Machine Learning 1: 403–422, 1986

Determining Arguments of Invariant Functional Descriptions

MIECZYSLAW M. KOKAR (KOKAR @ NORTHEASTERN.CSNET)
Department of Industrial Engineering and Information Systems, Northeastern University, Boston, MA 02115, U.S.A.

(Received September 30, 1985)
(Revised August 8, 1986)

Key words: functions, invariance, arguments, discovery, physical laws

Abstract. In this paper we examine the problem of determining arguments of invariant functional descriptions from incomplete observational data. Physical laws are one example of invariant functional descriptions. For such functions, we show that one can test the relevance of the function's arguments even though their values remain constant throughout the observational data. We present a method, called COPER, for discovering invariant functional descriptions. COPER eliminates irrelevant arguments, generates additional relevant arguments, and generates a functional formula. We focus on the first two of these features, giving two examples of how the methodology can be applied to determining arguments of physical laws.

1. Introduction

Determining the relevant attributes (features) of a domain is one of the central problems of machine learning. This task consists of several subproblems: deciding which attributes[1] should be eliminated as irrelevant, deciding which attributes should be added as relevant, and combining the relevant attributes into some useful concept description. This paper presents a methodology for automating this process.

We will assume that the learning system is provided with some observational data represented by series of n-tuples of attribute-value pairs (the values can be numerical or nominal). Such n-tuples are called *events*, and the space of all

[1] We will assume that each attribute has a name along with a set of possible values. In the literature, the same name is sometimes used for both an attribute and its values. Also, the name of the attribute sometimes represents both a variable and the value of that variable, i.e., one element from the value set of the given attribute. The reader should be aware of these terminological ambiguities.

syntactically correct events is called *event space*. A *concept* is a subset of the event space that is described by some predicates. Usually these are conjunctions of selectors, i.e., of formulas represented as

⟨attribute-name # attribute-value⟩

The symbol # stands for a relational operator (cf., Michalski et al., 1983). In an extreme situation a concept can be described by one selector.

Concept learning can occur under several different forms:

- deciding which attributes and values are irrelevant/relevant to the concept
- generating a description for a new relevant attribute
- combining the selectors into a logical formula (such as conjunctions, disjunctions, etc.)

In this paper we focus on the first two categories of concept learning. We assume here that, in the data presented to the learning system, some of the attributes are relevant to the domain, some are irrelevant, and some relevant attributes are missing (not known to the system). The goal of the learning system is to decide which of the arguments are irrelevant (and eliminate those from consideration), to decide whether there are some arguments missing and, if so, to generate the descriptions of the missing arguments.

Any concept learning system must be provided with certain kinds of information. One of these is a language in which the concept descriptions can be expressed. Moreover, the system must have some feedback from the environment that tells which events are instances of the concept and which are not. One source of such information is a teacher who classifies events as examples or nonexamples of the concept. This approach is called 'learning from examples' and has been widely studied within machine learning. In this paper we address the more difficult task of 'learning from observation.' In this paradigm, events may span many different concepts, and the learner itself must assign events to appropriate classes. This is very similar to the task of 'conceptual clustering' described by Michalski (1980). Earlier work in this area has dealt almost exclusively with symbolic concepts, while our concern is with functional descriptions whose arguments, and formulas relating the arguments, fulfill some syntactic requirements.

2. Functional descriptions

A *functional description* maps the values of one or more independent attributes (or *arguments*) onto the values of some dependent attribute (or *argument*). Mathematically, a set of events represents a function (fulfills the condition of functionality) if, for all events having the same value of the independent argu-

ments, the value of the dependent argument is constant. Thus, a functional description gives a rule for predicting the values of the dependent argument from events expressed in terms of the independent arguments. Such a rule constitutes a generalization of the given set of events.

Physical laws constitute a significant class of functional descriptions. For instance, the law of conservation of energy

$$E = M \cdot g \cdot h + M \cdot V^2/2,$$

and Bernoulli's law

$$p1 + \rho \cdot g \cdot h1 + \rho \cdot V1^2/2 = p2 + \rho \cdot g \cdot h2 + \rho \cdot V2^2/2$$

both represent functional descriptions. We will return to the second of these examples later in the paper.

We will say that a set of arguments of the function is *complete*, if any set of events described by this function fulfills the condition of functionality. However, such a set may include irrelevant arguments in addition to relevant ones. We will use the term *necessary* to refer to a complete set of relevant arguments.

With these definitions, we can now state the problem formally:

Given:

- An initial set of independent arguments.
- A dependent argument.
- A set of rules of generating descriptions of independent arguments.
- A set of events expressed in terms of the initial arguments.

Goal:

- Select the relevant arguments and eliminate the irrelevant ones.
- Generate additional relevant arguments. The resulting set must be a necessary set of arguments of the functional description.

2.1 Testing the functionality of descriptions

It seems straightforward to test whether a given set of events satisfies the condition of functionality. To check this, we simply subdivide the events into classes that have the same values of the independent arguments. If the value of the dependent argument is constant for each such class, then it would seem safe to assume that the set of arguments is complete. If not, then the condition of functionality is not fulfilled and some new argument should be generated.

Unfortunately, the situation is not so simple as it first appears. It is quite possible that the function depends on some unknown argument, but that the

values of this argument have remained constant for all observed events. In this case the events will seem to satisfy the functionality condition (and thus the known arguments will appear to be complete, when they are in fact not). Thus we need some more robust method for determining completeness.

The other situation, when the condition of functionality for a given set of events is not fulfilled, is also problematic. Even if we can detect when the set of arguments is incomplete, we still require some means for determining the missing argument(s).

In the following pages, we focus on the first of these problems, describing a method that lets us test the relevance and completeness of a set of arguments when only *some of the arguments* are varied. We then show how one can apply this method to select new arguments to fill out an incomplete set.

Our basic approach relies on the notion of *invariant meaningful functions*; these are functions which are invariant under transformations of the arguments by the generation rules of the description space. For such a function, one can show that knowing all its arguments, and their values, for one point (event) lets one predict the corresponding values for a whole class of events (*orbit*). Note, however, that we know some values of the function from observation. If the predicted values do not agree with the observed ones, then we can infer that the condition of completeness is not fulfilled and some arguments are missing. The important feature of invariant functions is that the predictions can be made without knowing what the functional formula is and, moreover, that to move from one point on the orbit to another, not all arguments have to be changed. This lets one test the relevance of arguments without varying their values. However, this requires knowledge of the generation rules for the description space and their effect on arguments. We consider the nature of the description space and its transformation rules in the following section.

2.2 *The space of descriptions*

The space of descriptions we are interested in consists of *physical quantities* as, for instance, 3 m, 5 kg/m^2, 7 s, etc. This space can be generated out of a few primitives (m, kg, s, and a real number) by two operations — multiplication and raising to a real power (Drobot, 1953; Whitney, 1968). These operations can be applied to any element of the description space. In addition, because real numbers are a subset of the physical quantities, any operation admissible for real numbers can be applied to the elements of this subset (e.g., addition, logarithm). Every physical quantity can be represented in terms of these four primitives as

$$X = \text{number}^{a0}\,\text{m}^{a1}\,\text{kg}^{a2}\,\text{s}^{a3}$$

Where $a0$, $a1$, $a2$, $a3$ are real numbers. For example, if the exponents have the values: $a1 = 1, a2 = 0, a3 = -1$, then the resulting description, $\text{number}^{a0}\,\text{m}^{1}\,\text{kg}^{0}\,\text{s}^{-1}$,

represents the physical quantity of 'velocity.' By assigning a value to the 'number' in the above expression, we obtain one particular value of velocity, such as $3\ m^1s^{-1}$ (usually represented as 3 m/s). By fixing the exponents $a1$, $a2$, $a3$ to the above settings and by varying the exponent $a0$, we can generate a subset in the description space representing the value set of an argument called velocity. In a similar way we can represent any other argument and its value set. Normally, we do not use the exponential representation of the real numbers and write simply: $F = e_f m^1 kg^1 s^{-2}$ to represent an argument of 'force,' where e_f stands for a real number.

In general, any argument (of a physical law) can be characterized by the exponents on the generators (in physics, the generators are called measurement units) and by numbers which distinguish particular argument values from among all the elements of the value set for the argument.

Elements of a description space can be represented not only in terms of the primitive generators but also in terms of several derived descriptors from the same space. For instance, consider the four physical quantities

$$V = 5m^1kg^0s^{-1}$$
$$S = 1m^2kg^0s^0$$
$$F = 8m^1kg^1s^{-2}$$
$$a = 50m^1s^{-2}$$

representing velocity (V), area (S), force (F), acceleration (a), and a number, say 2. Acceleration (a) can be expressed in terms of V and S as:

$$2^1V^2S^{-0.5}F^0$$

However, none of the three descriptors V, S, and F can be expressed in terms of the remaining two. We will say that such a set of descriptors is *syntactically independent*, and we will use the term *base* to refer to any maximal set of syntactically independent descriptors. According to this definition, the above descriptors (V, S, and F) are syntactically independent and constitute a base. Adding any descriptor to this set would make it syntactically dependent.

Just as we can define the notion of a base for descriptors (values of the arguments), we can introduce the concept of *base arguments*. We will say that a set of arguments is syntactically independent if none of these arguments can be expressed solely in terms of the other arguments in the set. Otherwise, the set will be a dependent set of arguments. Any maximal set of independent arguments will be called a base. Thus the arguments V, S, and F constitute a base, while V, S, F, and a do not.

One can show that any descriptor from a given description space can be expressed in terms of any base, and that the representation is unique (in the given base). This means that, if we have a mapping of one base onto another base, then this mapping defines a transformation of the space. This lets one easily transform

arguments and events based on one representation into corresponding arguments and events based on another representation. For a description space representing physical quantities, selecting another base means selecting another system of measurement units, and transforming the space means expressing all physical quantities in another system of units.

Since we are concerned with the automatic generation of bases, it is important to know how many elements a base should contain. Fortunately, it is easy to see that, if a space has n generators, then the number of descriptors (arguments) in a base is also n. In the case of physical quantities, one of these will be a real number, since numbers are common to any description space for physical quantities.

We have barely touched on the problems that arise with transformations of description spaces, but the reader can find more detailed discussion in publications on dimensional analysis (Luce, 1978; Drobot, 1953; Causey, 1969).

2.3 Invariant functional descriptions

In the previous section we described the space of arguments in which we are interested. Now let us turn to the form of the functions involved and discuss the property of invariance of functional descriptions.

In this paper we are concerned with functions that are *invariant*. In short, this means that, if the arguments of the function are changed according to the rules of generation of the description space, then the values are changed according to the same rules. The practical implication is that, if we know that a function is invariant and we know the value of the function for one point, then we can easily calculate its values for many other points. We can compute these values from the starting point using transformations, even without knowing the formula that describes the function.

Luce (1978) has linked the notion of invariance with a more elementary notion of *meaningfulness* of functions. Meaningful functions utilize only the operations which define the decription space. In the case of physical laws, these are multiplication and exponentiation (applicable to all arguments) and other operations, as addition or logarithm, applicable to those arguments which are purely numerical. Under this interpretation, addition of two arguments of the same kind (e.g., velocity to velocity, force to force) is understood as multiplication of one of them by a number. Such an interpretation cannot be applied to addition of velocity to force. Thus it is meaningful to add two forces, but it is not meaningful to add distance to mass, or velocity to force.[2] Moreover, nonnumerical arguments can be combined into monomials that are purely numerical, and then *any* numerical operation can be applied to these transformed arguments. One can transform

[2] We refer the reader to Luce (1971, 1978), Causey (1969), Krantz, Luce, and Suppes (1971), and Kokar (1985) for a fuller discussion of invariance and the meaningfulness of functions.

arguments of a meaningful function into a (smaller) number of numerical arguments simply by dividing a derived argument by its representation in a selected base. For instance, division of acceleration (a) by its representation in the base (V, S)

$$a/(V^2S^{-0.5})$$

results in a purely numerical (dimensionless) value (see section 2.2).

Invariance is a very important property of the meaningful functions. It lets us predict the value of the function for some values of its arguments, provided we know the value of the function for another point related to the second one through the generating transformations. We will rely on this property in our methodology for determining functional descriptions.

2.4 Invariant functions and orbits

Now we are ready to show how the notion of invariance can be used to determine the completeness of a set of arguments.

Suppose a functional description has n arguments. As we found in section 2.2, we can then select m of them as base arguments (provided they are syntactically independent) and express the remaining $r = n - m$ in terms of this base. In doing so, we would use the generating operations of the language, multiplication and exponentiation in the case of physical quantities. For instance, if the base arguments are length (m) and time (s), then the derived argument acceleration (m/s^2) can be expressed in terms of the base descriptors as

$$\text{acceleration} = \text{number} \cdot \text{length} \cdot \text{time}^{-2}$$

By varying the numerical values of the base arguments in the same formula, we can express the same value of derived argument in terms of the base arguments in many ways. We will use the term *orbit* to describe a set of points for which the derived arguments are constant and the base arguments are varied in such a way that they fulfill the equations relating the two sets of arguments.

In the above example, the points (8 m, 2 s, 2 m/s^2), (18 m, 3 s, 2 m/s^2) and (32 m, 4 s, 2 m/s^2) would belong to the same orbit, for all of them the above formula has the value of 2 m/s^2.

Let Z represent a dependent argument, and let $A1, \ldots, Am, B1, \ldots, Br$ represent independent arguments of the functional description F. If we choose $A1, \ldots Am$ as our base arguments, then we can represent the value of $B1, \ldots, Br$ in terms of $A1, \ldots, Am$ values in many ways. Moreover, we can design experiments in such a way that the $B1, \ldots, Br$ values are constant for several combinations of $A1, \ldots, Am$ (points of the same orbit). Therefore, for these observations, Z will depend only on $A1, \ldots, Am$ and we can predict the changes

of Z for each orbit, given the value of the function for one point in each orbit.

This is true provided that $A1, \ldots, Am, B1, \ldots, Br$ are the only arguments of the function. If there exists another relevant argument, Bk, of which we are not aware, then our predictions will not agree with the observations of Z's values. This is because we did not take the argument Bk into consideration when designing the experiments (the orbits), and because the changes in Z caused by changes in $A1, \ldots, Am$ can be predicted only if the arguments of the function are in the same orbit. This is not the case when Bk remains unchanged (or if its changes are beyond our control) and $A1, \ldots, Am$ change their values independently of Bk.

Based on this reasoning, we can introduce a measure of *disagreement*, or *nonconformity*, between the predicted values of a function and the observed ones. We define this measure as the average difference between observed and predicted values. Complete agreement can be achieved only if one takes into account all the relevant arguments of the function in predicting the values of that function. A high value of the nonconformity measure indicates that some of the arguments are missing, and including an additional relevant argument will improve (lower) this measure.

This gives not only a method for testing the completeness of a function's arguments, but also suggests a method for deciding *which* argument should be included as relevant (or excluded as irrelevant). The important point here is that we do not need to alter the values of an argument in order to test its relevance. On the other hand, if the argument is changing without our control, it will affect our predictions of the function's value, since we are not able to stay within an orbit. In such cases, the value of the nonconformity measure will be high. Thus, the nonconformity measure can be utilized as an indicator of incomplete relevance both when the unknown argument is constant *and* when it is changing without one's control.

3. COPER: A methodology for determining invariant functional descriptions

The above reasoning suggests a method for identifying relevant arguments and for discovering descriptions of new arguments, even when the values of these arguments remain constant throughout one's observations. This procedure can be described in the following steps:

1. Determine the base descriptors (generators) and the syntactic rules for generating new descriptors.
2. Determine the initial independent arguments and the dependent argument of the function.
3. Select the base arguments from the set of independent arguments.
4. Express the rest of the arguments in terms of the base arguments (using the rules of generation); these are the derived arguments.

5. Design experiments (orbits) in which the base arguments are changed in such a way that the values of the derived arguments, when expressed in terms of them, will remain constant.[3]
6. Calculate the predicted values of the function for each orbit and compare these values with the observed ones.
7. If the difference between the predictions and observations (nonconformity measure) is significant, then generate a new descriptor to be taken into consideration in the functional description. To this end perform a search of the description space, looking for that descriptor which minimizes the measure of nonconformity. The new descriptor will produce a new partition of the observations into orbits, so repeat steps 5–7.

We call the above method COPER, and we have implemented the algorithm as a running computer program. We have tested the system on its ability to discover physical laws, to which we now turn.

3.1 The law of falling bodies

To better explain the COPER method let us consider its application to rediscovering the arguments of a well-known physical law. The law of falling bodies can be represented by the following function

$$S = V \cdot t + g \cdot t^2/2$$

where S stands for the distance, V for the velocity, t for time, and g for the acceleration of gravity. We will assume that COPER is given a set of events (experimental data), stated as the values of the independent arguments and of the dependent argument. In this example we will consider three different situations:

- When all argument descriptors are known, we will show that the COPER method confirms that the set of arguments is complete.
- When one of the descriptors (the gravity acceleration g) is missing, we will see that the method correctly determines that the set of arguments is incomplete.
- When the system generates a candidate descriptor which is irrelevant, we will see that its nonconformity measure is high; this means that the method can eliminate irrelevant descriptions.

Note that the input data will be generated (from the above equation) in such a

[3] One can apply a similar strategy to nonplanned experimental data by subdividing the observed events into orbits.

way that the argument g will be constant. Consequently, there will be no indication about nonfunctionality of the data.

Let us follow the steps described in the previous section.

Step 1. The discovery system must first identify the primitives (generators) of the description space and the rules for generating derived descriptors (cf., section 2.2). This information is provided by the user. In this case the primitives are m (meter) and s (second), representing length and time respectively. The generating operations for this description space are multiplication by another descriptor and exponentiation (raising to a power). This means that every descriptor can be represented[4] as $X = e^{a0}\mathrm{m}^{a1}\mathrm{s}^{a2}$, or after replacing e^{a0} with a real number e_x as $X = e_x\mathrm{m}^{a1}\mathrm{s}^{a2}$.

Step 2. In the second step, the system must identify the dependent argument and the known independent arguments. Again, the user provides this information. In this example we have the dependent argument S, which can be represented as:

$$S = e_s\mathrm{m}^1\mathrm{s}^0 = e_s\mathrm{m}$$

the values of which are 3 m, 5 m, 6 m, ..., etc. The independent arguments are

$$V = e_v\mathrm{m}^1\mathrm{s}^{-1} = e_v\mathrm{m/s}$$
$$g = e_g\mathrm{m}^1\mathrm{s}^{-2} = e_g\mathrm{m/s}^2$$
$$t = e_t\mathrm{m}^0\mathrm{s}^1 = e_t\mathrm{s}$$

Thus, the function relating these terms has the form:

$$S = F(V, t, g)$$

Step 3. At this point the system must select a base. Since we have two primitive generators (besides a numeral) the base should consist of two arguments, and this means that the system must find such two arguments that are syntactically independent. One can test syntactic independence for physical quantities by computing the determinant of the arguments' exponents. For the arguments V and t, the determinant of the matrix

$$\begin{matrix} 1 & -1 \\ 0 & 1 \end{matrix}$$

is 1, and since it is non-zero, we can conclude that the terms are independent. In

[4] In this example, we have at most two syntactically independent descriptors (not counting numbers), e.g., ($3\ \mathrm{m}^1\mathrm{s}^{-1}$, $5\ \mathrm{m}^3$). The descriptors $5\ \mathrm{m}^1\mathrm{s}^{-1}$ and $6\ \mathrm{m}^{-1}\mathrm{s}^1$ are not syntactically independent because one of them can be expressed in terms of the other, e.g., $3\ \mathrm{m}^1\mathrm{s}^{-1} = 18(6\ \mathrm{m}^{-1}\mathrm{s}^1)^{-1}$.

this example the system has three choices for the base arguments: (V, t), (V, g), and (g, t). All these pairs fulfill the condition of syntactic independence. Suppose the arguments t and V are selected for this purpose.

Step 4. Now the arguments g and S need to be expressed in terms of the chosen base arguments, giving: $g = b_g V^1 t^{-1}$, and $S = b_s V^1 t^1$. Note that in expressing these arguments, we have used only the rules of generation: multiplication and exponentiation. Also, it is important to realize that the formula for S does not represent the functional dependency which is the goal of the discovery process; it is merely a syntactic expression useful for the purposes of prediction.

Step 5. Suppose the values of the arguments are: $V = 6$ m/s, $g = 10$ m/s^2, $t = 2$ s, and in this situation we observe $S = 32$ m. The goal at this point is to design an experiment in which the values of V and t are varied, but in which the value of g, expressed in terms of V and t using rules of generation, remains constant. The first three columns of Table 1 present some values of these arguments. Recall that in the base we have selected, $g = b_g V^1 t^{-1}$, and from values of V and t in the Table we can compute that the value of g is always 10.

Step 6. The next step is to calculate the predicted values of S. This requires that we take into account the changes in the base arguments V and t, using the formula $S = b_s V^1 t^1$ that we identified in step 4. We know the value of S for the first point, and from this we can calculate the value of $b_s = 32/(6 \cdot 2) = 2.67$. The values for other points can be predicted by multiplying b_s by the new values of V and t. These predictions are shown in the fourth column of Table 1.

Step 7. As Table 1 shows, the predicted and observed values of S are identical, telling us that our set of arguments is complete. This shows the strength of the invariance assumption; using this constraint, we can determine the completeness of the argument set even when we have no idea of the function that relates these terms.

In this example, because the predicted values agreed with those obtained by observation, there is no reason to search for a new argument. We can conclude

Table 1. Predicted and observed values for the law of falling bodies.

V(m/s)	t(s)	g(m/s^2)	S-predicted	S-observed
6	2	10	32	32
9	3	10	72	72
12	4	10	128	128
15	5	10	200	200
18	6	10	288	288
21	7	10	392	392

that the behavior of falling bodies can be expressed in terms of the four arguments S, V, t, and g. But can the method also reveal when the one of the arguments is missing?

3.2 Inferring the descriptor of gravity

To analyze the situation when the set of arguments is incomplete, let us take the same example but with different starting information. Assume that we do not know about g, the argument for acceleration due to gravity. Instead, we assume the functional description of falling bodies has the form $S = F(V, t)$.

From a purely mathematical point of view this is very nearly true, because it is very difficult to alter g under typical circumstances (such as on earth). More precisely, let us assume that we can only observe situations in which the acceleration due to gravity, g, is the constant 10 m/s^2 (for simplicity we will use 10 instead of 9.81). Using the method described earlier we will show that the set of arguments composed of V and t is not complete, and that this state of affairs can be detected even when the unknown term g is constant.

However, we will review only steps 5, 6, and 7 of the method in this context. The first four steps remain essentially the same, with the exception that the argument g is not taken into account. This means that in step 3 we have no flexibility in choosing the base; the only base consists of V and t, and these are the only known arguments as well.

Step 5. We are not limited in the design of our experiment because there are no arguments besides the base. Therefore, any values for V and t are acceptable. Obviously, this is a very simple example, and if we were dealing with a more complex physical law we would have to take more arguments into account. The first two columns of Table 2 show some possible values of V and t.

Step 6. We calculate the predicted values in the same way as before. First, we compute the value of b_s from the first experimental point for which the value of S is known; this gives $b_s = 32/(5 \cdot 2) = 3.2$. We then compute the predicted values

Table 2. Predicted and observed values for falling bodies with g unknown.

V(m/s)	t(s)	S-predicted	S-observed
5	2	32	32
10	12	384	840
15	10	480	650
20	8	752	480
25	6	480	330
30	4	384	200

Table 3. Results of an experiment testing the relevance of $X(m^2)$.

V(m/s)	t(s)	$X(m^2)$	S-predicted	S-observed
1	32	1	5152	5152
2	16	1	5152	1312
4	8	1	5152	352
8	4	1	5152	112
16	2	1	5152	52
32	1	1	5152	37

of S, using the definition $S = b_s V^1 t^1$. The last two columns of Table 2 present the results of these predictions and the corresponding observations.

Step 7. As one can see, this time the predicted values significantly differ from the observed ones. This indicates that the complete relevance condition was not fulfilled, and this is because we did not group the experimental points into orbits, having not known about the argument g. Notice that we arrived at this conclusion without changing the value of the missing gravity argument.

Now, let us consider how the COPER method can be used to generate the missing argument. We have already seen that it can recognize g as an acceptable candidate, and we will shortly see that it can eliminate other arguments. These two features make it possible to perform a systematic search through the space of argument descriptions.

Take as an example the candidate argument $X = e_x m^2 s^0$, which represents an argument of type 'area.' In step 4 this argument would be represented in terms of the base as

$$X = b_x V^2 t^2$$

According to step 5 of the method, we design an experiment such that for any orbit, $V^2 \cdot t^2$ remains constant. Table 3 presents the results of such an experiment, including the predicted and observed results for the arguments. Again, we assume here that the value of the function for the first point is known.

As we can see from the Table, the predicted and the observed values do not agree, and this would lead the method to reject X as a candidate argument. However, recall that m and s are generators for the description space, and we can use these to consider systematically the set of possible arguments. COPER uses this approach to generate all such arguments (up to a given level of complexity), and uses the above method to evaluate arguments in turn. Of all such generated arguments, $g = b_g V^1 t^{-1}$ gives the lowest measure of nonconformity, so this term is selected to complete the set of functional arguments.

3.3 *Bernoulli's law of fluid flow*

As our second example, we will consider the more complex functional description known as Bernoulli's law. This function describes the nonviscous, steady, incompressible flow of a fluid through a pipeline. The pipeline has a crosssection $A1$, located at an elevation $h1$ from some level of reference, and a crosssection $A2$ at an elevation $h2$. Let $p1$ be the pressure at $A1$, and $V1$ be the speed of flow at $A1$, $p2$ be the pressure at $A2$, and $V2$ be the speed at $A2$. Using these terms, Bernoulli's law can be expressed as:

$$p1 + \rho \cdot g \cdot h1 - \rho \cdot V1^2/2 = p2 + \rho \cdot g \cdot h2 + \rho \cdot V2^2/2$$

where ρ represents density of the fluid, and g stands for acceleration of gravity.

In this example we will constrain our attention to testing the complete relevance condition and to discovering missing arguments if the complete relevance condition is violated. In particular, we will see how the COPER method can be used to discover the arguments ρ and g, even if the values of these terms never change during the experiment.

The generators of the description space in this example are m (meters), s (seconds), kg (kilograms), and the real numbers. We will treat the variable $p2 = e_{p2}\text{kg}^1\text{m}^{-1}\text{s}^{-2}$, as the dependent argument, and we will assume that the system knows about five independent arguments; these are: $p1 = e_{p1}\text{kg}^1\text{m}^{-1}\text{s}^{-2}$, $h1 = e_{h1}\text{m}^1$, $h2 = e_{h2}\text{m}^1$, $V1 = e_{v1}\text{m}^1\text{s}^{-1}$, and $V2 = e_{v2}\text{m}^1\text{s}^{-1}$. The system must infer the arguments ρ and g on its own initiative, since these are not provided as input.

Thus, at the outset COPER's default assumption was that the function had the form $p2 = F(p1, h1, h2, V1, V2)$, and it set out testing that belief. The first step in this process involved reexpressing the independent arguments and the dependent argument in terms of a set of base arguments. In the actual run COPER selected $p1$, $h1$, and $V1$ as the elements of the base, giving the derived definitions $p2 = b_{p2}p1$, $h2 = b_{h2}h1$, and $V2 = b_{v2}V1$. The system's next action was to generate an experimental plan involving 1024 observations (16 orbits with 64 points in each of the orbits), and in which the values of $h2$ and $V2$ remained constant on each orbit.[5]

Upon comparing the predicted and observed values of the argument $p2$, COPER computed a nonconformity measure of 141.5. On this basis, the system concluded that its set of arguments was incomplete, and it initiated a systematic search of the description space to remedy this problem.[6]

[5] In this example, we provided the system with idealized data based on Bernoulli's law. The actual data would involve a certain amount of noise, and extending the method to handle such noise is an important direction for future research.

[6] COPER can also operate in an interactive mode, accepting candidate arguments from the user and returning an evaluation of their usefulness. However, our focus here is on the automatic generation of arguments.

This search process involved generating candidate arguments using the general form $e_x kg^{a1} m^{a2} s^{a3}$, and calculating the nonconformity score for each candidate. COPER generated different descriptors by varying the exponents $a1$, $a2$, $a3$ in the above formula (in the range between -3 and 3). The best of these descriptors was $e_x kg^1 m^{-3} s^0$, which corresponds to the ρ term in Bernoulli's law. However, introducing this argument did not lower the nonconformity measure to zero, so COPER recursively applied its method to the revised set of data. The lowest scoring of the remaining descriptors was $e_x kg^0 m^1 s^{-2}$, which corresponds to the term for gravity in Bernoulli's law. Taken together, the ρ and g terms let COPER perfectly predict the values of $p2$; this gave a nonconformity measure of zero, and the system halted, having generated a complete set of arguments.

In summary, COPER can be viewed as carrying out a heuristic search through the space of argument sets. The operator for this space involves adding an argument to the existing set (according to the generation rules for the language of descriptions), and the branching factor is quite large ($3^7 = 2187$ for the Bernoulli's law example). This is offset by the nonconformity measure, however, which acts as an evaluation function to select the best state (argument set) at each step in the search. This measure also acts as the termination condition, when its value reaches zero. Using this approach, COPER can discover complete sets of arguments for complex functions like Bernoulli's law.

3.4 Generating a functional formula

This paper is primarily concerned with the COPER's ability to discover arguments of functional descriptions, but this system also can generate functional formulas relating arguments of the functions (cf., Kokar, 1978, 1979, 1986). The method for discovering functional formulas employed in COPER is based on three main principles: the meaningfulness of functions, the Weierstrass theorem, and the ability to change the representation base.

In section 2.3 we stated that meaningful functions are those which can be expressed solely in terms of the defining operations of the description space. In the case of physical laws these are multiplication and exponentiation (applicable to any argument), and any numerical operation (applicable to numerical arguments only). One can transform all derived arguments into numerical arguments by dividing them by the respective formulas representing the derived arguments in a base. In the theory of dimensional analysis, the Pi theorem states that any invariant function can be represented as a function of such transformed numerical arguments. The principle of meaningfulness lets one use any numerical operations in the search for a functional formula and lets one reduce the dimensionality of the search space. Given n arguments and b base arguments, we need only search for a function of $n - b$ arguments (the number of derived arguments).

In the current implementation, COPER searches the space of polynomial functions. This stems from the second principle — the Weierstrass theorem —

which says that any function (fulfilling some formal conditions) can be approximated with any degree of accuracy by a polynomial. A simple-minded application of this theorem to function discovery leads to two problems: it may generate formulas that have no physical meaning, and it may result in formulas much more complex than the physical laws we are seeking. Fortunately, the principle of meaningfulness lets one avoid the first problem, and the problem of complexity can be solved by changing the representation base.

There are usually many ways to subdivide a set of arguments into a base and a set of derived arguments. Each base leads to a different 'language' for stating a functional relation, and COPER takes advantage of this fact. The system first iterates through each base looking for a simple functional formula (a low-degree polynomial). If none of these give satisfactory results (high approximation errors) it reconsiders each base using more complex polynomials.

In the case of Bernoulli's law, COPER discovered the exact functional formula using a polynomial of the second degree and using a base composed of $p1$, ρ, and g.

4. Discussion: COPER, BACON, and ABACUS

Although the basic goal of COPER is similar to the goals of BACON (Bradshaw, Langley, & Simon, 1980; Langley, 1981; Langley, Bradshaw & Simon, 1981, 1982) and ABACUS (Falkenhainer, 1985; Falkenhainer & Michalski, 1986), its approach to solving the problem is totally different. Below we briefly characterize BACON and ABACUS, point out some of drawbacks of those systems, and discuss how COPER deals with these problems.

The activities performed by BACON can be summarized by the following points:

1. BACON generates functions $y = f(x1, x2, \ldots, xn)$ that describe a set of data.
2. The system holds $n - 1$ variables constant (at some level) and varies only one term, $x1$. Therefore it analyzes a function $y = f1(x1)$ when $x2, \ldots, xn$ are held constant.
3. It assumes that $f1$ is either a linear or a hyperbolic function ($y = Cx$ or $y = Cx^{-1}$), where C depends on the variables $x2, \ldots, xn$.
4. If one of these assumptions fits the data then BACON derives the hypotheses $y/x1 = C$ or $y \cdot x1 = C$ respectively, where $y/x1 = C$ (or $y \cdot x1$) is called a 'theoretical term.' If the independent variable $x1$ is of nominal type and y changes when $x1$ changes, then an 'intrinsic property' is proposed whose values are set initially to the values of y.
5. The system treats $y/x1$ (or $y \cdot x1$) as a new variable dependent on $x2, \ldots, xn$.
6. BACON then postulates that $C = y/x1 = C'x2$ (or $C'/x2$) and repeats the

entire procedure again, having reduced the problem size by one variable. The method of reduction is decomposition of a function into a product of functions, each of them is dependent on one only variable,

$$f(x1, x2, \ldots, xn) = f1(x1) \cdot f2(x2) \cdot \ldots \cdot fn\,(xn).$$

Despite its successes, BACON's ability to discover physical laws is limited along a number of dimensions:

1. BACON generates hypotheses by taking into consideration only one hypersurface in the experimental space (varies one variable while holding the rest of them constant). It ignores all cross-impact effects (for a different values of the constants $x_2, \ldots, xn$ the hypothesis could be different) and, moreover, it does not take into consideration all the information available to it at the moment. All this makes BACON highly vulnerable to errors.
2. The system is not able to make any decisions about arguments that have not been changed in the experiments. For instance, if the resistance of the wires stayed constant for all cases it would not be able to test the relevance of this variable.
3. BACON does not have provisions for testing the completeness of arguments. If one or more relevant arguments stay constant in the experiments, the system cannot detect that arguments are missing from the description.
4. The program treats physical arguments as strictly numerical, and henceforth can generate relationships that have no physical meaning, such as $\text{mass} = \text{time}^{-1}$.

ABACUS incorporates most of the methods used by BACON, as well as several new abilities:

1. The system does not require the user to distinguish between independent and dependent arguments.
2. ABACUS constrains the search for functional dependencies by analyzing the units of the arguments.
3. The program has the ability of subdividing the domain and fitting a function to each of the subdomains separately.
4. ABACUS generates logical expressions describing each of these subdomains; these expressions serve as preconditions for the functional descriptions.

These additional capabilities make ABACUS more powerful than BACON. Nevertheless, it shares many of the limitations we have already listed for BACON.

In contrast to these systems, the COPER method has several advantages:

1. Most important, COPER can reason about those arguments which have not changed in an experiment. The system can test relevance of such arguments and it can discover arguments, such as acceleration due to gravity (g) or density in Bernoulli's law (ρ), which were not varied in the observations.
2. Another advantage is that COPER can test whether a set of arguments satisfies the condition of complete relevance. If this condition is violated, COPER can generate new descriptions (dimensional formulas) for the missing arguments.
3. COPER bases its conclusions on the entire data that are available, making it less vulnerable to errors.
4. Physical quantities are treated not as numbers but rather as elements of the 'physical quantities space' used in the theory of dimensional analysis. Thus, the COPER method takes both syntactic and observational information into account. The powers in dimensional formulas are not derived by experimentation, but are instead generated by the syntactic rules for the language of physical quantities.

In summary, the COPER method has a number of advantages over other machine discovery systems. Of course, more work remains to be done, but we feel the current system provides a solid base for this research.

5. Conclusions

In this paper we have focused on testing and generating arguments of functional descriptions from observation. We described COPER, a system that accepts as input an initial set of arguments expressed in terms of the description language, along with a set of observations (values of the arguments). The goal of the system is to test whether the set of arguments is complete and, if not, to generate descriptions for additional relevant arguments. COPER uses the functionality of an event set to decide whether to include a particular argument in the set of relevant arguments, and to determine whether that set is complete. The observed events fulfill the condition of functionality if the dependent argument is constant for all events having the same values of the independent arguments.

For purely numerical functions, detecting nonfunctionality of an event set does not aid the search for missing arguments. Moreover, if a relevant argument does not change in the observed events, then the functionality condition is not violated, and one cannot detect that an argument is missing. In this paper, we showed that for a special class of functions — known as invariant functions — one can detect missing arguments even when functionality is satisfied.

Invariance of functions is closely linked to the notion of meaningfulness. Functions are meaningful in a language if they are expressed solely in terms of the

syntactic operations of the description space. Such meaningful functions are also invariant.

We have seen that, if a function fulfills the invariance condition, then one can predict its values for a class of events (orbit), provided that the value of the function for one event from this class is known. This feature lets one test whether a given set of arguments is complete and whether a particular argument is relevant. Moreover, one can reach conclusions about completeness and about relevance of an argument even from observations in which its value did not change.

Based on this insight, we described COPER, a system for testing the completeness of an argument set and the relevance of a particular argument. The system requires a set of observations that can be classified into orbits, and for which it is possible to predict the value of the function. If the predicted values are equal to the observed ones, then there is no reason to search for additional arguments. However, if the observed values differ significantly from the predictions, then COPER carries out a search for additional arguments. The system adds that argument which causes the greatest improvement in predictive ability, and the process is repeated until the data have been completely predicted.

Thus, COPER has the ability both to evaluate existing arguments and to generate new terms. We have tested these abilities on a number of physical laws, including the law of falling bodies and Bernoulli's law. Also, because the assumption of invariance is epistemological in nature, we should be able to use the same approach for discovering arguments of nonphysical functional descriptions.

The ultimate goal of COPER is similar to the goal of BACON (Bradshaw et al., 1980; Langley, 1981; Langley et al., 1981, 1982) and ABACUS (Falkenhainer, 1985; Falkenhainer & Michalski, 1986), but the approach is quite different. First, COPER makes use of invariance to direct the search for new arguments. Another difference is that the search is not purely data driven. The methodology takes into account syntactic knowledge and observational information. This gives COPER the ability to discover new arguments such as gravity and density, even though these terms were not varied during the experiment. In this paper we focused on methods for discovering the arguments of a functional description, but COPER includes also methods for discovering sophisticated functional forms such as Bernoulli's law (cf., Kokar, 1986).

Acknowledgments

The author wishes to thank Ryszard S. Michalski for his helpful suggestions and remarks concerning this work. His comments, criticism and encouragement have been most helpful for the author. The author also wishes to express his sincere gratitude to Pat Langley for his very detailed and constructive comments and suggestions, as well as for reformulation of many complex sentences and paragraphs in two drafts of this paper.

References

Bradshaw, G.L., Langley, P., & Simon, H.A. (1980). BACON.4: The discovery of intrinsic properties. In *Proceedings of the Third National Conference of the Canadian Society for Computational Studies of Intelligence* (pp. 19–25). Victoria, BC, Canada.

Causey, R.L. (1969). Derived measurement, dimensions and dimensional analysis. *Philosophy of Science, 36*, 252–269.

Drobot, S. (1953). On the foundations of dimensional analysis. *Studia Mathematica, 14*, 84–89.

Falkenhainer, B. (1985). Proportionality graphs, units analysis, and domain constraints: Improving the power and efficiency of the scientific discovery process. In A. Joshi (Ed.) *Proceedings of the Ninth International Joint Conference on Artificial Intelligence* (pp. 552–554). Los Angeles: Morgan-Kaufmann.

Falkenhainer, B.C. & Michalski, R.S. (1986). Integrating quantitative and qualitative discovery: The ABACUS system. *Machine Learning, 1*, 367–401.

Kokar, M.M. (1978). A system approach to a search of laws of empirical theories. In J. Klir (Ed.), *Current topics in cybernetics and systems*. Berlin: Springer-Verlag.

Kokar, M.M. (1979). The use of dimensional analysis for choosing parameters describing a physical phenomenon. *Bulletin de l'Academie Polonaise des Sciences, Serie de Sciences Techniques, XXVII, 5/6*, 249–254.

Kokar, M.M. (1985). *On invariance in dimensional analysis* (Technical Report MMK-2/85). Boston: Northeastern University, College of Engineering.

Kokar, M.M. (1986). Discovering functional formulas through changing representation base. In *Proceedings of the Fifth National Conference on Artificial Intelligence* (pp. 455–459). Philadelphia: Morgan-Kaufmann.

Krantz, D.H., Luce, R.D., & Suppes, P. (1971). *Foundations of measurement*. New York: Academic Press.

Langley, P. (1981). Data-driven discovery of physical laws. *Cognitive Science, 5*, 31–54.

Langley, P., Bradshaw, G., & Simon, H.A. (1981). BACON.5: The discovery of conservation laws. In *Proceedings of the Seventh Joint International Conference on Artificial Intelligence* (pp. 121–126). Vancouver, BC, Canada: Morgan-Kaufmann.

Langley, P., Bradshaw, G., & Simon, H.A. (1982). Data-driven and expectation-driven discovery of empirical laws. In *Proceedings of the Fourth National Conference of the Canadian Society for Computational Studies of Intelligence* (pp. 137–143). Saskatoon, Canada.

Luce, R.D. (1971). Similar systems and dimensionally invariant laws. *Philosophy of Science, 6*, 157–169.

Luce, R.D. (1978). Dimensionally invariant physical laws correspond to meaningful relations. *Philosophy of Science, 45*, 1–16.

Michalski, R.S. (1980). Knowledge acquisition through conceptual clustering: A theoretical framework and an algorithm for partitioning data into conjunctive concepts. *Policy Analysis and Information Systems, 4*, 219–244.

Michalski, R.S., Carbonell, J.G., & Mitchell, T.M. (Eds.). (1983). *Machine learning: An artificial intelligence approach*. Palo Alto, CA: Tioga.

Whitney, H. (1968). The mathematics of physical quantities. *American Mathematical Monthly*, 75, 115–138 and 227–256.

Winston, P.H. (1975). Learning structural descriptions from examples. In P.H. Winston (Ed.), *The psychology of computer vision*. New York: McGraw-Hill.