if they cannot be proved false, during classification, as well.

As we can notice, rules (7) and (8) contain non-operational predicates. This is allowed, not only for making assumptions in classification, but also for obtaining a more compact $\mathcal{KB}$. Let us consider, for instance, the predicate $\text{GOOD-HEAT-CONDUCTOR}(x)$; we know that metals are good heat conductors, so rule (7) may correspond to a possibly large number of operational rules, each one obtained by substituting $\text{steel}(x)$, $\text{aluminum}(x)$, $\text{silver}(x)$, etc., in place of $\text{GOOD-HEAT-CONDUCTOR}(x)$. These rules are essentially the same rule. It is, then, more convenient to allow non-operational predicates to occur in the left-hand sides of the rules (Segre, 1987; Hirsh, 1988). Another advantage can be obtained with this choice: we do not need to write a $\mathcal{KB}$ separate from the justification structure. It is sufficient to link, in $\mathcal{G}$, the non-operational predicate $\text{UNSAFE-TO-TOUCH}(x)$ to an

AND node corresponding to the left-hand side of each rule. In this way, the process of classifying a new instance coincides with that of justifying it and also of locating the failure point when an error occurs.

In the example given above, rules (7) and (8) are consistent (they do not cover any counterexample), but they are not complete (only eight examples are covered by them). It turns out that one more consistent rule can be obtained in this phase, covering a hot pan inside an oven. For the remaining objects (three types of bulbs and two cups filled with hot coffee), no explanation can be obtained (all paths are cut because of the failure of necessary conditions), and, hence, no rule. This case requires a session of theory revision. Even if in the example presented this is not the case, it may also happen that explanations can be found by making some assumptions. Then, the obtained rules may not be consistent. This case requires a session of rule refinement.

5.2. Knowledge refinement

Knowledge refinement will only be briefly outlined here, because it has already been discussed elsewhere (Botta, Giordana, & Saitta, 1990; Baroglio, Botta & Saitta, 1992). It is not necessary that knowledge refinement is invoked on a $\mathcal{KB}$ acquired as described in the previous subsection. On the contrary, the $\mathcal{KB}$ may come from a different source, for instance, from an expert. Hence, let us suppose that someone suggested the following rule:

$$r \equiv \text{OBJ}(x) \wedge \text{body}(y) \wedge \text{part-of}(y, x) \wedge \text{closed}(y) \wedge \text{METAL}(y) \wedge \text{CABLE}(z) \wedge \\ \text{part of}(z, x) \wedge \text{medium-size}(x) \Rightarrow \text{UNSAFE-TO-TOUCH}(x) \quad (9)$$

By considering that $\text{CABLE}(x)$ can be operationalized by $\text{thread}(x)$ and by $\text{ELT-CABLE}(x, y)$, rule (9) has two alternative operationalizations on the set of 32 training examples:

$$r_1 \equiv \text{OBJ}(x) \wedge \text{body}(y) \wedge \text{part-of}(y, x) \wedge \text{closed}(y) \wedge \text{aluminum}(y) \wedge \text{thread}(z) \wedge \\ \text{part-of}(z, x) \wedge \text{medium-size}(x) \Rightarrow \text{UNSAFE-TO-TOUCH}(x) \quad (10)$$

$$r_2 \equiv \text{OBJ}(x) \wedge \text{body}(y) \wedge \text{part-of}(y, x) \wedge \text{closed}(y) \wedge \text{steel}(y) \wedge \text{cord}(z) \wedge \text{plug}(w) \wedge \\ \text{connected-to}(w, z) \wedge \text{part-of}(z, x) \wedge \text{medium-size}(x) \Rightarrow \\ \Rightarrow \text{UNSAFE-TO-TOUCH}(x) \quad (11)$$

Rule r_1 correctly covers a hot electric iron, but incorrectly covers a windowshade handle, which is safe to touch. On the other hand, r_2 correctly covers three plugged-in irons and incorrectly covers two unplugged irons. If we try to justify rule r_1 , we notice that the windowshade handle belongs to the extension of $\text{CABLE}(x)$, but not to that of $\text{ELT-CABLE}(x, y)$. Therefore, $\text{CABLE}(x)$ is too general, and the counterexample can be cut by specializing (9) as follows:

$$\begin{aligned} \text{OBJ}(x) \wedge \text{body}(y) \wedge \text{part-of}(y, x) \wedge \text{closed}(y) \wedge \text{METAL}(y) \wedge \text{ELT-CABLE}(z, w) \wedge \\ \text{part of}(z, x) \wedge \text{medium-size}(x) \Rightarrow \text{UNSAFE-TO-TOUCH}(x) \end{aligned} \quad (12)$$

Rule (12) still covers two unplugged irons. Using the algorithm for finding causal paths, described in section 4, we can prove that, for the set of hot irons,

$$\begin{aligned} \exists \vdash [\text{OBJ}(x) \wedge \text{body}(y) \wedge \text{part-of}(y, x) \wedge \text{METAL}(y) \wedge \text{cord}(z) \wedge \text{plug}(w) \wedge \\ \text{connected-to}(w, z) \wedge \text{resistance}(u) \wedge \text{socket}(v) \wedge \text{connected-to}(z, u) \wedge \\ \text{inside}(u, y) \wedge \text{inserted-in}(w, v) \Rightarrow \text{UNSAFE-TO-TOUCH}(x)] \end{aligned} \quad (13)$$

Comparing rule (13) and rule (12), we notice that there is an overlapping part:

$$\rho \equiv \text{OBJ}(x) \wedge \text{body}(y) \wedge \text{part-of}(y, x) \wedge \text{METAL}(y) \wedge \text{ELT-CABLE}(z, w) \wedge \text{part of}(z, x)$$

which constitutes the justified part of rule (12). Notice that the predicates $\text{closed}(y)$, referring to the body, and $\text{medium-size}(x)$ are not justified by the theory. In order to cut the two unplugged irons, rule (13) suggests a set of predicates to be possibly added to rule (12): $\{\text{resistance}(u), \text{socket}(v), \text{connected-to}(z, u), \text{inside}(u, y), \text{inserted-in}(w, v)\}$. By analyzing the extensions of these predicates, we notice that $\text{inserted-in}(w, v)$ covers exactly the four irons that we want to cover and nothing else. However, by climbing the justification forest, we find the node $\text{PLUGGED-IN}(w)$, which is more general (hence potentially covering more examples) and does not cover any counterexamples, so this last node is chosen to be added. Finally, we obtain the consistent rule:

$$\begin{aligned} \text{OBJ}(x) \wedge \text{body}(y) \wedge \text{part-of}(y, x) \wedge \text{METAL}(y) \wedge \text{ELT-CABLE}(z, w) \wedge \text{part of}(z, x) \wedge \\ \text{PLUGGED-IN}(z) \Rightarrow \text{UNSAFE-TO-TOUCH}(x) \end{aligned} \quad (14)$$

6. Interaction with the expert and theory revision

An interesting aspect of the system WHY is its ability to work in an interactive manner during several phases of its behavior. In order to facilitate this interaction, a friendly, menu-driven user interface has been implemented.

Interaction with the expert occurs at the beginning, when he/she is required to give an initial bulk of phenomenological theory and an abstract causal model. Since this knowledge can be easily revised later, it is not very critical that it is as complete and consistent as possible from the beginning. Owing to the deep integration of the performance element into the learning process, the expert can let WHY run on his/her suggested modifications, analyze the results, and modify the theory accordingly. This is a two-way interaction: the feedback from WHY helps the expert focalize his/her attention on defective parts of the theory, whereas the expert can direct the system to explore the consequences of a subset of the current theory, saving computational efforts.

Human expertise can also be exploited in knowledge refinement, since the expert can directly suggest heuristic rules or parts of them, or can guide the system toward more promising rule specializations or generalizations, telling explicitly which predicates to try to remove or add. For instance, in this case, the expert is presented with the following menu:

REFINE MENU

The expert selects a rule r to be refined. Rule r exists in the knowledge base or can be a hint from the expert.

Current rule: r

Rule syntax: $\varphi \rightarrow h$

Rule assumptions: {set of conjuncts in φ that have been assumed}

Dependency rules: {if φ contains an Ω , a dependency rule may be associated to it, in order to limit induction to a subspace of formulas}

Extension: {set of bindings for each example covered, containing the identifier of the instance, the correct classes, the sets of instance parts satisfying φ , and the number of assumptions made for each binding}

General: {set of rules, more general than r , existing in $\mathcal{KB}$ }

Specific: {set of rules, more specific than r , existing in $\mathcal{KB}$ }

The expert can select among the following choices:

Delete = Disactivate r

Forest = Show the justification forest for r

Save = Save r

Spec = Activate a rule in **Specific**, instead of r

Gen = Activate a rule in **General**, instead of r

Ref-pred = Modify r by refining a predicate {climbing a taxonomy or choosing an operationalization}

Ref-ind = Specialize r by invoking induction

The intervention of the expert during theory revision can be spontaneous or requested by WHY, when some contradiction arises. As an example, let us consider the case of the three hot bulbs, which cannot be justified even by making assumptions. The reason is that there is not mention, in $\mathcal{I}$, of how electric current can flow into the bulb filament. Since we know that the bulbs are positive instances of the concept UNSAFE-TO-TOUCH, and no explanation could be found, we are in presence of incompleteness in the theory. Then, WHY signals the problem and presents the expert with the part of the justification forest instantiated on the bulbs and with the set FAIL of failed causal paths. At the same time, it asks the expert if he/she wants to modify the causal model or the phenomenological theory. According to the expert's answer, an appropriate menu appears (not reported here because its structure matches that of the REFINE MENU).

In the case considered, the expert chooses to revise $\mathcal{P}$, because he knows that the problem is in the incomplete definition of what TERMINALS is. Then, he adds a new operationalization of TERMINALS, stating that there can be an electric contact between a part of the bulb and a part of the supporting lamp. It is interesting to note that a first trial to refine an incomplete explanation for the bulbs by induction succeeded: in fact, WHY covered the three bulbs and no counterexamples by adding to the partial rule (which had been

extracted from the justification forest) the predicate $\text{bright}(x)$, true for all the three objects. However, this modification is doubtful, because of the number of considered examples, whereas the same predicate, added after completing the theory, can be trusted.

7. Conclusions

The system WHY, which learns diagnostic rules from a set of examples (by induction), from a phenomenological theory (by deduction), and from a causal model (by abduction), has been presented. The system can work both automatically and in an interactive mode. The system has been tested on an artificial domain of heat transfer and then has been applied to a real domain of mechanical troubleshooting (Giordana et al., 1993). In the troubleshooting application, the rules learned from only 30 examples (5 for each class) showed, on a test set of 170 examples, performances comparable with those of the rule base learned without causal model (but with domain theory) from 140 examples and tested on 60 examples (Saitta et al., 1991). Moreover, the rules obtained from the causal model (several of which coincided with the previous ones) were perfectly understandable to the expert, in terms of his knowledge of the domain. The suggested conclusion is that a causal model allows robust and meaningful knowledge to be learned with acceptable computational complexity.

References

- Baroglio, C., Botta, M. & Saitta, L. (1992). WHY: A system that learns using causal models and examples. In R. Michalski & G. Tecuci (Eds.), *Machine learning: A multistrategy approach*. Morgan Kaufmann.
- Bergadano, F., Giordana, A. & Saitta, L. (1988). Automated concept acquisition in noisy environment. *IEEE Transactions PAMI*, PAMI-10, 555–575.
- Bergadano, F. & Giordana, A. (1988). A knowledge intensive approach to concept induction. *Proceedings of the Machine Learning Conference* (pp. 305–317). Ann Arbor, MI.
- Bergadano, F., Giordana, A. & Ponsero, S. (1989). Deduction in top-down inductive learning. *Proceedings of the Machine Learning Conference* (pp. 23–25). Ithaca, NY.
- Bergadano, F., Giordana, A. & Saitta, L. (1990). Automated versus manual knowledge acquisition: a comparison in a real domain. *Proceedings of the First Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop* (pp. 301–314). Tokyo, Japan.
- Botta, M. & Saitta, L. (1988). Improving knowledge base system performances by experience. *Proceedings of the EWSL-88* (pp. 15–23). Glasgow, UK.
- Botta, M., Giordana, A. & Saitta, L. (1990). Knowledge base refinement using a causal model. In Z. Ras & M. Zemankova (Eds.), *Intelligent systems: State of the art and future trends*, Ellis-Horwood, Chichester, U.K.
- Cestnik, B. & Bratko, I. (1988). Learning redundant rules in noisy domains. *Proceedings of the ECAI-88* (pp. 348–350). Munich, Germany.
- Chandrasekaran, B. & Mittal, S. (1983). Deep versus compiled knowledge approaches to diagnostic problem-solving. *International Journal of Man-Machine Studies*, 19, 425–436.
- Clark, K. (1978). Negation as failure. In H. Gallaire & J. Minker (Eds.), *Logic and data bases*. Plenum Press, New York, NY.
- Console, L., Torasso, P. & Theseider Dupré, D. (1990). A completion semantics for object-level abduction. *Proceedings of the AAAI Symposium on Automated Abduction* (pp. 72–75). Stanford, CA.
- Cox, P.T. & Pietrzykowski, T. (1987). General diagnosis by abductive inference. *Proceedings of the IEEE Symposium on Logic Programming* (pp. 183–189).
- Danyluk, A. (1991). Gemini: An integration of analytical and empirical learning. *Proceedings of the First International Workshop on Multistrategy Learning* (pp. 191–206). Harpers Ferry, WV.
- Davis, R. (1984). Diagnostic reasoning based on structure and behavior. *Artificial Intelligence*, 24, 347–410.

- DeJong, G. & Mooney, R. (1986). Explanation based learning: An alternative view. *Machine Learning*, 1, 47–80.
- DeJong, G. (1990). Plausible inference vs. abduction. *Proceedings of the AAAI Symposium on Automated Abduction* (pp. 48–51). Stanford, CA.
- de Kleer, J. & Seely Brown, J. (1986). Theories of causal ordering. *Artificial Intelligence*, 29, 33–61.
- De Raedt, L. & Bruynooghe, M. (1991). CLINT: a multistrategy interactive concept learner and theory revision system. *Proceedings of the First International Workshop on Multistrategy Learning* (pp. 175–190). Harpers Ferry, WV.
- Genesereth, M. (1984). The use of design descriptions in automated diagnosis. *Artificial Intelligence*, 24, 411–436.
- Giordana, A., Saitta, L., Bergadano, F., Brancadori, F. & DeMarchi, D. (1993). ENIGMA: A system that learns diagnostic knowledge. *IEEE Trans. on Knowledge and Data Engineering*, 5 (1).
- Hirsh, H. (1988). Reasoning about operationality for explanation-based learning. *Proceedings of the Machine Learning Conference* (pp. 214–220). Ann Arbor, MI.
- Kahn, G. (1984). On when diagnostic systems want to do without causal knowledge. *Advances in Artificial Intelligence*, (pp. 21–30).
- Keller, R. (1988). Defining operationality for EBL. *Artificial Intelligence*, 35, 227–242.
- Kodratoff, Y. (1991). Induction and the organization of knowledge. *Proceedings of the First International Workshop on Multistrategy Learning* (pp. 34–48). Harpers Ferry, WV. (See also this issue.)
- Lebowitz, M. (1986). Integrated learning: Controlling explanation. *Cognitive Science*, 10, 219–240.
- Matwin, S. & Plante, B. (1991). A deductive-inductive method for theory revision. *Proceedings of the International Workshop on Multistrategy Learning* (pp. 160–174). Harpers Ferry, WV. (See also this issue.)
- Michalski, R. (1983). A theory and methodology of inductive learning. *Artificial Intelligence*, 20, 111–161.
- Michalski, R. (1991). Inferential learning theory as a basis for multistrategy task-adaptive learning. *Proceedings of the First International Workshop on Multistrategy Learning* (pp. 3–18). Harpers Ferry, WV.
- Michalski, R.S. & Chilausky, R.L. (1991). Learning by being told and learning from examples: an experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis. *International Journal of Policy Analysis and Information Systems*, 4, 125–126.
- Mitchell, T. (1982). Generalization as search. *Artificial Intelligence*, 18, 203–226.
- Mitchell, T., Keller, R. & Kedar-Cabelli, S. (1986). Explanation based generalization. *Machine Learning*, 1, 47–80.
- Mooney, R.J. & Ourston, D. (1991). A multistrategy approach to theory refinement. *Proceedings of the First International Workshop on Multistrategy Learning* (pp. 115–131). Harpers Ferry, WV.
- Morris, S. & O'Rorke, P. (1990). An approach to theory revision using abduction. *Proceedings of the AAAI Symposium on Automated Abduction* (pp. 33–37). Stanford, CA.
- Pazzani, M.J. (1988). Integrating explanation-based and empirical learning methods in OCCAM. *Proceedings of the European Working Session on Learning* (pp. 147–165). Glasgow, UK.
- Poole, D. (1988). Representing knowledge for logic-based diagnosis. *Proceedings of the International Conference on Fifth Generation Computer Systems* (pp. 1282–1290). Tokyo, Japan.
- Quinlan, R. (1986). Induction of decision trees. *Machine Learning*, 1, 81–106.
- Reiter, R. (1984). A theory of diagnosis from first principles. *Artificial Intelligence*, 32, 57–95.
- Robinson, J.A. & Siebert, E.E. (1982). LOGLISP: An alternative to Prolog. *Machine Intelligence*, 10, 399–419.
- Saitta, L., Botta, M., Ravotto, S. & Sperotto, S.B. (1991). Improving learning by using deep models. *Proceedings of the First International Workshop on Multistrategy Learning* (pp. 131–143). Harpers Ferry, WV.
- Segre, A.M. (1987). On the operationality/generality trade-off in explanation-based learning. *Proceedings of the IJCAI-87* (pp. 242–248). Milan, Italy.
- Tecuci, G. (1991). Learning as understanding the external world. *Proceedings of the First International Workshop on Multistrategy Learning* (pp. 49–64). Harpers Ferry, WV.
- Torasso, P. & Console, L. (1989). *Diagnostic problem solving*. Van Nostrand Reinhold, New York, NY.

Received September 30, 1991

Accepted September 30, 1991

Final Manuscript February 28, 1992.