Machine Learning, 12, 17-47 (1993)
© 1993 Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

An Integrated Framework for Empirical Discovery

BERND NORDHAUSEN BERND@ISS.NUS.SG
Institute of Systems Science, National University of Singapore, Kent Ridge, Singapore 0511

PAT LANGLEY LANGLEY@LEARNING.SCR.SIEMENS.COM
Learning Systems Department, Siemens Corporate Research, 755 College Road East, Princeton, NJ 08540 USA

Abstract. In this article we present a framework that integrates three aspects of empirical discovery—the formation of taxonomies, the generation of qualitative laws, and the detection of numeric relations. We specify a control structure that integrates these component processes, embedding qualitative discovery within taxonomy formation, and embedding numeric discovery within both of these activities. We also describe the framework's basic representation and organization of knowledge, which combines elements from recent work in machine discovery and qualitative physics. In addition, we describe IDS, a running system that instantiates this framework, and report its behavior on problems from the history of science. Finally, we discuss some limitations of the system as revealed by experimental studies, and propose some directions for future research.

Keywords. Empirical discovery, taxonomy formation, qualitative laws, numeric laws, qualitative physics, clustering

1. Introduction

Research on machine discovery aims for computational understanding of the processes that underlie scientific behavior. Naturally enough, previous work in this area has partitioned the complex process of discovery into manageable components. For instance, some researchers have focused on *empirical discovery*, which produces laws that summarize data, whereas others have studied *theory formation*, which generates deeper explanations of those data (Falkenhainer & Rajamoney, 1988; Kulkarni & Simon, 1990; Rose & Langley, 1986). Researchers have further divided empirical discovery into three broad activities:

- *taxonomy formation*, which involves the discovery of conceptual hierarchies (Fisher, 1987; Lebowitz, 1987; Michalski & Stepp, 1983; Nordhausen, 1986);
- *qualitiative discovery*, which concerns finding general qualitative regularities (Emde, Habel, & Rollinger, 1983; Jones, 1986; Langely, Simon, Bradshaw, & Żytkow, 1987); and
- *quantitative discovery*, which involves the induction of numeric laws (Falkenhainer & Michalski, 1986; Langley, Bradshaw, & Simon, 1983; Żytkow, Zhu, & Hussam, 1990).

Typically, research efforts have examined each of these activities in isolation from the rest, and this divide-and-conquer strategy has led to significant improvements in our understanding of scientific activities. As a result, machine discovery has developed into an active and successful subfield of machine learning (Shrager & Langley, 1990; Thagard, 1989).

However, an understanding of science's components is not sufficient; a full theory of scientific discovery should explain the interaction and feedback among these components. An integrated framework should account for aspects of discovery that models of isolated components cannot handle. Our research in this area has been driven by three major questions:

- What *representation and organization* of knowledge will suppport an integrated approach to scientific discovery?
- What *control structure* will support the interactions among the components of an integrated discovery system?
- What *novel capabilities* emerge from an integrated approach to scientific discovery?

Our response to these questions has involved the construction of IDS, an AI system that integrates the three aspects of empirical discovery described above. We cast our tentative answers throughout the article as a series of claims, each supported by tests of IDS on examples from the history of science. Our goal is not to account for the details of historical discoveries, but to show evidence of generality across a broad range of tasks similar to those encountered by scientists in the past.

We have divided the remaining pages into four broad sections. In the first we consider the system's representation of its observations, its background knowledge, and the regularities that it finds. After this we present IDS' overall control structure, including its mechanisms for forming taxonomies, qualitative laws, and numeric relations. We then examine the system's ability to rediscover a number of known scientific laws, comparing its behavior to those of earlier systems on the same tasks. Finally, we consider some limitations of IDS, some directions for future research, and the overall contributions of the work.

2. Representation and organization in IDS

IDS' representation draws upon recent work in qualitative physics. The system describes observations as sequences of descriptions, each expressed as a 'qualitative state' augmented by numeric information, and it uses a similar notation to specifiy its taxonomy and laws. To organize its acquired knowledge, IDS borrows from recent work on incremental approaches to conceptual clustering. In this section, we give the details of the system's representation and organization, first dealing with its inputs and then with its outputs.

2.1. Inputs to IDS

Most discovery systems start with some background knowledge about their domain of operation, whether this is made explicit or not. In IDS, this knowledge takes the form of a simple *is-a* hierarchy that describes classes of objects or substances the system may encounter. For instance, the system might be given knowledge that a specific object `o1` is a member of the class `HCl`, and also that it is `liquid`. Thus, nodes in the is-a hierarchy may have multiple parents. Furthermore, IDS might be told that `HCl` is a member of the more general class `acid`. Other objects and other classes would be connected in a similar fashion; we will refer to the set of such connections as the *background hierarchy.*

However, an empirical discovery system also requires data or observations. IDS is given this information as sequences of qualitatives states, with each state augmented by information about the value of each numeric variable. We will refer to these sequences as *histories*, even though our meaning diverges slightly from that used by Hayes (1979). Each state in

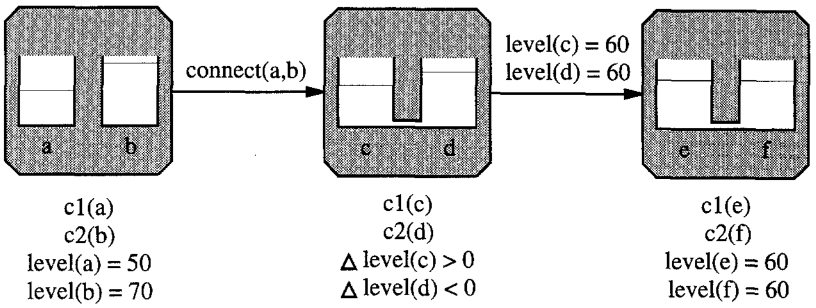

Figure 1. A typical history for a fluid flow experiment.

a history covers an interval of time during which the basic qualitative structure is constant; we will describe what we mean by 'constant' shortly. This representation borrows heavily from work in qualitative physics (Forbus, 1985, Kuipers, 1985; Williams, 1985).

Figure 1 shows the history for a simple fluid flow scenario with three distinct qualitative states. The initial state of this history involves two containers, c1(a) and c2(b), which are not connected. Each container holds a different level of fluid. When these containers are connected by a pipe, the fluid level in c1 begins to rise while the level in c2 begins to drop, producing a second state.[1] During this period, the level of fluid in one container is decreasing, while that of the other is increasing; this is a form of constant behavior, in that the signs of the derivatives remain unchanged. In the final state of the history, the fluid flow has stopped and the fluid levels in both containers remain constant.

IDS represents each qualitative state as a frame with four slots, each of which describes a different aspect of a situation. These include:

- the *object description*, which specifies the objects present in the state using the language of the background (object) hierarchy;
- the *structural description*, which expresses relations that hold among objects during the state;
- the *changes* slot, which contains a list of the changes occurring in the state; as in much qualitative physics work, changes are expressed in terms of derivatives; and
- the *quantity* slot, which describes numerical attributes that remain constant during a state.

For example, the object description for c in Figure 1 is c1(c), and the structural description for the second state is connected(c, d). The changes for this state include a decrease in level of c, which is expressed as Δ level(c) < 0. Finally, the variable level(e) takes on a constant value during the following state. As we will see later, each slot plays a different role in the evaluation function IDS uses to organize states in memory. The system does not store values for numeric terms that vary within a state or information about inequalities. For instance, the description for the second state does not mention the fact that one level is greater than another, although one could infer this from the derivatives and the equality of levels when the state ends.

A transition from one qualitiative state to another occurs whenever any of a state's slots change their content. This includes situations in which an observed variable that was constant suddenly starts to increase or decrease, as well as the reverse situation. A transition between states also results when an object description or the structural description changes, including the creation or destruction of objects. In Figure 1, the first state in the history ends when an external agent connects the containers and the levels of the substances in the containers start to change. Another transition occurs when the levels stop changing, producing the final state. Although IDS is given these boundaries, DeCoste (1991) has recently explored an approach to determining them automatically from a sequence of snapshots.

Histories are inherently temporal in nature, and IDS represents temporal connections between pairs of observed states using *successor* links. These links can be labeled by *transition conditions*, which describe the situation under which the current state ends and its successor begins. A transition condition may involve quantitative descriptions, such as `level(c) = level(d)` or `level(c) = 60`, the actions of an external agent, such as `connect(a, b)`, or both. For instance, a substance having temperature at its melting or boiling point could serve as a transition condition. Such conditions may include features that are irrelevant to the transition, but this will not be apparent from a single history. IDS processes a set of histories incrementally, handling each one it observes in turn. The system also processes the qualitative states in each history one at a time, in chronological order.

Before considering the system's outputs, we should present a partial answer to the first question asked in the introduction.

Claim 1. *Qualitative histories augmented with numeric information provide a useful representation of observations for an integrated discovery system.*

The evidence for this claim will have to wait until later in the article, after we have examined IDS' behavior on some example cases. But it seems worth stating explicitly, so that readers can keep it in mind along the way.

2.2. Outputs of IDS

Because IDS is incremental, the system continues processing states as long as they are available. As a result, it has no explicit outputs in the traditional sense. However, the system does augment its memory after each experience, and in this section we examine the nature of the resulting structures. We will focus on three aspects of this 'output'—the taxonomy, qualitative laws, and numeric laws.

2.2.1. Taxonomy of states

The IDS taxonomy organizes observed qualitative states into a conceptual hierarchy, with input states as terminal nodes and with abstract or generalized states as internal nodes. Thus, this knowledge structure is similar in spirit to those created by CLUSTER/2 (Michalski & Stepp, 1983), UNIMEM (Lebowitz, 1987), and COBWEB (Fisher, 1987). Unlike the background hierarchy, which allows multiple parents, the state taxonomy takes the form of a tree, with no state belonging to more than one abstract category. This state taxonomy makes reference to the background hierarchy of objects, but the two are distinct. Nodes in the state hierarchy include the same slots as specific states (i.e., a description of the structure, the objects involved, the changes occurring during the state, and the constant quantities).

Figure 2 shows the complete state taxonomy after IDS has processed two histories from the fluid flow domain. (The figure also shows the laws found by the system, which we will consider shortly.) These histories contain one class of states (labeled state 1) in which the fluid level of c1 in both children (states 3 and 4) is 50, whereas the initial level of container c2 is 70 in one and 80 in the other. In another class (state 5), c1 and c2 are connected and the fluid level of c1 increases while the level of c2 decreases. In a third abstract state (node 6), the fluid levels of c1 and c2 remain equal. Also, states 5 and 6 have a common abstraction or generalization, which indicates simply that there are two connected objects, c1 and c2.

After observing additional histories in which the initial fluid level of c1 is less than the level of c1, and in which the initial level of c1 is varied, IDS produces a revised taxonomy in which the top-level nodes are more abstract. For instance, the specific containers c1 and c2 are replaced by the general descriptor container, which is the common parent

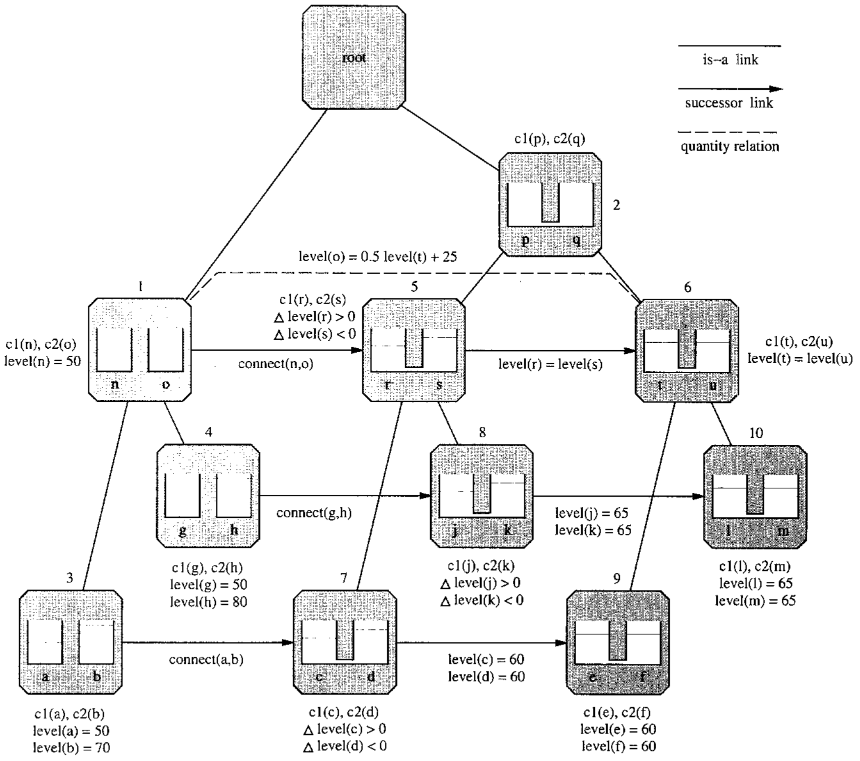

Figure 2. The state taxonomy and laws for fluid flow after six states have been observed.

of `c1` and `c2` in the background object hierarchy. However, some aspects are unmodified, such as the fact that in one abstract state, the level of one container increases while the other decreases. Taken together, these abstract nodes summarize all qualitative states that have occurred in the observed histories.

2.2.2. *Qualitative laws*

In addition to taxonomic knowledge, IDS also represents qualitative laws in terms of relations between pairs of abstract states. Recall that states in observed histories are connected by *successor* links, and these can also be used to express temporal relations between abstract nodes in the state taxonomy. Briefly, such a connection specifies that states in one class are followed by states in another class. Figure 2 uses dashed arrows to indicate such abstract successor links. For example, the link between nodes 1 and 5 specifies that instances of node 5 occur directly after instances of node 1.

The current version of IDS only allows one successor for each node under a given transition condition. Thus, given a pair of two-state histories in which the first states are similar, the transitions are identical, and the last states are dissimilar, the system would form a relatively specific node from the two initial states, with a successor link to a much more abstract node (possibly the root) that summarizes the two following states. This is equivalent to stating that, given an instance of the initial class of states, one can only predict the most general characteristics of the following state. However, given either of the original states, IDS could still predict the states that follow, since it retains the original histories in memory.

Successor links may also specify the conditions under which a transition occurs. For instance, the label on the link between nodes 1 and 5 in Figure 2—`connect(n, o)`—indicates that this transition occurs when the two objects in node 1 are physically connected by an external agent. Taken together, internal nodes and successor links represent qualitative laws similar in content to those found by GLAUBER (Langley et al., 1987). Unlike the earlier system, IDS does not use explicit universal and existential quantifiers; rather, it treats each law as though objects in the preceding state are universally quantified and those in the succeeding state are existentially quantified. Thus, node 1, node 5, and the link between them asserts that when one encounters any situation in which there are two containers and one has a level of 50, if one connects the containers, the initial state will be followed by one in which the levels of the containers move toward each other.[2] The figure also reveals a second qualitative law, stating that any instance of node 5 will be followed by some instance of node 6. After the system has seen additional histories, it formulates analogous laws at a higher level of abstraction.

The embedding of qualitative laws within a taxonomy is one feature that distinguishes IDS from its predecessors, and this leads to a second answer to our question about representations that support integrated discovery.

Claim 2. *Transitions between abstract qualitative states in a taxonomic hierarchy, along with conditions on these transitions, constitute an important class of qualitative laws.*

Again, evaluation of this statement is best delayed until after we have seen examples of the system in operation.

2.2.3. Numeric relations

The fluid flow domain also introduces the third aspect of IDS' knowledge structures—the augmentation of nodes and successor links with numeric laws. The system stores numeric relations in three different places, each of which serves a different purpose:

- a law associated with a single state relates quantities that occur within that state;
- a numeric relation associated with a transition link specifies the numeric conditions for moving from a state to its immediate successor;
- a numeric law associated with a *quantity relation* link between two states relates a quantity in the later state to quantities in the earlier state (sometimes many steps back).

Figure 2 provides examples of all three types of numeric relations. For instance, the equality of levels in node 6 constitutes a simple state equation. In contrast, the successor link between nodes 5 and 6 specifies a transition law or boundary equation, which indicates that the levels of connected containers change until they become equal, after which they are stable. Transition laws may also specify constant values, such as mass reaching zero or temperature reaching the boiling point for a substance. Finally, the quantity relation link between nodes 1 and 6 specifies a dynamic equation that linearly relates the fluid level of container c2 in state 6 and its level in state 1 (i.e., the final and initial levels of c2). IDS stores a more general version of this law after seeing additional histories. This relation, which states that the final level of the containers is half the sum of their initial levels, can be used to predict the final levels.

One aspect of numeric relations that does not appear in the figure is the notion of *intrinsic properties*. Briefly, these are inferred numeric attributes, such as mass or specific heat, which IDS includes in its quantitative laws. Numeric values of an intrinsic property are stored with specific objects or classes of objects in the background hierarchy. This approach lets IDS decompose numeric laws into a general equation, which it stores on a qualitative state, transition link, or quantity relation link, and specific parameters, which it stores with the objects or classes to which the law applies. For instance, IDS represents one version of momentum conservation as the general law $m_1 V_1 + m_2 V_2 = m_1 U_1 + m_2 U_2$, where V and U are the velocities before and after a collision. This law is stored on a quantity relation link connecting two states at the top of the taxonomy, whereas values for the intrinsic property m (mass) are stored on the property lists of specific objects that it has observed colliding.

The numeric laws generated by IDS are similar to those found by BACON (Langley, Bradshaw, & Simon, 1983), ABACUS (Falkenhainer & Michalski, 1986), and FAHRENHEIT (Żytkow, 1986; Żytkow et al., 1990), but there is an important difference, which suggests a third response to our question about representation.

Claim 3. *Numeric relations can usefully augment a taxonomy of qualitative states and its associated qualitative laws, which in turn provide a physical context for the numeric relations.*

Earlier numeric discovery systems found quantitative laws and conditions on them, but their statement of the laws contained little information about the structure or physical situation in which they occurred. The IDS taxonomy of states and successor links specify such a physical context.

3. The control structure of IDS

The IDS system can be characterized as using an incremental hill-climbing approach (Gennari, Langley, & Fisher, 1989). Such systems process one experience at a time, selecting one among alternative ways to incorporate each experience into memory and moving on from there. Furthermore, they hold only one structure in memory, do not reprocess substantial numbers of previous experiences, and retain no information about their learning steps, so explicit backtracking cannot occur. This makes them efficient in terms of both processing and memory. Below we present the top-level IDS control structure, and then examine each component algorithm in detail.

3.1. The top-level algorithm

Table 1 presents pseudocode for the overall IDS algorithm. The system processes one qualitative state at a time, invoking the subroutine modify-taxonomy to sort the state through its current taxonomy, incorporate it as a terminal node in the hierarchy, and create new generalized nodes as needed. This routine returns the resulting parent node of the new state, which may be either an existing node in the taxonomy or a newly created one. Because the initial state has no predecessors, the system treats it as a special case; we will focus on its treatment of the remaining states.

Table 1. The top-level control structure for the IDS algorithm.

```
Input: The root node N of the state hierarchy.
       An observed history H of states.
Results: A hierarchy of states, augmented with
           qualitative and quantitative laws.
Variables: I is an observed state.
           P, Q, and R are nodes in the hierarchy.
           S is a successor link between nodes.

IDS(N, H)
   Let I be the first state in history H.
   Let H be the remaining states in H.
   Let P be (the node generated by) Modify-taxonomy (N, I).
   Find-numeric-within(P)
   While H is not empty,
      Let I be the next state in H.
      Let H be the remaining states in H.
      Let Q be (the node generated by) Modify-taxonomy(N, I).
      Let R be the closest common ancestor of the
          successors of node P's children.
      Find-qualitative-law(P, R).
      Let S be the successor link between P and R.
      Find-numeric-within(Q).
      Find-numeric-transition(S).
      Find-numeric-across(Q).
      Let P be the node Q.
```

After classifying a new state I and determining its parent Q, IDS returns its attention to Q's predecessor P. The system then invokes its second major subroutine, find-qualitative-law, which first determines whether P already has a successor. If so, it checks to see if the new child conforms to the qualitative law stored on the successor link. In contrast, if the parent P is a new abstract node, it does not yet have a successor; thus, the system must determine one, along with an associated qualitative law. As described more fully in Section 3.3, this involves finding the closest common ancestor R of P's children (which may or may not be Q), establishing a new successor link, and storing on it the transition conditions common to the links connecting R and P's children.

Once the system has determined the qualitative law involving the abstract node Q, it invokes three related routines that uncover numeric relations. First IDS calls find-numeric-within, which searches for quantitative relations that hold with the node Q's children. Next it uses find-numeric-transition to find laws that summarize numeric aspects of the transitions between the children of P and Q. Finally, it calls on the subroutine find-numeric-across to discover numeric laws that relate attributes in Q's children to ones in the children of its predecessors.

As we describe in Section 3.4, the function find-numeric-law is used by all three of these algorithms, but with different arguments and once for each numeric term occurring in the state or link. As in the qualitative case, if Q already existed before observing I, IDS checks to see if I obeys the laws stored with Q. If the observed values diverge sufficiently from an existing law, the system attempts to formulate an improved law. If Q has been newly created, IDS tries to induce an initial numeric law that summarizes its children. In both cases, it also attempts to assign values to intrinsic properties of objects involved in nodes and successor links, as we discuss in Section 3.5.

Now that we have summarized the overall IDS algorithm, we can propose our answer to the second question asked in the introduction.

Claim 4. *A useful control structure for integrating empirical discovery embeds the search for numeric relations within the search for qualitative laws, and embeds both of these within the process of taxonomy formation.*

We believe this framework reflects a common path in the history of science. For instance, the early chemists first grouped substances into classes such as acids, alkalis, and salts, which involves the formation of a taxonomy. After this, they began to note qualitative regularities, such as the claim that acids react with alkalis to form salts. Finally, after chemists had identified the basic classes and laws of qualitative structure, they started to detect numeric relations, such as the law of definite proportions. This suggests an overall control structure like that given in Table 1, although it reflects the behavior of an entire community rather than that of an individual scientist. We will not argue that this organization is the only one that supports an integrated approach to discovery, and the history of science includes cases in which feedback runs along other paths. However, we believe the current structure provides a solid foundation on which to build more sophisticated schemes.

3.2. Forming a taxonomic hierarchy

Now let us turn to the components of IDS, starting with the algorithm for clustering or taxonomy formation. Table 2 summarizes the modify-taxonomy algorithm, which has been heavily influenced by Lebowitz's (1987) work on UNIMEM and Fisher's (1987) work on COBWEB. Briefly, upon receiving a new qualitative state, IDS sorts this state through its hierarchy. Starting at the root node, the system computes the similarity between the new state and each child of the current node, measuring similarity with a *lexicographic evaluation function* (Forgy, 1979; Michalski & Stepp, 1983). Because each state may involve multiple objects, the matcher generates all possible bindings between the variables in the two states being compared. As we discuss below, the matcher then calculates an overall similarity score between the two states for each set of bindings and returns the one with the best score.

The system finds the child P that matches the state most closely. If their similarity score exceeds a user-specified threshold, then IDS checks whether P is strictly more general than the observed state. If so, modify-taxonomy is called recursively with P and the state as arguments, so as to pass the state further down the existing hierarchy. If the node P is no more general than the observed state, the system calls on the subroutine Merge, which attempts to generalize the state and P.

If the similarity score fails to exceed the threshold, IDS determines the child Q with the second highest similarity to the observed state. If P is more similar to Q than either is to the state, then it would be inappropriate to combine the state with either node. Instead,

Table 2. The IDS algorithm for taxonomy formation.

```
Variables: N, P, and Q are nodes in the hierarchy.
           I is an observed state (a very specific node).
           X and Y are similarity scores between pairs of nodes.

Modify-taxonomy(N, I)
    For each child C of N,
        Compute the similarity score between C and I.
    Let P be the node with the highest similarity score.
    Let X be the similarity score between I and P.
    If X is greater than a user-specified parameter,
        Then if P is more general than I,
                Then return Modify-taxonomy(P, I).
                Else store I as a child of N.
                     Return Merge(P, I).
        Else let Q be the node with the second highest score.
             Let Y be the similarity between P and Q.
             Store I as a child of N.
             If X is greater than Y,
                Then return Merge(P, I).
                Else return Merge (P, Q).

Note: The object hierarchy is used to generalize and merge states.
```

the system attempts to generalize Q and P using the Merge routine, with the state being stored as a new child of the current parent node.[3] If the best-matching node P is more similar to the state than to Q, IDS falls back on the scheme of merging P with the state.

The first step in merging two nodes, A and B, involves computing a generalization G based on their common structure. In constructing this generalization, IDS uses its matcher to find a correspondence between the variables in A and B, which it then uses to fill the slots of the generalized node. In general, each slot in the new node G is the intersection of that slot's values in the original nodes. However, in determining the generalized object description, IDS collects the predicates that modify each mapped pair of objects and then determines all closest common ancestors—along all paths in the object hierarchy—of each pair. These common ancestors become the predicates in the object description in the new node. If G is more specific than the node P, the system removes A and B as children of P, stores them as children of G, and stores G as a new child of P; otherwise the taxonomy remains unchanged.

The evaluation function incorporates four measures, each corresponding to one of the four slots in state descriptions. For each competitor, it computes a weighted sum based on separate similarity scores, giving a weight of 1000 to the structural description, 100 to the description of changes, ten to the object descriptions, and one to the description of numeric attributes. If the competitors tie on all measures, one is selected at random. The emphasis on structural descriptions and qualitative changes in taxonomy formation has implications for IDS' other discovery mechanisms. In order to formulate predictive qualitative and numeric laws, the system must first define classes of states that have a common physical context.

In general, the similarity score between the slot values for two states is simply the number of literals they share after substitution. For example, the similarity score for the changes slots between States 7 and 8 in Figure 2 is 2, because the states have two changes in common, $level(x) < 0$ and $level(y) > 0$. However, IDS uses the object hierarchy to determine the similarity score for object descriptions. The system collects the predicates used to describe objects in the two states, and then computes the traversal distances between all corresponding pairs of predicates in the object hierarchy. This similarity score is defined to be the inverse of the sum of all traversal distances.

For example, consider one state in which the sole object is described as liquid(x) and HCl(x), and a second state in which the sole object is described as liquid(y) and $HNO_3(y)$. Since HNO_3 and HCl have the same parent, their traversal distance is one, whereas that between liquid and liquid is zero. The predicates liquid and HNO_3, like liquid and HCl, have no common ancestor; thus, they describe different facts of the object and do not affect the sum. As a result, the similarity score for this description slot is $1/1 = 1$, whereas that for two states described as liquid HCl and liquid KOH is the inverse of the sum of 0 and 4, and hence 1/4.

As we noted above, IDS' clustering component has been influenced by Lebowitz' (1987) UNIMEM and Fisher's (1987) COBWEB, but there are some important differences. For instance, IDS and COBWEB form only disjoint taxonomies, in which each node has a single parent. In contrast, UNIMEM can sort an instance down multiple paths, producing a nondisjoint hierarchy. IDS differs from both earlier systems in that it includes no counts or probabilities on its features; each description in the taxonomy is categorical. Our system is probably most akin to COBWEB, though it uses a different evaluation function and lacks the latter's splitting operator (the inverse of the merge operator). Lenat's (1978) AM also incrementally organizes its concepts into a taxonomic hierarchy. However, this system generates new concepts by 'mutating' the definitions of existing ones and then testing them. This 'exploratory' approach to discovery contrasts with the 'data-driven' approach taken in IDS, COBWEB, and UNIMEM, which generate new concepts in direct response to observations.

3.3. Discovering qualitative laws

Having described the IDS clustering algorithm, we can turn to the method for qualitative discovery embedded within it. We have seen that IDS represents qualitative laws in terms of abstract qualitative states and the successor links connecting them. With the exception of the final state in each history, every node in the taxonomy must have some successor node. This temporal information is given as part of the input and specifies the links for leaves in the state hierarchy. From these data, the system must induce the links between abstract nodes.

Whenever IDS forms a new abstract qualitative state (i.e., a nonterminal node in the hierarchy), it returns attention to the previous node *P* returned by the taxonomy formation module and determines the successor node for *P*. This is a simple process that involves finding the closest common ancestor of the successors to *P*'s children.[4] As an example, consider the taxonomy shown in Figure 2. Node 1 has two children, which are labeled nodes 3 and 4. These nodes have as their successors nodes 7 and 8, respectively. In this case, IDS determines that the closest common ancestor of node 7 and 8 is node 5, and asserts this node as the successor of node 1.

In addition, IDS attempts to attach transition conditions to the new successor link, which may take the form of a quantity relation or an external action, such as heating an object. The system determines these conditions in the same way that it forms merged nodes: by finding the structure common to the two successor links. For example, the transition formed between nodes 1 and 5 is labeled `connect(n, o)`, because this action is stored on the successor link connecting nodes 3 and 7, as well as that connecting nodes 4 and 8. The act of adding this condition is equivalent to inducing a law that states 'if containers `c1` and `c2` are connected, their levels will approach one another.'

If IDS finds that the children's links have only some actions in common, it includes only the shared structure in the abstract link. If it can find no common structure, the system creates the successor link but specifies transition conditions that are completely unconstrained. In other cases, the conditions on the specific links involve numeric relations, such as reaching zero mass or achieving boiling point (Żytkow et al., 1990). In this situation, IDS attempts to find a numeric relation that covers the specific cases, using the algorithm described in the next section.

We are now ready for our first claim about the advantages of the integrated approach taken in the IDS framework.

Claim 5. *The embedding of qualitative discovery within the process of taxonomy formation significantly constrains the search for qualitative laws.*

Note that in detecting qualitative relations, IDS considers only ancestors of the successor link between two observed qualitative states. These ancestors are completely determined by the placement of abstract states in the taxonomy, so the system must only find common parents to determine the placement of abstract transition links. Thus, IDS's discovery of qualitative laws is driven almost entirely by the state hierarchy produced during taxonomy formation.

This approach contrasts sharply with that taken by Langley et al.'s (1987) GLAUBER, even though the qualitative laws found by the two systems have similar content. The earlier system considered many different qualitative laws in turn, rejecting some and accepting others based on their ability to summarize observed qualitative relations. GLAUBER included a simple component for clustering substances based on common features, but this was coupled to the law formation process in a much looser fashion than in the current system. Our qualitative laws are more similar in spirit to the *protohistories* posited by Forbus and Gentner (1986), though their approach embeds qualitative discovery within a process of analogy rather than within taxonomy formation.

3.4. Discovering quantitative laws

The third major component of IDS focuses on finding numeric laws. They can specify the conditions for moving from one state to another, a relation between numeric attributes within a given qualitative state, or a quantitative relation between variables in different states. Each of these cases involves storing a law at a node or link in the taxonomy that summarizes information in the children of that node or link. Like Żytkow et al.'s (1990) FAHRENHEIT system, IDS uses a single procedure to find all forms of numeric laws.

For a given data set, the system attempts to find a law that covers these data by conducting a beam search through the space of numeric terms. More precisely, the search task can be stated as:

- *Given*: a set of base terms $a, b, c, \ldots$, along with one designated term (a) from that set;
- *Find*: a term $x = a^{n_0} \cdot b^{n_1} \cdot c^{n_2} \ldots$ such that a linear relation of the form $a = mx + n$ holds.

IDS searches from simple terms to more complex ones, using *correlation analysis* (Freund & Walpole, 1980) to direct the search process. As in Langley et al.'s (1983) BACON, the basic operators involve defining new terms as products and ratios of existing terms. The system initially assumes that a designated term has a constant value. If this does not summarize the data, it examines correlations between the designated term and observable attributes, in hopes of finding a simple linear relation. If this does not produce an acceptable law, it uses the correlations to select promising products and ratios, and then recursively searches for more complex regularities.

Table 3 gives the basic algorithm for finding numeric relations. The top-level function, `find-numeric-law`, is given three arguments: the designated term `D` (which IDS hopes to predict), the set of base terms `S` (the attributes to be used in prediction), and a set of current terms `C` (arithmetic combinations of the base terms). If IDS is attempting to revise

Table 3. The IDS algorithm for finding numeric laws.

```
Variables: S is the set of base terms; D is the designated term;
           A is a defined term; C is the set of current terms;
           P and Q are sets of terms.

Find-numeric-law(D, S, C)
    Let A be the term in the current set C that has the
        highest correlation with the designated term D.
    If the correlation between D and A is above threshold T,
      Then return A, along with the slope and
         intercept of the line that best fits D and A.
      Else if the maximum search depth is reached,
           Then return the empty set.
           Else let C' be Find-best-terms(D, S, C).
                Find-numeric-law(D, S, C').

Find-best-terms(D, S, C)
    Let P be the products of the terms of S and C.
    Let Q be the quotients of the terms of S and C.
    For each term A in the union of P and Q whose
             exponents do not exceed the maximum allowed,
        Compute the correlation between D and A.
    Return the terms with the N highest correlations.

Parameters set by the user:
  Width of the beam N (memory size); Threshold T of the correlation (accuracy);
  Maximum exponents in terms (law complexity); Maximum search depth (when to halt).
```

an existing law, the current set C contains only the single term that occurs in the right-hand side of that law. If the system is searching for an entirely new law, C becomes the set of observable terms S. The system iterates through the set of observable terms, treating each as the designated term D in turn and invoking the algorithm to search for a law that predicts D.

At each point in the search, IDS defines all of the products and ratios between the terms in the set S and those in C, but it retains only those terms having the (typically five) highest correlations with the designated term D. These new terms become the current set C, and the function find-numeric-law is called recursively, with the designated term D and the base terms S remaining the same. If any term in C has a sufficiently high (typically 0.99) correlation with D, IDS ends the search and calculates the slope and intercept of the line that best fits them. The system continues in this fashion until it finds such a linear relation or until it exceeds the user-specified maximum search depth (usually twelve). Also, it abandons any terms that include an exponent larger than a user-given amount (typically five). If the search fails, IDS assumes that no law covers all the observed data. The use of correlation to handle noise contrasts with that taken in Żytkow et al.'s (1990) FAHRENHEIT, which uses an error propagation technique.

The algorithm find-numeric-law is invoked whenever the system adds a new qualitative state (or transition link) to memory. In this case, IDS attempts to find an initial law that covers the children of the new abstraction. The routine is also called whenever the system stores a node S (or link T) as a new child of an existing node (or link) in the hierarchy. In this case, it checks to see if S (or T) obeys the laws currently stored at the node. If not, IDS searches for new laws that cover the new child and its siblings, using the old law as a starting point. This method does require that one store and reprocess all the data that led to the rejected law. Thus, the routine does not quite fit with our definition of an incremental learning system, but it typically requires few observations to induce a law and so remains efficient.

Experiments with the numeric discovery algorithm produced encouraging results on the use of correlation analysis as a heuristic to guide the search for complex numeric relations. In particular, we found that a single setting of the correlation threshold works well for a wide range of noise levels, that the method tolerates noise that results from both constant and relative error in measurement instruments, and that the numbers of irrelevant variables and complexity of the function reduce learning rate only slightly. Nordhausen and Langley (1990) describe these experimental studies in detail. However, one should treat these results with caution, because they involved tests of the numeric component in isolation, rather than in the context of the complete system.

We noted earlier that IDS employs the same basic algorithm for finding three types of numeric relations. Recall that find-numeric-law takes as arguments the designated term D, the set of base terms S, and the set of current terms C. The initial settings of these arguments differ for the three forms of laws because they emphasize different quantities.[5] Thus, when IDS attempts to discover a relation involving quantities within a state, the routine find-numeric-within calls find-numeric-law with one of these quantities as the designated term D, and the union of all within-state quantities and transition quantities as the set of base terms S. Find-numeric-within repeats this process for each quantity in the state, possibly finding multiple versions of a law.

Similary, when find-numeric-transition attempts to find relations on a successor link between two states, each quantity mentioned in the transition condition becomes in turn the designated term D, whereas the set of base terms is the union of all within-state quantities for the first state and its transition quantities. This lets the system discover the conditions for moving between qualitative states. Figure 2 showed such a linear relation that the system finds in the fluid flow domain, which states that one moves from the state 5 (in which levels are changing) to state 6 (when levels are equal). Transition laws can also involve simple constants (which the system prefers to linear relations), as in the case of melting and boiling points. Such relations let IDS implicitly specify inequalities, since they state that certain changes continue until the transition conditions are met.

In order to find numeric laws between qualitative states, the routine find-numeric-across uses a forward propagation method that first attempts to relate a quantity in the current state to the quantities in the immediately preceding state. If the subroutine find-numeric-law cannot infer a law between an attribute in a state and one in its immediate predecessor, it looks for a numeric relation involving the state and the predecessor's predecessor, continuing this chain until it finds a relation or it reaches a state with no predecessor.[6] For instance, in the fluid flow domain described earlier, the system finds a numeric relation between the initial and final states, as shown in Figure 2. After additional histories, IDS finds a more abstract law that lets it predict the final levels from the initial ones.

At this point, we make a second statement about the advantages of integration within the IDS framework.

Claim 6. *The embedding of numeric discovery within the processes of qualitative discovery and taxonomy formation directs the search for quantitative laws.*

The numeric component of IDS finds laws similar in form to those produced by BACON (Langley et al., 1983) and ABACUS (Falkenhainer & Michalski, 1986). Moreover, they employ similar methods to control their search for useful numeric terms, using simple correlations to focus attention. These systems differ in the details of their search control, but a much more important difference resides in the manner of their use. IDS passes only certain sets of variables to its numeric routine, and these are constrained by the taxonomy and qualitative laws it has formed.

Specifically, IDS attempts to formulate within-state laws only for variables that occur within qualitative states with the same taxonomic parent, and that have constant values within those states. Similarly, it searches for transition laws only for pairs of states that obey a common qualitative law, considering only variables that change in a constant direction within the first state. Finally, the system aims for between-state laws only for states that occur in the same chain of transitions, focusing on variables with constant values that have not been included in a within-state law. Although the different representations used by these systems make it difficult to quantify the effect on search, it seems clear that IDS' embedding of numeric discovery within its other processes focuses its attention in ways that its predecessors relied on users to provide.

3.5. Inferring intrinsic properties

As we mentioned in Section 2, IDS postulates intrinsic properties such as mass and specific heat to let it formulate more general numeric laws than would otherwise be possible. These properties are stored with objects or classes in the background *is-a* hierarchy. For example, what we call *density* would be stored with classes of objects, such as silver or gold, whereas what we call *mass* would be stored with specific objects. Now let us consider how IDS uses and infers these properties.

Whenever the system searches for a numeric law, it retrieves the values of the intrinsic properties for the objects involved, which lets it use these properties in formulating quantitative laws. Recall that, in IDS' representation, objects are intentionally defined. For example, the system represents the fact that variable `a` in a state describes object `o1` as `o1(a)`, where `o1` is a terminal node in the object hierarchy (and thus treated like a very specific class). If `o1` has a mass of 9.6 grams, then the quantity slot of the state includes `mass(a) = 9.6`.

Hence, quantities are only indirectly related to nodes in the object hierarchy through shared arguments, as in `o1(a)` and `mass(a) = 9.6`. IDS uses this indirect relation to retrieve the values of the intrinsic properties for an object/class. Because variable `a` describes object `o1`, the system retrieves the intrinsic properties for `o1` in the object hierarchy. The object taxonomy supports *property inheritance*, so that children inherit the properties of their ancestors. For example, if `o1` is composed of `silver` and if a value of the intrinsic property *density* is associated with the `silver` class, IDS would retrieve the density value for `silver` and associate it with any instance of `o1` whenever attempting to find a law that covers that object.

The system uses four rules to infer intrinsic properties and to determine their values. Initially, IDS assumes that the value of a quantity is constant for a given object or class of objects. That is, whenever the system encounters a numeric attribute (either observed or defined), it assumes that this quantity is an intrinsic property and stores the value with the object/class in the object hierarchy. If IDS observes disconfirming evidence (i.e., a different value for the same quantity of the object/class), the system retracts the intrinsic property for this object/class.

The first rule for inferring intrinsic properties states that when IDS encounters a quantity for the first time, it infers a new intrinsic property associated with that quantity.[7] The second rule handles cases in which the system encounters a previously observed quantity, and in which the value of the associated intrinsic property for the observed object is not known. In this case, IDS stores (in the object hierarchy) the value of the observed quantity with the object as its value for the intrinsic property.

The third rule states that, if the system encounters a previously observed quantity, and if the value for the associated intrinsic property can be retrieved for the object/class, then the system should check whether the retrieved value is consistent with the observed value. If the values correspond, then no action is taken. However, if the retrieved value is not consistent with the observed value (i.e., the quantity is not constant for the object), then IDS retracts the quantity as an intrinsic property for the object/class. The system applies this in turn to all of the object's parents in the object hierarchy, thus finding the highest level at which it should retain each intrinsic property.

The last rule states that, if the value of an associated intrinsic property cannot be determined because the relation to the established measurement scale is not known, then IDS should infer no intrinsic values for the observed objects at that time. Taken together, these heuristics let IDS incrementally update the stored values of intrinsic properties, moving them up the object hierarchy when appropriate and removing properties when it finds evidence of overgeneralization. The overall strategy is similar to that in BACON, but more robust due to use of the object hierarchy. We will see the results of this process in the following section.

4. Illustrative examples of integrated discovery

We have already made some specific claims about the advantages of an integrated approach to empirical discovery over methods that address only the components of this process. In this section we examine IDS' behavior on more examples from the history of physics and chemistry, as further evidence of these claims and as motivation for some additional ones. We focus on three idealized historical cases: the reactive behavior of acids and alkalis; the law of displacement; and laws of heating and melting. In each case, we close with an argument for the advantages of an integrated approach.

For the runs described in this section, IDS was presented data in a systematic order, as if generated by an experimenter. This contrasts with 'observational' data, in which the data and their order are beyond the control of the scientist. That is, the system was presented with a succession of histories, with each history representing one experimental condition. The conditions for the initial state in an experiment were varied one at a time while others were held constant, producing a complete factorial design. For each of the runs, we specify the initial conditions and the order in which they were changed.

4.1. The reactive behavior of acids and alkalis

One early chemical discovery involved the reactive behavior of acids and alkalis. Figure 3 shows the history for a simple chemical reaction with three distinct qualitative states. The initial state contains two separate objects, liquid HCl and liquid HaOH. When these substances are combined by an external agent, a new state begins that contains three objects—the two original reactants and a new product, NaCl. During this state, the masses of the reactants are decreasing while the mass of product is increasing. This continues until all the HCl combines, at which point the reaction halts.[8]

Given histories of this form, IDS can rediscover regularities similar to those found by the early chemists. The experiment starts with the history from Figure 3, after which the system is given histories with different masses for the acid and then for the alkali. In this run, the acid is always the *limiting reagent*; that is, the acid is always used up before the alkali. After several experimental conditions using different masses, the substances of the objects are varied, producing some in which no reaction occurs. For example, if a sample of liquid HCl is combined with liquid H_2SO_4, no reaction occurs because both are acids. Background knowledge includes information about the classes of substances (e.g., HCl is an acid, NaOH is an alkali, and NaCl is a salt).

Figure 4 shows the top levels of the state hierarchy after IDS has processed all the histories. States 6, 2, and 7, together with the temporal links among them, represent an empirical law that liquid acids react with liquid alkalis to form liquid salt, similar to that proposed centuries ago. In discovering this law, IDS first notes that liquid HCl reacts with liquid NaOH to form NaCl. After encountering reactions of liquid HCl and liquid KOH, the system forms a more general version of this law, which states that all instances of liquid HCl react with all instances of liquid alkalis to form some instance of salt. Eventually, IDS observes instances of HNO_2 reacting with alkalis and induces the more abstract qualitative relation shown in the figure. The system also formulates the law that alkalis do not react with other alkalis, and arrives at an analogous law for acids in a similar manner.

In addition to these qualitative laws, IDS discovers the law of combining weights for each reaction. This law specifies that, for a particular reaction, the mass of the reaction's product is always proportional to the mass of the reactants. Let us review how the system

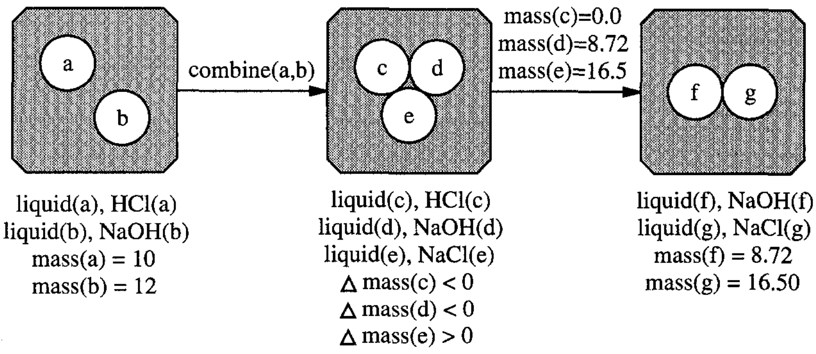

Figure 3. A typical history from the domain of acid-alkali reactions.

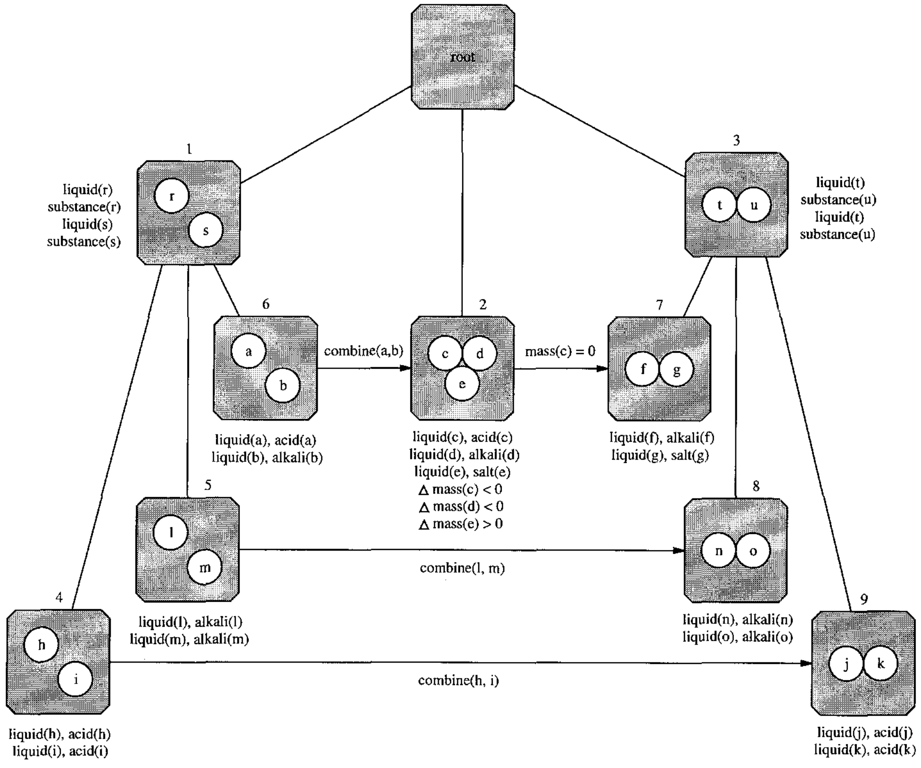

Figure 4. The top levels of the state hierarchy for the reactive behavior of acids and alkalis.

discovers the law of combining weights for the reaction HCl + NaOH $\rightarrow$ NaCl (+ H_2O). At first, IDS observes histories of reactions involving 4.0 grams of liquid HCl and various amounts of liquid NaOH, where HCl is the limiting agent. The system discovers that the reaction produces 6.60 grams of NaCl regardless of the initial amount of the NaOH. Furthermore, the numeric discovery component finds a linear relationship between the final and initial mass of the NaOH. The slope of this relation is 1.0 and the intercept is 4.51. This is equivalent to stating that if 6.60 grams of NaCl are produced, 4.51 grams of NaOH are consumed regardless of the initial mass of NaOH. The system stores both numeric laws on a quantity relation link between the two states.

Next, IDS processes histories of this experiment in which the initial amount of HCl is held constant at 5.00 grams. When the system incorporates these states into the taxonomy, it finds that these reactions produce 8.25 grams of NaCl regardless of the initial amount of NaOH. Furthermore, IDS observes a linear relationship between the initial and final mass of the NaOH, as before. Although the slope of this relation is again 1.0, the intercept this time is 5.64, stating that 5.64 grams of NaOH are consumed in the reaction. When the system encounters reactions in which the initial amount of HCl is 6.00 grams, it finds that 9.90 grams of NaCl are produced and that 6.77 grams of NaOH are consumed. Thus, the slope of the linear relation between the final and initial amounts of NaOH is always 1.0, whereas the intercept varies. The numeric discovery component infers that this intercept is 1.77 times the mass of NaCl. Because the intercept is equivalent to the amount of

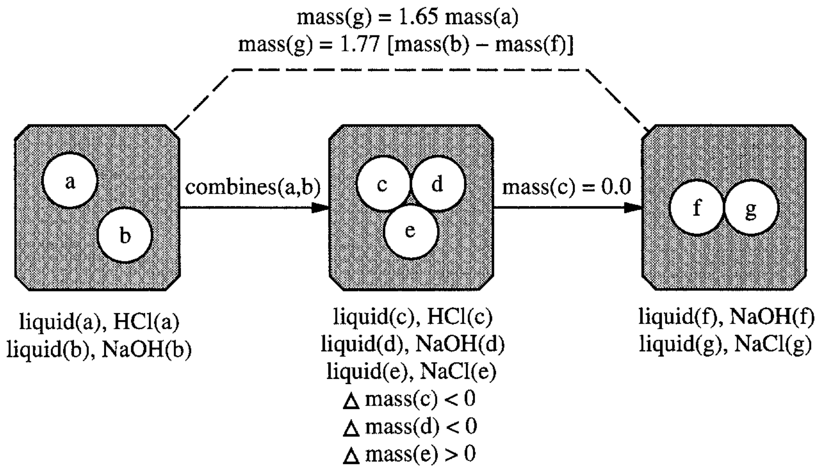

Figure 5. Laws summarizing the reactive behavior of HCl and NaOH.

NaOH consumed, we see this relation corresponds to the law that the mass of NaCl in the HCl + NaOH → NaCl reaction is 1.77 times the consumed mass of NaOH. In addition, IDS finds that the mass of the produced NaCl is always 1.65 times the mass of the HCl.

Figure 5 shows the abstract states from the state hierarchy that summarize the class of HCl and NaOH reactions, including the two associated laws of combining weights. Since they refer to a specific pair of substances, these laws occur at the middle levels of the hierarchy, below the more abstract qualitative relation about acids and alkalis. This points out another advantage of the integrated approach we have taken.

Claim 7. *The IDS framework can augment qualitative laws with numeric ones when they are present, but it can find useful qualitative relations even in domains or at levels where it can find no numeric laws.*

In this case, the system can only find quantitative regularities with respect to specific pairs of substances; the intrinsic properties it postulates do not support numeric relations of higher generality. However, this does not keep it from formulating more general *qualitative* laws that hold for the entire classes of acids and alkalis. Upon encountering a reaction between a new pair of substances from these classes, IDS would still be able to predict the qualitative result. The system takes a similar fallback position with respect to taxonomies, which it can form (and use for prediction) even in domains where qualitative laws are absent.

This capability contrasts sharply with earlier work on empirical discovery. For instance, Langley et al.'s (1987) GLAUBER was able to formulate qualitative laws about reactions between acids and alkalis, but it was unable to induce numeric relations to augment these laws. Similarly, Langley et al.'s (1983) BACON could find numeric laws of combining weights, but it could not even represent failed reactions, much less form qualitative laws describing them. Falkenhainer and Michalski's (1986) ABACUS fares somewhat better on this dimension, in that it established qualitative conditions on its numeric laws, but it could not summarize the qualitative changes that occur over time in many physical situations. Żytkow et al.'s (1990) FAHRENHEIT comes closer to IDS with its ability to identify quantitative 'boundary laws' on numeric relations, but the flavor of these laws differs from those produced by the current system.

4.2. Archimedes' principle of displacement

After the famous bathtub incident, Archimedes formulated his principle of displacement: the volume of a body immersed in fluid equals the volume of the liquid it displaces. Using this principle, the scientist was able to measure the volume of irregularly shaped objects, and thus determine their density and composition. This volumetric attribute can be viewed as an intrinsic property for which different irregular objects have different values. Once these values have been determined, they can be used to distinguish objects from one another.

Figure 6 shows a representative history of the experiment that IDS observes in rediscovering this principle. An irregularly shaped object is placed into a container filled with water, and the weight of the object and the level of water in the container are measured. The figure represents this event as a three-state history. Different experimental conditions vary the water levels and the type of substance. The object hierarchy used in this domain includes knowledge about each object's composition (e.g., o1 is silver, o2 is gold).

When the system processes the first state of the history in Figure 6, it infers two intrinsic properties, ip10 and ip11, based on the weight and the fluid level, respectively. As we discussed in Section 3.5, IDS establishes a relation between the variables in the state and the nodes of the object hierarchy. Because variable a is described as o1, the system assigns 55.0 as the value of ip10 for o1, and it asserts 10.0 as the value of ip11 for the container c1 in the same way. When the IDS processes the third state of this history, it encounters a new value for the water level. The system checks to determine whether the stored values of the intrinsic properties ip10 and ip11 are consistent with the observed values. The weight of o1 remains constant, so IDS does not retract ip10 as an intrinsic property for o1 or c1. However, the new value for the water level is not consistent with the retrieved value of ip11 for c1, so the system retracts that intrinsic property for c1.

Under the next experimental conditions the initial water level is varied, and the numeric discovery component finds that the final and initial water levels are linearly related. The

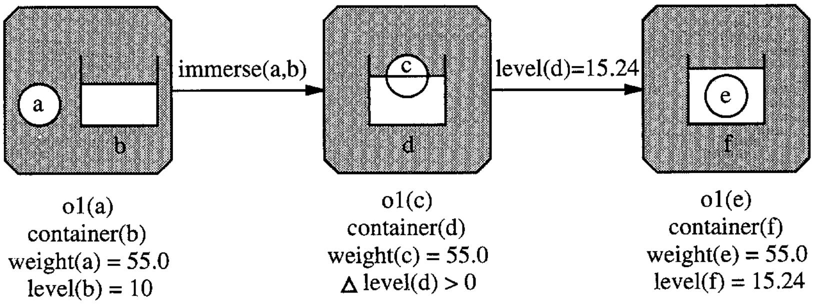

Figure 6. A history of submerging an object in a container containing water.

slope, s, of this relation is 1.0, and the intercept, i, is 5.24. That is, whenever object o1 is immersed in the container, the water level increases by 5.24. The intercept becomes a new numeric attribute,[9] and IDS infers a new intrinsic property, called ip26, associated with i. The system stores values of 5.24 for this intrinsic property both with o1 and with c1.

Next, IDS observes histories in which a different silver object (o2) is submerged in water. The system finds that the water level again increases by a constant amount, but that the value of the intercept i is now 0.95. IDS has already associated the intrinsic property ip26 with i, so it infers 0.95 as the value of ip26 for o2. However, the value of 0.95 is not consistent with the stored value of ip26 for the container c1, so the system concludes that ip26 is not an intrinsic property of c1. When IDS observes experiments in which silver object o3 is submerged, it determines a value of ip26 for o3.

Up to this point, all objects encountered have been composed of silver, so the top-level nodes in the state hierarchy describe the object immersed in the water as a silver object. The numeric discovery component determines that the weight of the object is directly proportional to the value of ip26 for the object (i.e., its volume) with a coefficient of 10.5. The system infers a new intrinsic property (ip39) associated with this coefficient, and stores 10.5 as a value of ip39 for both silver and the container c1. When the system encounters histories in which the immersed objects are made out of gold and lead, it observes new values for this intrinsic property. As a result, it retracts ip39 as an intrinsic property of c1, but infers values of 19.3 and 11.3 for gold and lead, respectively.

This example suggests another conclusion that we can draw about the advantages of integration in machine discovery.

Claim 8. *Augmenting numeric discovery with an object hierarchy simplifies the generalization of intrinsic properties.*

In this case, IDS finds that the values of one intrinsic property (volume) are associated with specific objects, whereas those of another (density) are associated with entire classes of objects. The storage of intrinsic values on nodes in the object hierarchy provides a succinct representation of numeric laws. Moreover, it links the propagation of intrinsic values to the creation of abstract states, which also takes advantage of background knowledge in the object hierarchy.

The importance of this insight is best seen through a comparison with the notion of intrinsic properties used in BACON. This earlier system associated intrinsic properties with specific variables or attributes, and associated intrinsic values with specific values of those attributes. BACON had no representation of object classes, and although it was able to discover Archimedes' principle of displacement, it could only do so when given carefully crafted data that included two nominal attributes, one for the name of an object (o1, o2, o3), and another for its composition (silver, gold, lead). BACON (and to some extent ABACUS) was able to represent conditionality on intrinsic properties, but only in this awkward fashion.

4.3. Laws of heating and melting

Now let us consider the role of integration in rediscovering the law of specific heat. It was known by the 16th century that when one heats an object, say by placing it over a flame, its temperature will increase. However, it was not until the 1760s that Joseph Black accurately described the amount of change by introducing the notion of *specific heat*, another example of an intrinsic property.

Specifically, Black found that the amount of energy required to increase a substance's temperature depends on the amount of the increase, the mass of the substance, and the type of the substance. In general, equal masses of different substances require different amounts of heat to achieve an equal temperature increase. Specific heat is the property associated with a substance that determines the amount of energy required to produce a unit increase in temperature for a unit of mass. One can infer the value of specific heat for different substances by using a burner at a constant setting to heat different liquids that start at the same temperature. One then measures how long it takes for the liquids to reach various temperatures. Because the energy output of the burner is constant, equal amounts of different substances need different lengths of time to reach the same temperature.

The IDS system can rediscover this form of Black's law, provided we augment its representation of histories in a simple manner. Recall that the basic representation divides events into states, each of which lasts for an interval of time. The augmentation involves adding the duration of each state as a transition condition on the link between two states. That is, the time spent in a state becomes one of the quantities in the transition condition between that state and its successor, because one can use this variable to predict when the state will end.

Figure 7 shows a typical history representing this experiment as given to IDS. A burner at a setting of one Kilojoule per second heats a sample of a liquid substance until it reaches a predetermined temperature. The agent turns the burner off at this point, measuring the time it took for the liquid to reach this temperature. Thus, the transition condition between the second and the third state in each history consists of the action by the external agent (turning off the burner), the final temperature of the object, and the duration of the second state. This experiment varies the type of the sample, along with its mass and final temperature. The initial temperature is held constant at 20° Celsius.

In the first few histories given to the IDS, the substance of the sample is water. The mass of the liquid is held constant at 600 grams, but the final temperature is varied. From the resulting data, the system finds a linear relationship between the temperature and the

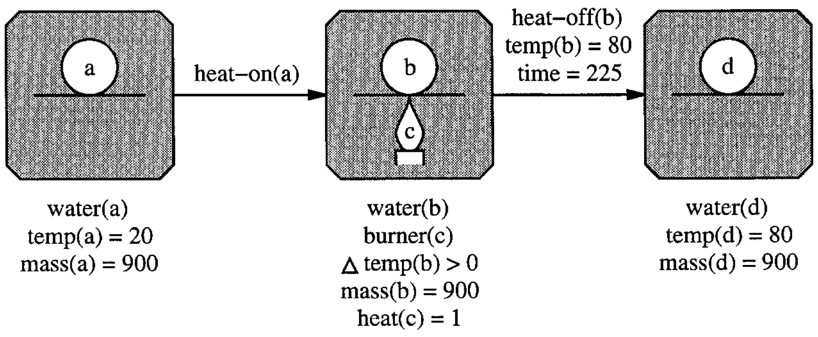

Figure 7. A history for heating a sample of water.

time required to heat the liquid to this temperature. For 600 grams of water, the slope, s, is 0.4 and the intercept, i, is 20.

When the mass of the sample is varied, IDS finds that the intercept, i, of the linear relation between the final temperature and time remains constant at 20. However, the slope, s, of this linear relation changes as the mass of the sample varies. When the mass is 900 grams, s has a value of 0.27, and when the mass is 500 grams, s is 0.48. Based on these values, IDS decides that $s = 239.24 \times 1/mass(b)$; Figure 8 shows the part of the state hierarchy relevant to this discovery, including the quantitative law. Restating these relations, one can deduce that

$$\mathtt{time} = \frac{1}{239.24} \times \mathtt{mass(b)} \times (\mathtt{temp(b)} - 20).$$

In this experiment, the burner has an output of one Kilojoule per second, so one second is equivalent to one Kilojoule of energy. Thus, the value of 239.24 is the reciprocal of the specific heat of water expressed in Kilojoules per gram degree Celsius. When IDS observes experimental conditions in which the sample is HCl and HNO_3, the system finds values of 143.47 and 65.75, respectively. As a result, it infers this coefficient as an intrinsic property and stores values for H_2O, HCl, and HNO_3 in the object hierarchy.

The system can also discover regularities from a more elaborate experiment that starts by heating a solid object until the solid begins to melt and a liquid appears. Heating continues until the solid disappears and the temperature of the liquid starts to increase. When

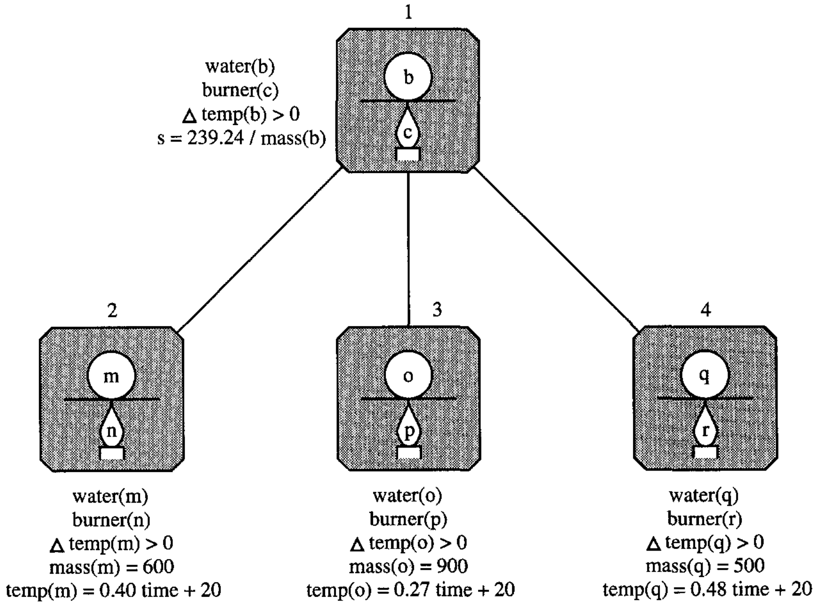

Figure 8. Partial state taxonomy for the discovery of specific heat for water.

the liquid reaches a certain temperature, a gas appears and the evaporation process begins. In the final state, the temperature of the gas increases. When IDS processes histories representing this experiment, it finds that six intrinsic properties are required to summarize the differences among classes of substances:

- the *melting point*, or the temperature at which a substance begins changing from solid to liquid;
- the *boiling point*, or the temperature at which a substance begins changing from liquid to gas;
- the *specific heat of the liquid*, which controls the rate of temperature rise for the liquid form;
- the *specific heat of the gas*, which controls the rate of temperature increase for the gaseous form;
- the *heat of fusion*, which determines the length of time taken for a substance to start melting; and
- the *heat of vaporization*, which determines the length of time taken for a substance to start boiling.

The first two properties revolve around the constant temperatures that IDS observes at particular state transitions. The last four properties emerge from the direct representation of state duration.

This example highlights another point about the integration of qualitative and quantitative discovery mechanisms.

Claim 9. *The explicit representation of temporal structure supports the discovery of an important class of intrinsic properties.*

To be specific, IDS' assumption that observations are represented as a sequence of qualitative states lets it represent and discover a class of intrinsic properties that would otherwise be difficult to handle. Again, this is best seen in comparison with one of the system's predecessors. Langley et al.'s BACON was able to discover a simple form of Black's law, including the property of specific heat. However, this required a carefully crafted representation of the input variables.

4.4. Generality of the approach

One important measure of success for any discovery system is generality, and the examples we have presented give some indication of the breadth of IDS's capabilities. These included the laws of fluid flow, the reaction of acids and alkalis, the principle of displacement, and the laws of heating and melting. In addition, the system has rediscovered a variety of additional empirical regularities:

- *chemical equilibrium*, in which N_2O_4 dissociates into NO_2, which then transforms back into the original substance at a rate that leads to equilibrium. The final concentration of N_2O_4 is proportional to the square of the NO_2 concentration.

- *conservation of momentum*, in which two moving objects collide and thus change their velocities. The unobserved *mass* of each object controls the exact constants of the relation.
- *Black's heat law*, in which the temperatures of two touching objects approach each other until they reach the same temperature. (This version involves more objects than the one in Section 4.3.) The unobserved *specific heat* determines the constant of this relation.
- *rates of chemical reactions*, including zero-order and second-order reactions, in which the rates of reaction are functions of the chemical concentration.

The diversity of these empirical laws, in terms of their taxonomic structure, their qualitative form, their numeric relations, and their intrinsic properties, demonstrates the generality of the IDS control structure and the representation on which it is based.[10]

Although earlier systems could discover many aspects of these regularities, none approaches the full range of empirical relations that IDS can induce from observations. For instance, BACON can generally find equivalent numeric relations and intrinsic properties, but lacks the taxonomic and qualitative knowledge found by the newer algorithm. FAHRENHEIT, and to a lesser extent ABACUS, discover numeric laws and place certain types of conditions on them, but lack the ability to handle temporal and taxonomic structures. GLAUBER formulates qualitative laws and forms simple taxonomies, but lacks explicit knowledge of time and ignores numeric relations entirely. The IDS representation and control structure, combined with its specific component algorithms, gives the system a breadth of coverage that is lacking in earlier work on empirical discovery.

5. Limitations of the approach

In previous sections we described IDS and presented examples of its successful operation. However, it is also important to consider limits as well as successes. In this section, we report some drawbacks of our approach to integrating the discovery process and propose some ways to improve it. The first issue involves problems with the component algorithms used in the current implementation; the remainder concern the general framework and its associated control structure.

5.1. The effects of instance order

Historical examples lend evidence for the relevance and generality of discovery methods, but in the runs described earlier we assumed that data resulted from systematic experimentation and that the system had appropriate background knowledge. To study IDS' dependence on these assumptions, we ran a number of experiments using artificial domains in which we measured the predictive ability of the taxonomies and laws induced by IDS on an independent test set. As Nordhausen (1989) describes in detail, we found that, when given data in which the irrelevant attributes were varied systematically, the system rapidly converged on perfect predictive accuracy. However, when given randomly ordered data, IDS' learning rate was slower and its asymptotic accuracy was lower. Predictive accuracy increased when the data were presented more than once, but it was still far from perfect. Moreover, the system fared even worse when observations were presented in 'experimental' mode but the relevant attributes were varied first.

Additional experiments with multi-state histories showed similar mixed results. In brief, when IDS received the data in a 'good' experimental order, it induced qualitative laws that let it perfectly predict test states from their immediate predecessors, and learning rate decreased only slighlty as the number of qualitative changes in observed states increased. However, given the same training histories in random order, IDS' predictive accuracy was again considerably lower than in the systematic case, and behavior was strongly affected by complexity. Similar results occurred when we varied the number of states in observed histories and the performance task involved predicting the final state from the initial one. These effects result from IDS' complete reliance on a well-formed taxonomy in generating qualitative laws.

Clearly, such behavior is undesirable in a machine discovery system, but we believe it constitutes a limitation of the implementation rather than the framework itself. In future work, we plan to replace the current clustering algorithm with one that includes an operator for splitting abstract states, which Gennari et al. (1989) have shown reduces order effects and which should let IDS recover from overgeneralizations. We also plan to incorporate a more principled evaluation function, such as Gluck and Corter's (1985) *category utility*, which has a theoretical grounding in information theory and which has been successfully used in Fisher's (1987) COBWEB.

5.2. From laws to taxonomies

One characteristic of the current framework is that law discovery is embedded within the process of taxonomy formation. However, the history of science suggests that laws can alter the classification of new observations. For instance, once chemists had determined that acids reacted with alkalis to form salts, they began to classify new substances as acids based on their reactive behavior rather than their taste.

An extended IDS would be able to mimic such behavior if the laws obeyed by a substance or state *S* were added as descriptions of *S*, and if it were allowed to relocate *S* in its taxonomy after it had already been classified. Thus, a state would originally be sorted through the taxonomy based on its isolated description, but it might be moved after obtaining information about its successor state. To retain the incremental flavor of the framework, it is important that such reorganizations remain local and inexpensive.

5.3. Designing experiments

Another important limitation of the IDS framework is that it does not include a component for experimentation, which plays a central role in many sciences. Fortunately, the current knowledge structures provide support for experimental design in a natural way, and we plan to incorporate ideas from recent work in this area. Our thoughts on qualitative experimentation have been influenced by Rajamoney (1990) and Karp (1990), but even more by Kulkarni and Simon's (1990) KEKADA, which includes a heuristic for focusing attention on surprising phenomena. In the IDS framework, one can instantiate this notion as an unpredicted qualitative state or a mispredicted numeric attribute. KEKADA attempts to identify the scope of the phenomenon, generating different initial conditions using a domain theory of substances much like the one in IDS. Moreover, some of Kulkarni and Simon's heuristics, such as dropping factors that have no influence, emerge from IDS' methods for taxonomy and law formation.

Of course, experimentation with numeric attributes is also important. Langley et al.'s BACON incorporated a simple scheme for varying one attribute at a time, but it required the user to specify the values it should consider. Żytkow et al.'s (1990) FAHRENHEIT employs a more sophisticated strategy, which requires less user guidance and which it uses to identify the conditions on numeric laws it has discovered. This approach should interface well with IDS' notion of numeric laws on transition conditions.

5.4. Composability and explanations

Yet another extension would attempt to move beyond empirical discovery to the formation of theories and explanations, by drawing on the successor links in IDS' qualitative laws. For instance, suppose one observes two contiguous qualitative states that are successfully classified, but that have never been observed in immediate succession. Further suppose that one *has* seen cases in which these states are linked through other states. In such a situation, one might infer that these intermediate states actually occurred in the current situation, but for some reason were not observed. Such an inference consitutes an important type of explanation, which would keep the system from adding an inappropriate successor link, as well as let it infer unobserved processes (the intermediate states) and unobserved objects (components of those states).

However, note that even the most general of IDS' laws refer to a specific number of objects. Once it has induced the laws of fluid flow for two containers, the system cannot adapt its knowledge to explain the behavior of three or four containers. In contrast, most actual scientific laws (like most processes in qualitative physics) are *composable*, in that they can be applied repeatedly to handle arbitrarily complex situations. One response would be to pass the induced between-state laws to an algorithm like Żytkow's (1990) GALILEO, which decomposes such regularities into modular components that can be recomposed in many ways.

6. Contributions of the research

At the outset of the article we identified three important components of empirical discovery—the formation of taxonomies, the generation of qualitative laws, and the detection of numeric relations. We argued that, although a divide-and-conquer research strategy had led to progress in machine discovery, it was essential to develop broader theories that integrated these different facets of science. We also posed three questions about the representation, control structure, and advantages of such an integrated framework.

In response to this challenge, we developed such a framework and implemented it in a system called IDS. In describing the framework and the system, we proposed some tentative answers to our questions. In summary, these were:

- Qualitative histories augmented with numeric information provide a useful representation of observations that occur over time.
- Abstract qualitative states, organized in a taxonomic hierarchy and connected by abstract transitions, constitute an important class of qualitative laws and provide a physical context for numeric relations.
- An appropriate control structure embeds qualitative discovery within taxonomy formation, and numeric discovery within both of these processes.
- This embedded control structure provides significant constraints on the search for qualitative and numeric laws, and supports discovery of some regularities even when others are absent.
- The framework supports the discovery of important classes of intrinsic properties and laws, including some that directly involve time.

We gave evidence for these claims in terms of example problems from the history of science, including the reactive behavior of acids and alkalis, Archimedes' displacement principle, and Black's law of specific heat. Although the content of the resulting laws was similar to that found by earlier discovery systems, we argued that IDS showed broader coverage and less need for hand-crafted representations than its predecessors. The current framework explicitly represents and discovers the qualitative structure and conditions on empirical laws, and it represents these regularities at different levels of abstraction.

Our systematic experiments with IDS showed that the system can use the taxonomies and laws it induces to make predictions about unseen attributes and states, and that its predictive ability improves with experience in a domain. However, we also found that the particular algorithm used for taxonomy formation was brittle and, since the results of this process constrain the behavior of IDS' other components, its improvement should receive high priority. We should also test the system on more complex physical domains (such as ones involving oscillation), and determine its ability to organize experiences of many different situations in memory. Nevertheless, we believe the framework itself has many advantages over ones that treat the components of discovery separately, and we think IDS has taken us one step closer to a unified theory. We hope our attempt will encourage other researchers to design and implement integrated theories of discovery, and that together we can progress toward a fuller understanding of the complex process known as science.

Acknowledgements

The research reported in this article was supported by a contract from the Computer Science Division, Office of Naval Research, while the authors were at the University of California, Irvine. We would like to thank members and alumni of the UCI machine learning group—especially Randy Jones, Don Rose, Doug Fisher, and John Gennari—for useful discussions that led to many of the ideas in this paper. We also thank Brian Falkenhainer, Jan Żytkow, Tom Dietterich, and Jeff Shrager for comments on an earlier draft. A preliminary version of this article appeared as a chapter in J. Shrager & P. Langley (Eds.), *Computational Models of Scientific Discovery and Theory Formation.*

Notes

1. Note that the objects are labeled by pattern-match variables that differ from state to state. IDS does not store variable bindings across states, and thus has no explicit notion of object constancy. As Nordhausen (1989) describes more fully, this would limit the system's ability to represent useful abstractions.
2. We will not argue that this approach can represent *all* forms of qualitative laws, but we do feel that relations among abstract, temporally ordered qualitative states constitute an important subset of such laws. Note that our use of 'qualitative' here is broader than in qualitative physics.
3. For an optimal solution, the system would have to consider merging all pairs of children. However, informal experiments suggest that considering all possible pairs seldom improves the resulting taxonomy.
4. This scheme requires that IDS know the successors of *P*'s children; this is the reason the system returns to *P* after it has incorporated a later state into the taxonomy.
5. Within-state and between-state laws incorporate only those attributes having constant values within particular states, whereas transition laws focus on attributes that change in a constant direction with particular states. The histories input to IDS include information about which quantities are constant within each state and which are changing in a constant direction, but one could attempt to identify these automatically.
6. This search process would be expensive in complex domains, but the worst-case cost should increase only as the square of the length of the histories.
7. In some cases, the measurement scale is determined by the particular instrument being used, as when one measures temperature with a thermometer. In other cases, one can only determine relative values of an intrinsic property, as when measuring mass with a balance scale. In this case, IDS assigns an arbitrary value to one object, thus inventing a measurement scale.
8. We have omitted the second product (liquid H_2O) for the sake of simplicity. A more realistic model would include an initial state in which one adds reactants to the water and final states in which the water evaporates before one measures the product.
9. The slope, s, also becomes a new numeric attribute. However, its value is always one, so it does not contain any helpful information. Hence, we will ignore it in our discussion.
10. We should note that all runs described in this section used idealized data; this contrasts with some recent work in machine discovery (e.g., Żytkow et al., 1990), which operates on data from actual experiments.

References

DeCoste, D. (1991). Dynamic across-time measurement interpretation. *Artificial Intelligence, 51,* 273–341.

Emde, W., Habel, C., & Rollinger, C. (1983). The discovery of the equator or concept driven learning. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence* (pp. 455–458). Karlsruhe, West Germany: Morgan Kaufmann.

Falkenhainer, B.C., & Michalski, R.S. (1986). Integrating quantitative and qualitative discovery: The ABACUS system. *Machine Learning, 1,* 367–422.

Falkenhainer, B.C., & Rajamoney, S. (1988). The interdependencies of theory formation, revision, and experimentation. *Proceedings of the Fifth International Conference on Machine Learning* (pp. 353–366). Ann Arbor, MI: Morgan Kaufmann.

Fisher, D. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine Learning, 2,* 139–172.

Forbus, K.D. (1985). Qualitative process theory. In D. G. Bobrow (Ed.), *Qualitative reasoning about physical systems.* Cambridge, MA: MIT Press.

Forbus, K.D., & Gentner, D. (1986). Learning physical domains: Toward a theoretical framework. In R.S. Michalski, J.G. Carbonell, & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach* (Vol. 2). San Mateo, CA: Morgan Kaufmann.

Forgy, C.L. (1979). *OPS4 user's manual* (Technical Report). Pittsburgh, PA: Carnegie Mellon University, Department of Computer Science.

Freund, J.E., & Walpole, R.E. (1980). *Mathematical statistics.* Englewood Cliffs, NJ: Prentice-Hall.

Gennari, J.H., Langley, P., & Fisher, D.H. (1989). Models of incremental concept formation. *Artifical Intelligence, 40,* 11-61.

Gluck, M., & Corter, J. (1985). Information, uncertainty, and the utility of categories. *Proceedings of the Seventh Annual Conference of the Cognitive Science Society* (pp. 283-287). Irvine, CA: Lawrence Erlbaum.

Hayes, P.S. (1979). The naive physics manifesto. In D. Michie (Ed.), *Expert systems in the microelectronic age.* Edinburgh: Edinburgh University Press.

Jones, R. (1986). Generating predictions to aid the scientific discovery process. *Proceedings of the Fifth National Conference on Artificial Intelligence* (pp. 513–522). Philadelphia, PA: Morgan Kaufmann.

Karp, P.D. (1990). Hypothesis formation as design. In J. Shrager & P. Langley (Eds.), *Computational models of scientific discovery and theory formation.* San Mateo, CA: Morgan Kaufmann.

Kuipers, B. (1985). Commonsense reasoning about causality: Deriving behavior from structure. In D.G. Bobrow (Ed.), *Qualitative reasoning about physical systems.* Cambridge, MA: MIT Press.

Kulkarni, D., & Simon, H.A. (1990). Experimentation in machine discovery. In J. Shrager & P. Langley (Eds.), *Computational models of scientific discovery and theory formation.* San Mateo, CA: Morgan Kaufmann.

Langley, P., Bradshaw, G.L., & Simon, H.A. (1983). Rediscovering chemistry with the BACON system. In R.S. Michalski, J.G. Carbonell, & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach.* San Mateo, CA: Morgan Kaufmann.

Langley, P., Simon, H.A., Bradshaw, G.L., & Żytkow, J.M. (1987). *Scientific discovery: Computational explorations of the creative processes.* Cambridge, MA: MIT Press.

Lebowitz, M. (1987). Experiments with incremental concept formation: UNIMEM. *Machine Learning, 2,* 103–138.

Lenat, D.B. (1978). The ubiquity of discovery. *Artificial Intelligence, 9,* 257–285.

Michalski, R.S., & Stepp, R.E. (1983). Learning from observation: Conceptual clustering. In R.S. Michalski, J.G. Carbonell, & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach.* San Mateo, CA: Morgan Kaufmann.

Nordhausen, B. (1986). Conceptual clustering using relational information. *Proceedings of the Fifth National Conference on Artificial Intelligence* (pp. 508–512). Philadelphia, PA: Morgan Kaufmann.

Nordhausen, B. (1989). *A computational framework for empirical discovery.* Doctoral dissertation, Department of Information and Computer Science, University of California, Irvine.

Nordhausen, B., & Langley, P. (1990). A robust approach to numeric discovery. *Proceedings of the Seventh International Conference on Machine Learning* (pp. 411–418). Austin, TX: Morgan Kaufmann.

Rajamoney, S. (1990). A computational approach to theory revision. In J. Shrager & P. Langley (Eds.), *Computational models of scientific discovery and theory formation.* San Mateo, CA: Morgan Kaufmann.

Rose, D., & Langley, P. (1986). Chemical discovery as belief revision. *Machine Learning, 1,* 423–451.

Shrager, J., & Langley, P. (1990). Computational approaches to scientific discovery. In J. Shrager & P. Langley (Eds.), *Computational models of scientific discovery and theory formation.* San Mateo, CA: Morgan Kaufmann.

Thagard, P. (1988). *Computational philosophy of science.* Cambridge, MA: MIT Press.

Williams, B.C. (1985). Qualitative analysis of MOS circuits. In D.G. Bobrow (Ed.), *Qualitative reasoning about physical systems.* Cambridge, MA: MIT Press.

Żytkow, J.M. (1986). Combining many searches in the FAHRENHEIT system. *Proceedings of the Fourth International Workshop on Machine Learning* (pp. 281-287). Irvine, CA: Morgan Kaufmann.

Żytkow, J.M. (1990). Deriving laws through analysis of processes and equations. In J. Shrager & P. Langley (Eds.), *Computational models of scientific discovery and theory formation.* San Mateo, CA: Morgan Kaufmann.

Żytkow, J.M., Zhu, J., & Hussam, A. (1990). Automated discovery in a chemistry laboratory. *Proceedings oj the Eighth National Conference on Artificial Intelligence* (pp. 889-894). Boston, MA: AAAI Press.

Received December 7, 1990
Accepted February 1, 1992
Final Manuscript May 13, 1992