PREDICATE CALCULUS FEATURE GENERATION

David Rothenberg
University College, Rutgers University

Abstract

A summary description (proof of theorems omitted): An adaptive pattern representation and recognition strategy for application to mechanized interpretation of (sampled) pictorial data (other applications are appropriate) is described. The system generates its own features which are formulae in a subset of the weak (in the sense that only quantification over finite sets is permitted) second order predicate calculus. The models of such formulae define the "objects" in a description of the data, which is hierarchical both with respect to features and extensions. The hierarchy is automatically constructed, thereby implementing changes in "problem representation". Relations between the syntax and semantics of formulae in the weak second order predicate calculus are derived (by extending the syntax of the calculus) and utilized. Minimal use is made of the finiteness of the input data by the methods employed. That is, in pictorial pattern recognitions, the adaptive feature generation (i.e., "learning") algorithms are independent of the fineness of grain of the sampling of an input picture. Because this approach is used, many difficult problems of a purely mathematical nature acquire practical importance. Computation is reduced through the use of topological methods and the system is at present in a stage of development appropriate for programming and use in a variety of practical applications.

1. The Problem

The problem definition used conforms well to the theoretical framework proposed by Banerji (1969, 9-16, 103-175):

From a finite set of points, $U = \{x_1, x_2, \ldots, x_n\}$, two[1] disjoint collections, $\bar{K}$ and $\bar{C}$, of given subsets of U (called regions) are given. That is, if "R_i" denotes a region in $\bar{K}$ and "R_j" a region in $\bar{C}$,

$$\bar{K} = \{R_1, R_2, \ldots, R_n\}$$

$$\bar{C} = \{R_{N+1}, R_{N+2}, \ldots, R_{N+M}\}$$

$$\forall i \forall j (R_i \neq R_j) \text{ when } R_i \in \bar{K} \text{ and } R_j \in \bar{C} .$$

A formula in the weak second order predicate calculus, $\bar{P}(A)$, (called an ideal predicate or ideal feature) must be generated which is strongly satisfied by all regions R_i in $\bar{K}$, and by no region R_j in $\bar{C}$. We define a formula as being strongly satisfied by a region if it is satisfied by that region and is a proper subset of no other region which satisfies the

[1] This, of course, is easily generalized to n distinct collections.

formula. (Of course, a feature can have only one free (set) variable ("A" in "$\bar{P}(A)$").) Trivially, such a formula can be a disjunction of the conjunction of (the memberships of) all points of U (i.e., "constants") in each set in the collection $\bar{K}$. However, the "shortest" possible formula (Banerji, 1969, 103-104) is sought: From a given set of unary and n-ary atomic predicates (all sequences of points in U which satisfy each of these is given), a formula is constructed in which both the number of constants (specific points in U) and variables (both point and set) are minimal. (This definition of "shortness" somewhat resembles those developed by Kolmogorev and discussed in Löfgren (1967) and Solmonoff (1964a,b).) Such a substitution of a formula containing quantified variables for a formula containing a large number of constants (but no quantified variables) tends to result in the satisfaction of the formula by sets not in any of the collections. Hence, the formula is said to generalize (Banerji, 1969; 104, 157, 168, 175) the expression which involves only constants.

Actually the predicate calculus syntax is here extended to include a modified quantifier, $\bar{Q}^{(k)}x$, which may be read, "for k · 100 percent of x" (similarly to the reading of "$\forall x$" as "for all x" and "$\exists x$" as "there exists an x"). The degree of satisfaction of a set, $R_n \subset U$, with respect to a feature P(A), when R_n does not satisfy P(A), is defined as -1 plus the largest value of k in all replacements of a single universal quantifier by $\bar{Q}^{(k)}$ in P(A) such that R_n satisfies P(A) as a consequence of that replacement. When R_n satisfies P(A), the degree of satisfaction is the largest value of k in all replacements of a single existential quantifier by $\bar{Q}^{(k)}$ such that R_n still satisfies P(A). A topology for the space, U, is developed so that U can be mapped into a space, S, with fewer points, in such fashion that the resulting loss of relevant information is bounded. That is, each $R_n \subset U$ has a "degraded image", $\bar{R}_n \subset S$, such that if R_n satisfies P(A), then the degree of satisfaction of $\bar{R}_n$ with respect to P(A) differs from that of R_n by a bounded (and relatively small) amount. If P(A) is an ideal feature and all $R_i \in \bar{K}$ satisfy P(A) with a degree of satisfaction greater than zero and all $R_j \in \bar{C}$ fail to satisfy P(A) (i.e., have degree of satisfaction with respect to P(A) of less than zero), then U may be mapped into a space S whose size (number of elements) is a function of the difference of such degrees of satisfaction. No loss of recognition ability results while the consequent economy in computation is evident.

Also, a hierarchical theory of levels is utilized so that the problem of generating an ideal feature can be solved by stages, wherein at each except the final level, features which reduce the magnitude of the problem are generated. Here the "points" at each successive level represent the sets which satisfy the features at the previous level and the "atomic predicates" at each succes sive level derive from the features at the previous

level. The method automatically advances one level whenever the number of elements in the regions to be partitioned (or the number of such regions) is thereby reduced. The procedure at each level is identical. A different "representation of the problem" is thereby expressed at each successive level (Nilssen, 1971; Amarel, 1962; Sherman and Ernst, 1969).

The material in this paper is organized as follows. The syntactic methods are described in the first portion. Features are defined as formulas in the weak second order predicate calculus with one free set variable. The atomic predicates used specify the positions of, darkness of, and other measurements at, and relations between, each of a set of data points called the data space.

These atomic predicates are called data independent if they do not vary for different inputs (e.g., the relative positions of the data points at which a photograph is sampled by a flying spot scanner are data independent, i.e., data independent predicates do not depend upon darkness, color, etc.). Atomic predicates are called data dependent if and only if they vary for different inputs (e.g., the darkness of a given data point on a particular photograph). Set membership is, of course, also a primitive of the system. Objects are regions (subsets) of the data space which satisfy a feature and which are maximal (i.e., they have no proper supersets which also satisfy the feature—this is called strongly satisfying the feature). A descriptive basis consists of all features generated by the learning algorithm at a particular stage in the operation of the system. A description consists of the descriptive basis and a list of the elements in each of the objects.

All formulae are put in distributive form. In such formulae the scope of each quantifier is as small as possible, negations occur only before atomic formulae, all subformulae within the scope of a universal quantifier are connected by disjunctions, and subformulae within the scope of an existential quantifier are connected by conjunctions. Such a formula is conveniently represented by a directed graph which is a rooted tree wherein the subtree below each node represents a subformula consisting of a quantifier (if such is present) and all subformulae within its scope. Each branch from that node represents a subformula within the scope of that quantifier, and unquantified atomic formulae are the bottom nodes of the tree.

All features have corresponding computer programs which evaluate that formula in a given space. If a formula is placed in prenex form and adjacent like quantifiers exchanged to generate additional prenex forms, distinct distributive forms may be obtained by operating on each such prenex form. This set of distributive forms is called the distributive set of the formula and each such distributive form corresponds to a computer program which can evaluate the formula.

A procedure called abstraction compares the distributive set of one feature with that of another to determine common subformulae (i.e., common subprograms). Such sub-

formulae are extracted, are called derived predicates and are retained by the system. In effect, abstraction extracts identical subtrees from the graphs of distinct features.

Another procedure called relation abstraction exposes the similarity of structure of pairs of trees with different atomic formulae. That is, formulae which differ only in that they contain different subformulae which, nonetheless, have the same free variables in corresponding positions, are extracted and retained. In these retained formulae, called predicate forms, differing subformulae in the features originally compared correspond to a dummy predicate, wherein the free variables are shown, but the identity of the predicate is not. Such dummy predicates will later be replaced by a derived predicate with the same free variables.

Derived predicates and predicate forms are compared to produce two hierarchies of formulae, the predicate hierarchy and the relational hierarchy, later used in the "learning algorithm".

Part 12 discusses relations between the syntax of the formulae and certain properties of those regions (i.e., subsets) of the data space which satisfy the formulae. Such regions are called models of the formula they satisfy and are compared both with those given regions we desire to be models of the formula, $\overline{K} = R_1, R_2, \ldots, R_N$, and those given regions we desire not to be models of the formula, $\overline{C} = R_{N+1}, \ldots, R_{N+M}$. Measures of the "similarity" between the actual models of a formula and the given regions determine various measures of success of the features generated. The ease of satisfaction of a formula is defined as the probability that a randomly selected element from the product space over which its free variables range satisfies the formula. The overlap of a pair of formulae is the ease of satisfaction of the conjunction of two formulae. Mutations map from one feature to another. Relaxing mutations take a formula into another which is implied by it (ease of satisfaction is increased), restricting mutations do the reverse and neutral mutations connect formulae which are not related by implication. Restricting and relaxing mutations (with the exception of quantifier exchanges) correspond to the addition or elimination of subtrees from some node on the graph of the formula, whereas neutral mutations correspond to replacements of subtrees of the graph. All are related to the abstraction procedure.

Each feature, P_i, is broken into a conjunction of two formulae, P_i^I and P_i^D wherein P_i^I contains only data independent atomic predicates. We now examine the models of P_i^I in two sets of spaces, one consisting of each of the given regions in $\overline{K}$ and $\overline{C}$ (some of which we wish to be models of P_i and some which we do not) and the other consisting of each model of P_i^D (in the data space). Because models of a feature must strongly satisfy it, we do not, in this case, have the law of the excluded middle (i.e., a region which is not a model of P_i need not be a model of $\sim P_i$), and we are able to derive results relating the ease of satisfac-

tion of a given feature with that of an ideal feature which we are attempting to generate (i.e., one of which each element of $\bar{K}$ is a model and none in $\bar{C}$ is). That is, by examining the size and number of models of a feature in the above spaces we are able to determine what kind of mutation is required.

One result of the above is: if a given region R_i is an element of $\bar{K}$ (those we wish to be models of P_i), then (a) when R_i does not satisfy P_i a relaxing mutation on P_i is required, and (b) when R_i satisfies P_i but not strongly, a restricting mutation is required. When R_i is an element of $\bar{C}$ (those regions which must not be models of P_i) and R_i strongly satisfies P_i, any mutation may suffice (although a restricting mutation is usually preferred). These criteria are, of course, averaged for all elements of $\bar{K}$ and $\bar{C}$.

Now a more powerful technique of mutation selection is exposed: Suppose $R_i \in \bar{K}$ and R_i does not satisfy P_i. In P_i each of the nested universal quantifiers ("$\forall x$") is replaced one at a time, starting with the outermost (highest on the tree) by the modified quantifier "$\bar{Q}^{(k)}x$" which computes the fraction of substitutions of x (as constrained by any set membership atomic predicates, "$\epsilon(x, A)$" within the scope of that quantifier) for which the entire formula holds true (such fractions may equal zero). That is, the original universal quantifier, "$\forall x$", is weakened to mean "for $k \cdot 100$ percent of x" instead of "for all x". This is done independently for each universal quantifier, and such a fractional value is thereby computed for each nested subformula within the scope of each universal quantifier (i.e., subtree on the graph). Clearly this value indicates the amount by which the "ease of satisfaction" of the subformula must be increased if the given set is to be a model of the resulting (modified) feature. The "degree of undersatisfaction" of a feature with respect to a given set is defined as the largest of the above computed fractional values minus one. After this procedure is completed, each of the existential quantifiers ("$\exists x$") are, one at a time, replaced by "$\bar{Q}^{(k)}x$" and again a fractional value is computed for each nested subformula within the scope of an existential quantifier. Now each (except the bottom) node on the tree representing a formula has such a fractional value associated with it.

Now suppose $R_i \in \bar{K}$ and satisfies P_i but not strongly. Only the latter of the above procedures is utilized: each existential quantifier is replaced by "$\bar{Q}^{(k)}x$" and a set of corresponding values is determined. The smallest of these fractional values is defined as the "degree of oversatisfaction" of the feature with respect to the given set.

It is then shown that modified calculations of the above fractional values can be used to estimate the ease of satisfaction of the various subformulae of a feature when that feature is evaluated. Thus, since derived predicates are extracted from features, their ease of satisfaction is automatically estimated and stored. These derived predicates are the subtrees which are added and eliminated from the graph of a feature when mutation is performed. Note that the replacement of universal quantifiers by a "$\bar{Q}^{(k)}x$" quantifier is equi-

valent to a relaxing mutation of the formula consisting of the universal quantifier and the subformulae within its scope. Such an alteration of the quantifier is then repaired by performing an actual relaxing mutation on the subformula and restoring the universal quantifier. This mutation usually involves the addition or elimination of a subformula (i.e., derived predicate) whose choice is determined by its ease of satisfaction, which should be as consistent as possible with the fractional value of the modified quantifier. Neutral mutations can be similarly (but more weakly) determined by the estimated "overlap" (defined previously) of the required mutations.

The above procedures will not always uniquely select a mutation. When a feature is generated its measures of success are evaluated and a corresponding number called a reinforcement, which is assigned to that feature together with all derived predicates and predicate forms contained within it, is appropriately altered. Thus all formulae and predicate forms have such corresponding indications of their "success" thus far. When several mutations of a feature are equally appropriate, a selection is made as follows: The feature is compared with each formula in the list of predicate forms. That predicate form with the highest reinforcement which will result from a mutation of the required type and amount (of change) is chosen. The dummy predicates are either replaced by subformulae of the features being mutated, or if not therein present, such replacements are chosen from the list of derived predicates (with the required numbers of free point and set variables). The derived predicates with the highest reinforcements are chosen first. Hence the enormous possibilities of mutation choice are reduced first, by the semantic criteria resulting from examination of the models of the feature being altered, and second, by the syntactic criteria which favor those predicate forms and derived predicates which characterize previously successful features.

The next part of the paper deals with the technique for reducing the size of the problem by means of the use of a hierarchical level theory (discussed earlier) for representing the given data by the formulae in the descriptive basis. That is, a technique is given for automatically representing objects as collections of subobjects which are collections of subsubobjects, etc., all such objects being defined by features which are similarly hierarchically organized.

The following part of the paper treats the topology of the data space as related to techniques for computation reduction. The most important notion is a set of reduction mappings which successively reduce the size of the data space (e.g., the fineness of grain of sampling). These mappings are related to invariances with respect to transformations which dilate or contract the space. The way in which resulting errors depend upon such a reduction mapping together with a particular feature are studied.

The final part describes the overall systematic procedure for solving the problem. Various important open questions, whose solution would vastly affect the power of the system are discussed. For an illustrative example which simulates the procedures described, see Rothenberg (1973b).

It is important to note that the techniques developed permit a human trainer to simply and directly insert any knowledge or intuitions he has about the problem at any time during or prior to computation. Hence the task of problem solution begins at the limits of the trainer's knowledge and is less difficult than would otherwise be expected.

The description below is geared for pictorial pattern interpretation. It is easily seen, however, that other pattern interpretation problems fit the same formalism

2. Syntax

The sensory data or input consists of a digitized sampling of a (single) picture. This sampling is at a finite ordered array of points called the data space, $U = \{x_1, x_2, \ldots, x_n\}$, which is the same for all pictures. For each such data point (which corresponds to a receptor in the sensory field such as a photosensitive cell), there is a vector of k values called its value vector, each element of which is called a data value. For illustrative purposes we will here assume that the first two positions in the vector specify the row and column positions, respectively, (in the ordered array) of the data points. The remainder of the data values specify different quantized qualities (i.e., color or darkness) of that data point on a particular picture (these correspond to a quantized energy density at a particular receptor). The array of all value vectors (at all data points) is called the value matrix and is denoted $\|v^{(k)}\|$ where k indexes the positions in the value vector (the particular fineness of grain of sampling is specified by the size of the matrix). For convenience, each lower case letter, "x", "y", etc. will be used to designate a data point (i.e. the first two positions of the value vector), the k^{th} data value of point x will be denoted "$v^{(k)}(x)$", and upper case letters, "A", "B", etc. will denote sets of data points.

Data values determine the truth values of a set of primitives (also called atomic predicates[1]), $\alpha^{(k,i)}(x)$, which are "true" if $v^{(k)}(x) = i$ and "false" otherwise. Those for which $k > 2$ are called data dependent primitives because they differ for different input photographs.

Also given are a set of n place primitives, $\beta(x_1, x_2, \ldots, x_n)$ which specify (geometrical) relations between data points. A typical example would be "$<(x, y, z, w)$" which here

[1] Conventionally, the term "primitive" refers to a specified "property" or relation (e.g., "ϵ" for "set membership"), whereas "atomic predicate" refers to the primitive together with a specific set of arguments (e.g., "$\epsilon(x, A)$"). The usage of these terms here ignores the distinction.

means that data point x is closer to y than z is to w. This is usually derived from a given metric on the data space (which is a function of $v^{(1)}(x)$ and $v^{(2)}(x)$ — the row and column positions of point x). Such atomic predicates (as well as $\alpha^{(1,i)}(x)$ and $\alpha^{(2,i)}(x)$) are called data independent because they are the same for all input photographs.

The following definitions clarify the syntax used.

(1) Region: Any subset of a data space.

(2) Feature: A formula, P(A), in the second order predicate calculus with equality such that:

(a) point variables $(x, y, \ldots)$ range over data points, set variables $(A, B, \ldots)$ range over regions, and all constants $(c_1, c_2, \ldots)$ are data points, or sets of data points $(C_1, C_2, \ldots)$.

(b) all variables are bound except one set variable, A (hence only regions can satisfy a feature).

(c) primitives include "$\epsilon(x, A)$" (conventionally "$x \epsilon A$") and "$= (x, y)$" (conventionally "$x = y$")[1], both of which are called elementary primitives. "card(A) < card(B)" may also be used.

(Also included are the data independent and dependent primitives (atomic predicates) defined above.)

Certain other conditions may be included which guard against formulae with trivial interpretations in the pictorial case. These will not be discussed here (see Rothenberg, 1973b).

(3) Object: A region which satisfies a feature (i.e., is defined as a realization of that feature) and is not a proper subset of any other realization of that same feature (i.e., it is "maximal). Note that while a region may be an object because strongly satisfies one feature, it may satisfy another feature, but not strongly. When an object satisfies a feature and is maximal with respect to that particular feature it will be said to strongly satisfy that feature and is called a model of the feature (as distinct from a realization). When a set of points satisfies a feature but is not maximal for that feature, it will be said to satisfy that feature, but not strongly.

(4) Descriptive Basis: All features which have been generated by the learning algorithm (which generates such features) at a particular stage in the adaptive development of the analysis.

---

[1] Note that although "$= (x, y)$" will be used as an atomic predicate, it can, in the case of our illustrations, be defined as follows:

$= (x, y) \equiv \forall z \left(\sim < (x, z, y, z) \wedge \sim < (y, z, x, z) \right)$ (see "$= (x, y, z, w)$" definition in Example 1).

(5) Description: This contains all features in the descriptive basis and lists all objects strongly satisfying each of these features and the elements of each such object. The satisfaction of a feature by an object which does not strongly satisfy that particular feature is also indicated.

(6) Object Class: A set of objects all of which strongly satisfy the same feature (also defined as the extension of the formula for that feature).

(7) Derived Predicates: A subformula of a feature which is recursively built up from primitives by the use of logical connectives and quantifiers. They are denoted "$D_i(x_1,\ldots,x_n, A_1,\ldots,A_m)$" where i indexes the predicate and its argument includes all free variables.

(8) Data Dependent Predicates: All formulae (including features) which contain data dependent primitives. All other formulae are called data independent.

Example 1

Let the primitives be "$\epsilon(X,A)$", "$=(x,y)$", and "$<(x,y,z,w)$", which specifies that the distance between data point x and data point y is less than the distance between z and w. Also let "$\bar{D}(x)$" indicate that a data value (the third position in the value vector at data point x) is equal to 1. Here we assume that these data values are either one or zero, the former indicating a "dark" and the latter a "light" point. Then examples of derived predicates are:[1]

$=(x,y,z,w) \equiv \sim <(x,y,z,w) \wedge \sim <(z,w,x,y)$ ("(x, y) equals (z, w)")[2]

$D_b(x,y,z) \equiv \sim =[x,y] \wedge \sim =[x,z] \wedge \sim =[y,z] \wedge \forall w\left(=[y,w] \vee <[x,y,x,w] \vee <[z,y,z,w]\right)$
(y is "between" x and z)

$D_c(A,B) \equiv \forall x\left(\epsilon[x,A] \rightarrow \epsilon[x,B]\right)$ ("A is contained in B")

$D_s(A,B) \equiv D_c(A,B) \wedge D_c(B,A)$ ("A is the same as B")

$D_\phi(A) \equiv \sim \exists x\left(\epsilon[x,A]\right)$ ("A is the null set")

$D_d(A) = \forall x\left(\epsilon[x,A] \rightarrow \bar{D}[x]\right)$ ("all points in A are 'dark' points")

$D_a(x,y) = \sim =[x,y] \wedge \forall z\left(\sim D_b[x,z,y]\right)$ ("x is adjacent to y")[3]

---

[1] Note that throughout this description square brackets "[]" and round brackets "()" are used interchangeably and are alternated for clarity only.

[2] Note that "=" when here used as a four place predicate, "$=(x,y,z,w)$", has a different meaning than when used as a two place primitive, "$=(x,y)$" (meaning that x and y are the same point).

[3] For use with a coarse (sampling) grid it is suggested that $D_a(x,y) = \sim =(x,y) \wedge \forall z\ (=[x,z] \vee \sim <[x,z,x,y])$ ("There is no point closer to x than y") be used instead of the above.

Note that all y which satisfy $D_b(x,y,z)$ when x and z are fixed will approximate a "straight line" even if the grain of the data space is coarse. As the grain becomes infinitely fine these points approximate a "true" straight line.

For convenience[1] we will use the symbol "$\psi(x_1,x_2,\ldots,x_n)$" to denote that all variables within brackets denote distinct objects:

$$\psi(x,y,z) = \sim = (x,y) \wedge \sim = (x,z) \wedge \sim = (y,z) \ .$$

Hence D_b ("betweenness") in Example 1 may be rewritten

$$D_b(x,y,z) = \psi(x,y,z) \wedge \forall w\Big(= [y,w] \vee < [x,y,x,w] \vee < [z,y,z,w]\Big).$$

Note that a large number of formulae are tautologous unless their free variables are distinct.

3. Semantics (Illustrations)

The translation from formulae to programs is simple. Quantifiers become "do loops" whose ranges are determined by the membership predicates, e.g.

$$\forall x_1 \left[\sim \epsilon(x_1,A) \vee P(x_1,\ldots,x_n)\right] \equiv \bigwedge_{x_1 \epsilon A} P(x_1,\ldots,x_n)$$

$$\exists x_1 \left[P(x_1,\ldots,x_n)\right] \equiv \bigvee_{x_1 \epsilon U} P(x_1,\ldots,x_n)$$

$$\exists x_1 \left[\epsilon(x_1,A) \wedge P(x_1,\ldots,x_n)\right] \equiv \bigvee_{x_1 \epsilon A} P(x_1,\ldots,x_n)$$

("U" is the universal set — i.e., the data space)[2]

Logical connectives are functions computed by subroutines, and atomic and derived predicates become subroutines in the main program expressing the feature.

Some examples of the richness of the system when the primitives of Example 1 are used are:

$P_1(A) \equiv \forall x \forall y \forall z \Big(\epsilon[x,A] \wedge \epsilon[z,A] \wedge D_b[x,y,z] \rightarrow \epsilon[y,A]\Big)$ is data independent and satisfied by convex sets only ($D_b[x,y,z]$ is defined as in Example 1). If $P_1(A)$ is a feature, then, by definition, only maximal regions are objects.

Let $D_d(A) \equiv \forall x \Big(\epsilon[x,A] \rightarrow \bar{D}[x]\Big)$ (as before) .

Then

$$P_2(A) \equiv D_d(A) \wedge P_1(A)$$

is a data dependent version of $P_1(A)$

---

[1] Although not described here, the language has been generalized to include "syntactic variables" which range over formulae and primitives which specify whether these variables are bound or free. Programming is thereby facilitated when the language is used as a compiler (see Pessin and Rothenberg reference).

[2] Quantified set variables range over regions. Methods of reducing such computation are discussed in Rothenberg (1973b)

$$P_3(A) \equiv \exists x\left[\epsilon(x,A) \wedge \forall y \forall z\left(\epsilon[y,A] \wedge D_b[x,z,y] \rightarrow \epsilon[z,A]\right)\right]$$

defines a "starlike" region, and a derived predicate which is satisfied only by a boundary point, x, of a region, A, is given by

$$D_1(x,A) \equiv \epsilon(x,A) \wedge \exists y\left(\sim\epsilon[y,A] \wedge D_a[x,y]\right)$$

A feature satisfied only by a connected[1] region can be constructed in two steps:

$$D_2(A,B) \equiv \forall x \forall y\left(\epsilon[x,A] \wedge \epsilon[y,B] \wedge D_a[x,y] \rightarrow \epsilon[y,A]\right)$$

is a derived predicate which asserts that A is "closed" in B under the relation of adjacency and

$$P_6(A) \equiv \forall B\left(D_c[B,A] \wedge \sim D_\phi[B] \wedge D_2[B,A] \rightarrow D_s[B,A]\right)$$

is satisfied iff A is a connected region ("for all B, if $B \subset A$ and $B \neq \phi$ and $D_2[B,A]$ then $B = A$"—see Example 1). Other equivalent formulae exist.

Simple connectedness ("A has no holes") is easily expressed by stating that the boundaries of both A and its complement are connected sets. Other illustrations are easily constructed.

An example of a data dependent feature is $P_7(A)$ which accommodates "noisy" dark areas:

$$P_7(A) \equiv \forall x\left(\epsilon[x,A] \rightarrow \bar{D}(x) \vee \forall y\left[D_a(x,y) \rightarrow \bar{D}(y)\right]\right).$$

4. Distributive Normal Forms

This is a form in which all quantifiers are distributed as much as possible along the terms of a formula so that a minimum number of terms lie within the scope of each quantifier. (It is the opposite of a prenex normal form and is similar to, but not identical to, the "mini-scope" of Wang (1960) and somewhat resembles the "distributive normal forms" of Hintikka (1953). This is accomplished on a formula by reversing the techniques for placing formulae in prenex normal form (Rothenberg, 1971). A formula is in distributive normal (henceforth abbreviated "distributive") form iff:

(1) Negations appear only before atomic subformulae.

(2) All subformulae[2] within the scope of a universal quantifier are connected by disjunctions.

---

[1] By "connected region" is intuitively meant that for each pair of sub-regions which partition the region there exists a pair of adjacent points, one of which is contained in each sub-region.

[2] A subformula is either an atomic predicate or derived predicate or either of these quantified and/or negated.

(3) All subformulae within the scope of an existential quantifier are connected by conjunctions.

(4) All subformulae within the scope of a quantifier are either prefixed by a quantifier or are atomic formulae or their negations. (That is, they are not bracket-enclosed conjunctions or disjunctions of other subformulae.)

(5) All subformulae within the scope of a quantifier contain the quantified variable as an argument.

(6) If two or more like quantifiers are adjacent (e.g., "$\forall x \forall y$" or "$\exists x \exists y$"), all subformulae within their scope must contain such adjacently quantified variables as arguments.

Example 2.

Consider the formula

$$P_8(A) \equiv \forall x \exists y \left(\left[\epsilon(y, A) \vee \epsilon(x, A) \right] \rightarrow \exists z \left[D_b(x, y, z) \wedge \epsilon(z, A) \right] \right) .$$

It has only one distributive form (in general, this is not the case—to be discussed):

$$P_8(A) \equiv \forall x \left(\sim \epsilon [x, A] \vee \exists z \left[\epsilon(z, A) \wedge \exists y \left(D_b [x, y, z] \right) \right] \right)$$
$$\wedge \exists y \left(\sim \epsilon [y, A] \right) \vee \forall x \exists z \left(\epsilon [z, A] \wedge \exists y \left(D_b [x, y, z] \right) \right)$$

For convenience we here insist that, within the scope of each quantifier, (a) set membership primitives precede all others, (b) quantified terms be placed last, and (c) those terms whose arguments contain fewer variables precede terms whose arguments contain a greater number of variables.

Note that, because of (1) - (6) above, a distributive formula may be easily represented by directed graph which is a rooted tree wherein the subtree below each node represents a subformula consisting of a quantifier (if such is present) and all subformulae within its scope.[1] Each branch from that node represents a subformula within the scope of that quantifier, and unquantified atomic formulae (and their negations) are the nodes with no successors. The number of distinct bound variables in the formula corresponds to the number of edges in the longest path down the tree. Circular nodes will be used to contain quantifiers of point variables, two concentric circles will contain quantifiers of set variables, and rectangular nodes will contain atomic predicates. The point variables will be indexed in their order of quantification (starting at the top of the tree) by integers. Set variables will be similarly indexed by Roman numerals. Free point variables will be denoted by

[1] When the entire formula is not quantified but is instead a logical combination of formulae which are quantified, the top nodes of the graph correspond to connectives "$\wedge$" or "$\vee$" instead of quantifiers. The graph representation is here used only for illustrative purposes.

$0_1, 0_2, \ldots$ etc. in the order in which they appear in atomic predicates. Free set variables will be similarly denoted $\phi_1, \phi_2, \ldots$ etc. When a feature is represented, there will be only one free variable, the set variable, ϕ.

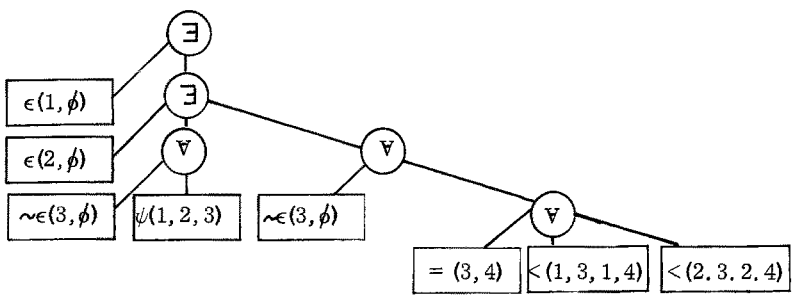
Here an example of a first order feature is shown (for a graph of a second order feature see Example 5):

Example 3

Consider the feature $P_{10}(A)$, which is satisfied by regions which are "straight lines" containing more than two points: A distributive normal form of this formula is

$$P_{10}(A) \equiv \exists x \Big[\epsilon(x,A) \wedge \exists z \Big(\epsilon[z,A] \wedge \forall y \big[\sim\epsilon(y,A) \vee \psi(x,y,z)\big] \wedge$$
$$\forall y \big[\sim\epsilon(y,A) \vee \forall w \big(=[y,w] \vee <[x,y,x,w] \vee <[z,y,z,w]\big)\big]\Big)\Big].$$

A graph corresponding to this formula is



Thus, to decode the above graph, we may begin at the bottom quantifier: Let $A = \phi$, $x = x_1$, $y = x_2$, $z = x_3$, $w = x_4$ and we obtain the following subformula which we will denote "$D_1(x,y,z,A)$"(where x,y,z and A are the free variables):

$$\forall w\big(=[z,w] \vee <(x,z,x,w] \vee <(y,z,y,w]\big) \equiv D_1(x,y,z,A) \quad .$$

If we include the note above it we obtain

$$\forall z\big(\sim\epsilon[z,A] \vee D_1[x,y,z,A]\big) \equiv D_2(x,y,A).$$

Now if we write the formula corresponding to the node to the left of the one just considered:

$$\forall z\ \big(\sim\epsilon[z,A] \vee \psi[x,y,z]\big) \equiv D_3(x,y,A) \quad .$$

Moving up a node on the graph we obtain:

$$\exists y\big(\epsilon[y,A] \wedge D_2[x,y,A] \wedge D_3[x,y,A]\big) \equiv D_4(x,A) \quad .$$

Proceeding to the top node we obtain

$$\exists x\left(\epsilon\left[x, A\right] \wedge D_4\left[x, A\right]\right) \equiv P_{10}(A)$$

which is an alphabetic variant of the formula for $P_{10}(A)$ given at the beginning of the example.

Note that this graph representation ensures that the minimum number of variables is used (Rothenberg, 1973b). In general, prenex normal formulae have more than one distributive form; also, a given distributive formula will often have more than one prenex form (e.g., when two branches from the same node contain differing quantifiers and all branches from that node contain quantifiers, e.g., $\forall x\left(\forall y\left[p(x, y)\right] \vee \exists y\left[q(x, y)\right]\right)$ can be put into the prenex normal forms $\forall x \forall y \exists z\left(p\left[x, y\right] \vee q\left[x, z\right]\right)$ and $\forall x \exists z \forall y\left(p\left[x, y\right] \vee q\left[x, z\right]\right)$.)

Note that the prenex form has more quantified variables than the distributive form. This is often the case.

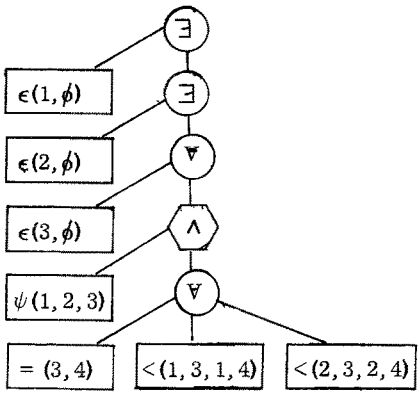
We are here interested in the distributive set of a formula. This is defined as the set of formulae obtained by placing a distributive formula in prenex form, obtaining all prenex forms resulting from the exchange of adjacent like quantifiers, placing each of such prenex forms in distributive form and eliminating the identical formulae which result (more economical methods exist).

Another form of interest here is called the nested form. This corresponds to a prenex formula wherein the quantifiers are imported[1] into the subformulae within this scope until a further such importation would result in a circular node on the graph having two circular nodes as immediate successors. Such a formula is obtained from a distributive formula, whenever two or more quantified subformulae (i.e. subtrees on the graph) are the immediate successors of a quantifier (circular node on graph), by consolidating such subformulae within the scope of a single quantifier. A nested form of the formula in Example 3 is (also see Example 5):

$$P_{10}(A) \equiv \exists x\Big[\epsilon(x, A) \wedge \exists z\Big(\epsilon\left[z, A\right] \wedge \forall y\Big[\sim\epsilon(y, A) \vee \Big(\psi\left[x, y, z\right] \wedge \forall w\left[= (y, w) \vee < (x, y, x, w) \vee < (z, y, z, w)\right]\Big)\Big]\Big)\Big].$$

Note that both conjunctions and disjunctions now appear within the scope of the quantifier of the variable, y. Nested formulae may be graphed by including nodes containing disjunctions and conjunctions. The graph of the nested form of $P_{10}(A)$ above is

[1] That is, prefixed to a subformula rather than the entire formula initially within its scope.



Although formulae, if in distributive normal form, translate into efficient computer programs (nested form is more efficient when it contains no more variables than the distributive form), the principal motivation here is the necessity for such form in the procedures which follow. Also, the "learning" algorithm generates and operates upon formulae in their distributive form.

5. Compression and Decompression of Formulae

In order to compare different formulae which contain identical derived predicates as subformulae, it is necessary to guarantee that such will not be concealed by their form. Hence, we define the following procedures:

Compression: This operation consists of replacing an innermost phrase of a formula by an abbreviation, $D_i(X_1 \ldots x_n, A_1 \ldots A_m)$, with the number of arguments equal to the number of free variables (see Example 3 above). If this predicate is a derived predicate which has already been generated (to be discussed), substitute its abbreviation. This is called one layer of compression. This may be repeated until all phrases are exhausted. When formulae for (all) existing derived predicates are replaced by their abbreviations, the resulting formula is defined as in collapsed form.

Example 4

$$P_1(A) \equiv \forall x \left(\sim\epsilon [x, A] \vee \forall y \left[\epsilon(y, A) \vee \underbrace{\forall z \left(\sim\epsilon [x, A] \vee \sim D_b [x, y, z]\right)}\right]\right)$$

$$P_1(A) \equiv \forall x \left(\sim\epsilon [x, A] \vee \underbrace{\forall y [\epsilon(y, A) \vee D_1(x, y, A)]}\right)$$

$$P_1(A) \equiv \forall x \left(\sim\epsilon [x, A] \vee D_2[x, A]\right)$$

Decompression: This is the reverse of compression where each derived predicate has its definition substituted for its abbrevation in the formula being decompressed. This is done one derived predicate at a time, working from the innermost outward. Each such

substitution is defined as one layer of decompression. This may be repeated until only primitives remain, in which case the resulting formula is called the complete formula. In Example 3 above, we would substitute

$$\sim\psi(x,y,z) \vee \exists w\left(\psi[y,w] \wedge \sim <[x,y,x,w] \wedge \sim <[y,z,w,z]\right) \text{ for } \sim D_b(x,y,z) \ .$$

6. Simple Abstraction

This is a procedure whereby the collapsed (distributive) forms in the distributive sets of P_i and P_j (or the graph of these forms) are mechanically compared and common subformulae of maximum length (i.e., with maximum number of contiguous symbols) are extracted and form derived predicates. Each comparison proceeds from terms of the innermost phrase outward to the remaining phrases.[1] Whichever of P_i or P_j has fewer distinct collapsed forms is chosen, say P_j. Each such form is compared with any collapsed form of P_i and $\sim P_i$. The longest subformulae which are common to both formulae are extracted and labeled as derived predicates. Permutation of terms within a phrase in order to extract a longer formula is permitted. Also permitted are exchanges of variables in primitives where the extension remains unchanged thereby (e.g., $<(x,y,z,w) \equiv <(y,x,z,w)$). Substitutions of equivalent primitives may also be used, e.g., $\sim <(x,y,x,z) = <(x,z,x,y) \vee = (x,y,x,z)$. When comparison is made, if a distributive (collapsed) form deviates from nested (collapsed) form because a quantifier has been distributed through brackets, an additional comparison is made after nested form has been restored (see Example 5 below).

Notice that the use of collapsed forms guards agains excessive computation by not manipulating formulae for existing derived predicates (which resulted from simple abstraction previously performed).

Example 5

Here we compare (the distributed form of) a formula, $P_9(A)$, which is strongly satisfied by the union of all boundary points of dark circular regions (i.e., those regions which satisfy derived predicate $D_{11}(A) \equiv \exists x \exists y \exists z\left(\psi[x,y] \wedge [\epsilon(z,A) \longleftrightarrow <(x,z,x,y)]\right)$ with feature, $P_{10}(A)$, of Example 4 (for convenience $\psi(x,y,z)$ and $\sim\psi(x,y,z)$ will not be decompressed):

$$P_9(A) \equiv \forall x\Big[\sim\epsilon(x,A) \vee \exists B\Big(\epsilon[x,B] \wedge D_{11}[B] \wedge D_d[B] \wedge \exists y\ \big[\sim\epsilon(y,B) \wedge \psi(x,y) \wedge \forall z\ \big(\sim\psi[x,y,z] \vee \exists w[\psi(z,w) \wedge \sim <(x,z,x,w) \wedge \sim <(y,z,y,w]\big)\big]\Big)\Big]$$

$$P_{10}(A) \equiv \exists x\Big[\epsilon(x,A) \wedge \exists z\Big(\epsilon[z,A] \wedge \forall y\ [\sim\epsilon(y,A) \vee \psi(x,y,z)] \wedge \forall y\ \big[\sim\epsilon(y,A) \vee \forall w\big(= [y,w] \vee <[x,y,x,w] \vee <[z,y,z,w]\big)\big]\Big)\Big].$$

---

[1] That is, those strings of symbols (starting from the right of the distributive formulae) which are common to both formulae and which contain no proper substrings common to both are extracted.

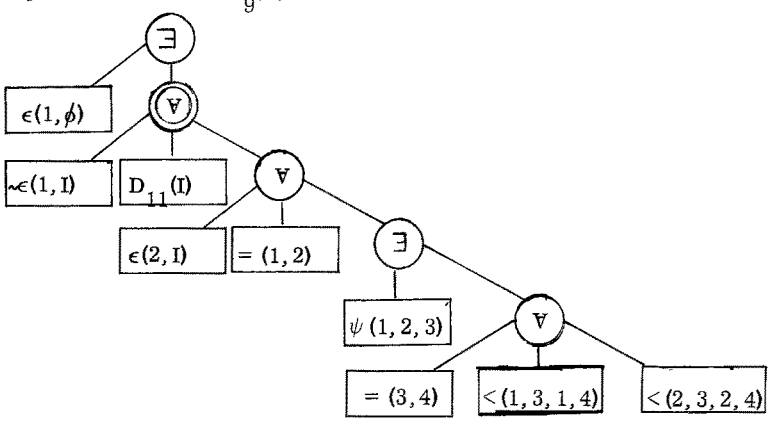
Although $P_{10}(A)$ is in distributed form, it does not conform to nested form Hence we consolidate all terms quantified by "$\forall y$" within the scope of a single such quantifier:

$$P_{10}(A) \equiv \exists x\Big[\epsilon(x,A) \wedge \exists z\Big(\epsilon[z,A] \wedge \forall y\big[\sim\epsilon(y,A) \vee \big(\psi[x,y,z] \wedge \forall w[=(y,w) \vee <(x,y,x,w) \vee <(z,y,z,w)]\big)\big]\Big)\Big]$$

$P_9(A)$ has another distributive form, but we here take the negation of the above form and label our variables so that their order in the innermost phrase is similar to the order of variables in the innermost phrase of $P_{10}(A)$:

$$\sim P_9(A) \equiv \exists x\Big[\epsilon(x,A) \wedge \forall B\Big(\sim\epsilon[x,B] \vee \sim D_{11}[B] \vee D_d[B] \vee \forall z\big[\epsilon(z,B) \vee = (x,z) \vee \exists y\big(\psi[x,z,y] \wedge \forall w[=(y,w) \vee <(x,y,x,w) \vee <(z,y,z,w)]\big)\big]\Big)\Big]$$

A graphical representation of $\sim P_9(A)$ is



Comparison (from right to left) yields the common maximum subformula

$$D_b(x,y,z) \equiv \psi(x,y,z) \wedge \forall w[=(y,w) \vee <(x,y,x,w) \vee <(z,y,z,w)]$$

and we see that

$$P_{10}(A) \equiv \exists x \exists z\Big(\epsilon[x,A] \wedge \epsilon[z,A] \wedge \forall y\big[\epsilon(y,A) \longrightarrow D_b(x,y,z)\big]\Big)$$

and if we substitute $D_a(x,y)$ for $\psi(x,y) \wedge \forall z\left(\sim D_b[x,z,y]\right)$ ("x adjacent to y", see Example 1),

$$P_9(A) = \forall x \exists B\Big(\epsilon[x,A] \longrightarrow \epsilon[x,B] \wedge D_{11}[B] \wedge D_d[B] \wedge \exists y\big[\sim\epsilon(y,B) \wedge D_a(x,y)\big]\Big)$$

This procedure is more easily seen by extracting common subtrees of the graphs of $\sim P_9(A)$ and $P_{10}(A)$ (see previous section).[1]

[1] Although not done in the diagrams above, for mechanized execution of the abstraction procedure, variables should be indexed in reverse order of quantification (i.e., starting from the bottom of the tree).

Comparison of even the simplest formulae often yields derived formulae. See Rothenberg (1973b) for examples.

In essence, the formulas obtained by abstraction (and by "relation abstraction", to be discussed) are those "properties" of features we consider in this system.

7. The Predicate Hierarchy

Complete abstraction is the application of the techniques of simple abstraction to complete formulae (i.e., formulae which have been decompressed till only primitives remain—see Section 5). This is performed at critical points in the flow of the system, such as when computer memory is becoming exhausted. Then all derived predicates retained by the system are eliminated and complete abstraction is performed with all pairs of features in the descriptive basis (i.e., "properties" of features are found). The derived predicates obtained are retained by the system (for purposes to be described in Section 15) and each pair of these is again compared to extract maximal common subformulae which again become derived predicates (i.e., "properties of properties" of features are found). This procedure continues until only primitives remain, and the resulting set of derived predicates is called the predicate hierarchy. In this manner an economical representation of the structure of the descriptive basis may be obtained.

8. Relation Abstraction

Here we compare two formulae (again in compressed form), which although they possess no common subformulae, have identical logical form (i.e., identical relations between their subformulae), or which have subformulae of identical logical form. Consider

$$P_{15}(A) \equiv \forall x \forall y \left(\epsilon [x, A] \wedge D_a [x, y] \longrightarrow \epsilon [y, A] \right)$$

which expresses closure under the relation of adjacency, and

$$P_{16}(A) \equiv \forall x \forall y \left(\epsilon [x, A] \wedge D_5 [x, y] \longrightarrow \epsilon [y, A] \right)$$

which defines a region closed under another binary relation, $D_5(x, y)$. When relation abstraction is here performed a dummy predicate $K(x, y)$, replaces the differing binary predicates in the two formulae, and a predicate form results:

$$S\left(A, K[x, y]\right) = \forall x \forall y \left(\epsilon [x, A] \wedge K [x, y] \longrightarrow \epsilon [y, A] \right)$$

This variable predicate expresses closure with respect to some unspecified binary relation $K[x, y]$.

The procedure for comparing P_i and P_j is as follows: Compress whichever of P_i and P_j has the most variables until both formulae contain the same number of variables.[1]

[1] See Example 4 for illustration of how the number of variables is reduced by compression.

If the substitution of k (at first k = 1) distinct dummy predicates in each of P_i and P_j makes them identical, such resulting formula is a predicate form and is retained by the system. (This compression makes use of the negation of a formula and the other devices used in simple abstraction.) When such a substitution does not render P_i and P_j identical, both formulae are compressed one layer and the comparison procedure is repeated.[1]

Subformulae of a phrase may be permuted or combined, and a dummy predicate may be inserted where no subformula is present if needed to avoid combining two subformulae (one of which is in the other formula) into a single formula. (That is, if $P_i(x,y) \wedge P_j(y,z) \wedge P_k(y,z,w)$ is part of a phrase being compared with $P_i(x,y) \wedge P_k(y,z,w)$, the latter becomes $P_i(x,y) \wedge K(y,z) \wedge P_k(y,z,w)$ and $K(y,z)$ is also substituted for $P_j(y,z)$ in the former.)

Example 6

Compare a predicate which is strongly satisfied by the boundaries of all dark convex sets

$$P_{17}(A) \equiv \forall x \left(\epsilon(x,A) \longrightarrow \exists B \left[\epsilon(x,B) \wedge P_2(B) \wedge \exists y \left(\sim\epsilon(y,B) \wedge D_a(x,y)\right)\right]\right)$$

with a predicate which defines the boundaries of all dark sets with property $P_3(A)$:

$$P_{18}(A) \equiv \forall x \left(\epsilon(x,A) \longrightarrow \exists B \left[\epsilon(x,B) \wedge \forall x \left(\epsilon(x,B) \longrightarrow \bar{D}(x)\right) \wedge P_3(B) \wedge \exists y \left(\sim\epsilon(y,B) \wedge D_a(x,y)\right)\right]\right).$$

Putting both in distributive normal form:

$$P_{17}(A) \equiv \forall x \left(\sim\epsilon(x,A) \vee \exists B \left[\epsilon(x,B) \wedge P_2(B) \wedge \exists y \left(\sim\epsilon(y,B) \wedge D_a(x,y)\right)\right]\right)$$

$$P_{18}(A) \equiv \forall x \left(\sim\epsilon(x,A) \vee \exists B \left[\epsilon(x,B) \wedge \underline{\forall z \left(\sim\epsilon(z,B) \vee \bar{D}(z)\right)} \wedge P_3(B) \wedge \exists y \left(\sim\epsilon(y,B) \wedge D_a(x,y)\right)\right]\right)$$

$P_{17}(A)$ has fewer variables than $P_{18}(A)$. Hence compression of $P_{18}(A)$ (indicated by bracket under above formula) yields:

$$P_{18}(A) \equiv \forall x \left(\sim\epsilon(x,A) \vee \exists B \left[\epsilon(x,B) \wedge \underline{D_1(B) \wedge P_3(B)} \wedge \exists y \left(\sim\epsilon(y,B) \wedge D_a(x,y)\right)\right]\right)$$

The next to innermost phrase above has a greater number of subformulae than the next to innermost phrase of $P_{17}(A)$. Hence subformulae of $P_{18}(A)$ are combined (as indicated by bracket under above formula) to yield:

$$P_{18}(A) \equiv \forall x \left(\sim\epsilon(x,A) \vee \exists B \left[\epsilon(x,B) \wedge D_2(B) \wedge \exists y \left(\sim\epsilon(y,B) \wedge D_a(x,y)\right)\right]\right).$$

---

[1] The entire procedure may be repeated for $k = 1, 2, \ldots, n$ (n is chosen to conform with the availability of computer memory and to avoid excessive iteration. Hence it will usually be small.)

It is now easy to see that comparison of the compressed sets for $P_{17}(A)$ and $P_{18}(A)$ above will yield the following predicate form:

$$S_2(A, K(B)) \equiv \forall x\left(\sim\epsilon(x, A) \vee \exists B\left[\epsilon(x, B) \wedge K(B) \wedge \exists y\left(\sim\epsilon(y, B) \wedge D_a(x, y)\right)\right]\right)$$

which defines the boundaries of all regions, B, which satisfy K(B).

Again, the above procedure can be performed by comparison of the graphs of $P_{17}(A)$ and $P_{18}(A)$ for similarity of structure (exclusive of the different atomic formulae).

Relation abstraction is executed whenever simple abstraction is, and predicate forms are also retained by the system[1] (for uses to be described in Section 15). When complete abstraction is performed and the predicate hierarchy is restructured, relation abstraction is also performed on complete formulae and, in similar fashion, a hierarchy of predicate forms (called the relational hierarchy) is constructed.[2] At each successive level in this hierarchy the predicate forms acquire additional dummy variables. None are retained with more than n such dummy variables.

9. An Elimination Procedure for the Set Membership Primitive

For our purposes we define a feature, P(A) as a tautology iff P is satisfied by all subsets of the data space or only by the entire data space. P(A) is defined as a contradiction iff P is satisfied by no subset of the data space.

Now we eliminate vacant predicates and tautologies which derive from the use of the primitive, "$\epsilon(x, A)$". Contradictions and tautologies are characterized by: $\forall A\left(P[A]\right)$ and $\sim\exists A\left(P[A]\right)$. These formulae have a similar significance: (a) $\forall x\left(\epsilon[x, A]\right)$ and (b) $\sim\exists x\left(\epsilon[x, A]\right)$ or $\forall x\left(\sim\epsilon[x, A]\right)$. (We are not interested in the empty set or its complement.) Hence the following partial decision procedure: (1) Examine a formula and its negation (both in complete and distributive form). (2) If a term of form (a) (or (b)) occurs, label it "T" (or "F"). (3) If "T" (or "F") occurs as part of a conjunction or a quantified conjunction, label the conjunction "T" (or "F"). (4) If the entire formula becomes labelled "T" (or "F"), the formula is tautologous (or contradictory). The reverse is true for its negation. Hence,

$$P_{19}(A) \equiv \forall x \forall y \exists z\left(\epsilon[x, A] \wedge \epsilon[y, A] \wedge D_b[x, z, y] \longrightarrow \epsilon[z, A]\right)$$

which has the following distributive form of its negation

$$\sim P_{19}(A) = \exists x\left(\epsilon[x, A] \wedge \exists y\left[\epsilon(y, A) \wedge \forall z\left(D_b[x, z, y]\right) \wedge \underline{\forall z\left(\sim\epsilon[z, A]\right)}\right]\right)$$

is tautologous (examine its last term) and will not be generated by the system.

---

[1] All nested forms of a predicate form are retained by the system.

[2] Actually, relation abstraction is also performed on derived predicates as well as features.

Similar procedures can often be developed to eliminate tautologies and and contradictions resulting from the use of other primitives. (The above is, in essence, the familiar technique of "quantifier elimination".) This of course, depends upon the choice of primitives in the particular application.

Other peculiarities of the system permit the elimination of other useless formulae. Notice that because all objects are maximal regions, if

$$\left[\exists x_1, \ldots, \exists x_n \forall y \left(\epsilon [y, A] \longrightarrow D[x_1, \ldots, x_n, y]\right)\right]$$

is a data independent feature, an object which strongly satisfies it also satisfies

$$\left[\exists x_1, \ldots, \exists x_n \forall y \left(D[x_1, \ldots, x_n, y] \longrightarrow \epsilon [y, A]\right)\right]$$

Notice that if a formula for a feature is used as part of another feature it becomes a derived predicate and the maximality condition no longer applies. In such case "$\longleftrightarrow$" must be substituted for "$\longrightarrow$" in the first of the above formulae if its meaning is to remain unchanged.

Consider a feature with a collapsed form

$$P(A) \equiv \forall x \left[\epsilon(x, A) \vee D_F(x)\right] \equiv \forall x \left[\sim\epsilon(x, A) \longrightarrow D_F(x)\right] \tag{9.1}$$

where $D_F(x)$ contains no nested phrase of the form

$$\forall y \left[\sim\epsilon(y, A) \vee D_B(y)\right] \quad \text{or} \quad \exists y \left[\epsilon(y, A) \wedge D_C(y)\right] \quad . \tag{9.2}$$

$D_F(x)$ can be satisfied and still $x \epsilon A$. Hence P(A) will be strongly satisfied only by the entire data space. Also, if the collapsed form of the feature is $P(A) \equiv \exists x \left[\sim\epsilon(x, A) \wedge D_E(x)\right]$ its negation will be of form (9.1). Again, $D_E(x)$ can be satisfied and still $x \epsilon A$. Hence P(A) will be strongly satisfied only be sets consisting of the entire data space minus one point. Features which are strongly satisfied only by sets with a fixed number of points are of no interest here (except when "card(A)" is a primitive). Hence we insist that all formulae contain at least one phrase of form (9.2). [1]

Other elimination methods related to this discussion are described in Rothenberg (1973b).

---

[1] The formulae in (9.2) intuitively specify conditions for membership of a point in A (i.e., $\forall y \left[\epsilon(y, A) \longrightarrow D_B(y)\right]$) or for non-membership of a point in A (i.e., $\forall y \left[\sim D_E(y) \longrightarrow \epsilon(y, A)\right]$). The existentially quantified formula in (9.2) may result from a negation of the universally quantified formula.

10. Models

A formula is evaluated in the product set over which the vector of its free variables ranges. Thus if $S \subset U$ is temporarily our universal set, P(A) is evaluated in 2^S (i.e., the power set of S)[1] and D(x,y,A,B) is evaluated in $SXSX2^SX2^S$. (Note that formulae are never evaluated in the entire data space, U (to be discussed).)

A model of the formula, F_i, is an element of the product space (denoted by $\bar{S}$), over which the vector of its free variables ranges, such that the substitution of each component of that element for the free variable satisfies the formula. The model is denoted by $M_j^{(\bar{S})}(F_i)$ where j indexes all such models (the ordering is arbitrary). When F_i functions as a feature, it is denoted by P_i and a model of it $M_j^{(\bar{S})}(P_i)$ (as distinct from a "realization", $M_j^{(\bar{S})}(F_i)$) must strongly satisfy P_i in $\bar{S} = 2^S$ (see Section 2). $M^{(\bar{S})}(F_i)$ is the set of all models of a formula (called its extension) and $M^{(\bar{S})}(P_i)$ denotes the extension of feature, P_i.

The ease of satisfaction, $E(F_i)$, of a formula F_i, is defined as[2]

$$E(F_i) = \frac{\text{card}\left(\overline{M^{(\bar{S})}(F_i)}\right)}{\text{card}(\bar{S})}$$

("card" denotes the number of elements in the set within brackets).

The ease of satisfaction of a feature, P_i, is always defined in terms of satisfaction, not strong satisfaction (i.e., $E(P_i) = E(F_i)$).

The overlap of a pair of formulae, F_i and F_j, is defined as $E(F_i \wedge F_j)$.

The following applies only to features: $\left|M_j^{(\bar{S})}(P_i)\right|$ is the set of points of S in the j^{th} model of P_i in $\bar{S} = 2^S$. Let $n(P_i) = \text{card}\left(\overline{M^{(\bar{S})}(P_i)}\right)$. Then we define

$$\left(M_*^{(\bar{S})}(P_i)\right) = \bigcup_{j=1}^{n(P_i)} \left|M_j^{(\bar{S})}(P_i)\right| .$$

Restating our problem, we are given two disjoint classes of regions

$$\bar{K} = \left\{R_1, R_2, \ldots, R_N\right\}$$

$$\bar{C} = \left\{R_{N+1}, R_{N+2}, \ldots, R_{N+M}\right\}$$

and we wish to generate an ideal feature, $\bar{P}$, such that (see Section 1)

$$\forall k \exists j \left(\left|M_j^{(\bar{U})}(\bar{P})\right| = R_k\right) \qquad \text{where } R_k \in \bar{K}$$

---

[1] A in P(A) is allowed to range only over subsets of S. Points not in A and sets other than A, however, may lie anywhere in U. (In practice, in most cases, for purposes of computation reduction, points not in S are restricted to the boundary of the complement of S in U.

[2] Note that $E(F_i)$ is generally defined with S = U.

AND

$$\forall \ell \forall j \left(\left| M_j^{(\bar{U})}(\bar{P}) \right| \neq R_\ell \right) \qquad \text{where } R_\ell \in \bar{C}$$

Note that $\bar{K} \subset \overline{M^{(\bar{U})}(\bar{P})}$. When this inclusion is proper, $\bar{P}$ is said to generalize $\bar{K}$. We arbitrarily assign the subscripts of each given region, R_i. Now the subscripts of all $M_j^{(\bar{U})}(P_i)$ are assigned such that

$$(10.1) \qquad \frac{1}{N} \sum_{j=1}^{N} \left(\text{card}\left(R_j \cap \left| M_j^{(\bar{U})}(P_i) \right| \right) / \text{card}(R_j) \right)$$

is maximum for all possible indexings (any such maximal indexing suffices).

11. Measures of Success

Let us denote formula (10.1) as $\phi(P_i, \bar{K})$ when all $R_j \in \bar{K}$ and as $\phi(P_i, \bar{C})$ when all $R_j \in \bar{C}$. Let R_j^c be the complement of R_j in U and let $\bar{K}^c$ be set of R_j^c for all j such that $R_j \in \bar{K}$. Define $\gamma(P_i, \bar{K}) = \phi(P_i, \bar{K}) - \phi(P_i, \bar{K}^c)$ and let

$$\theta(P_i, \bar{K}, \bar{C}) = \gamma(P_i, \bar{K}) - \gamma(P_i, \bar{C})$$

be defined as the measure of success in segmentation. It will measure how well a feature approximates an ideal feature. Another such measure, called the success in partitioning is defined as

$$\xi(P_i, \bar{K}, \bar{C}) = \frac{\text{card}\left(\bar{K} \cap \overline{M^{(\bar{U})}(P_i)} \right)}{\text{card}(\bar{K})} - \frac{\text{card}\left(\bar{C} \cap \overline{M^{(\bar{U})}(P_i)} \right)}{\text{card}(\bar{C})} .$$

Other measures are also utilized (see Rothenberg, 1973b and Section 18).

Whenever feedback occurs, a set of weights, called reinforcements, on all features and derived predicates are accordingly altered. These are increased (or decreased) by adding, for example, $\xi(P_i, \bar{K}, \bar{C}) - .5 + [\theta(P_i, \bar{K}, \bar{C}) - .5]/k$ (where k is an integer)[1] to the weights which are initially set at zero. This is done for all derived predicates[2] and predicate forms which together form a successful (or unsuccessful) feature. Thus derived predicates and predicate forms accumulate reinforcement from a variety of features.

12. Mutations

We generate our features by starting with an arbitrarily generated feature, P_i, and successively altering its syntax by means of mutations which are functions which take one

[1] An optimal choice of k may be estimated by experimentation with the program. Actually any method of weighting $\xi(P_i, \bar{K}, \bar{C})$ more heavily than $\theta(P_i, \bar{K}, \bar{C})$ will suffice.

[2] The negations of derived predicates are also treated as derived predicates and accumulate reinforcement accordingly.

feature into another (i.e., $f_n(P_i) = P_k$). We specify three classes of mutations: restricting (iff $M^{(\bar{U})}(P_k) \subsetneq M^{(\bar{U})}(P_i)$), relaxing (iff $M^{(\bar{U})}(P_i) \subsetneq M^{(\bar{U})}(P_k)$), and neutral (otherwise).

Relaxing mutations (said to "relax" a formula) demonstrably include (Rothenberg, 1973b).

(1) Adding a subtree (i.e. derived or atomic predicate) below a node (on the graph of a distributive formula) which contains a universal quantifier.

(2) Eliminating a subtree below a node containing an existential quantifier.

(3) Replacing a universal quantifier by an existential quantifier (and restoring distributive form).

(4) Replacing "$\exists x \forall y$" by "$\forall y \exists x$" and restoring distributive form.

(5) Replacing a subformula (i.e. subtree) by one which it implies.

(6) Replacing "$\forall x \left(\epsilon[x, A] \vee D[x, \ldots]\right)$" by "$\exists x \left(\epsilon[x, A] \wedge D[x, \ldots]\right)$" when the former formula is not vacuous.

(7) Replacing "$\forall x \left(\epsilon[x, A] \vee D[x, \ldots]\right)$" by "$\exists x \left(\sim\epsilon[x, A] \wedge D[x, \ldots]\right)$" when the former formula is not satisfied by the universal set.

The inverses of the above are restricting mutations (said to "restrict" a formula). A neutral mutation consists of the replacement of a subtree on the graph of the distributive form of a feature by another subtree with the same free variables such that neither subtree represents a formula which implies the other. This includes permutations and exchanges of variables in a subformula. Other "composite" mutations are discussed in Rothenberg (1973b).

13. Mutation Choice

Here, among others, we will develop the following criteria: if given region R_j is an element of $\bar{K}$, then (a) when R_j does not satisfy P_i a relaxing mutation on P_i is required, and (b) when R_j satisfies P_i but not strongly a restricting mutation is required. When R_j is an element of $\bar{C}$ and R_j strongly satisfies P_i, either mutation may suffice (although a restricting mutation is usually preferred). These criteria are, of course, averaged for all elements of $\bar{K}$ and $\bar{C}$.

Assuming we have a feature, P_i, we now consider how to choose an appropriate mutation: Let P_i be decomposed into a conjunction of two formulae, P_i^I and P_i^D such that P_i^I is data independent and P_i^D contains data-dependent atomic predicates and only those data independent predicates which are inseparable from this portion of the formula (Rothenberg, 1973b).

(13.1) $$P_i = P_i^I \wedge P_i^D \quad .$$

The features are generated in form (13.1). Suppose we wish region $R_j \in \bar{K}$ to be a model of P_i (note: strong satisfaction is required). Now let $S_j = 2^{R_j}$.[1] Let "p[A]" represent the probability of event A. It has been proved (Rothenberg, 1973b) that if R_j is a randomly selected region,

$$(13.2) \qquad E(P_i^I) \leqslant E(P_k^I) \longrightarrow p\left[\text{card}\left(\overline{M^{(S_j)}(P_i^I)}\right) \geqslant \text{card}\left(\overline{M^{(S_j)}(P_k^I)}\right)\right] > 1/2 \ .$$

This holds only because all elements of $\overline{M^{(S_j)}(P_i)}$ are maximal realizations (i.e., models) of P_i. (13.2) can be intuitively seen if we consider the number of maximal:[2] (1) circles, (2) convex regions and (3) "starlike" regions that may be contained in a region consisting of the interior of an arbitrary ink blot. Clearly the numbers decrease from (1) - (3) while the ease of satisfaction of the formulae (features) corresponding to (1)-(3) increase. Let "$M_L^{(S_j)}(P_i^I)$" denote the largest (i.e. most points) model of P_i^I in S_j. Then the following also holds:

$$(13.3) \qquad E(P_i^I) < E(P_k^I) \longrightarrow p\left[\left|M_L^{(S_j)}(P_i^I)\right| = S_j\right] < p\left[\left|M_L^{(S_j)}(P_k^I)\right| = S_j\right]$$

$$(13.4) \qquad \text{card}\left(\overline{M^{(S_j)}(P_i^I)}\right) > \text{card}\left(\overline{M^{(S_j)}(P_k^I)}\right) \longrightarrow p\left[E(P_i^I) < E(P_k^I)\right] > 1/2$$

$$(13.5) \qquad E(P_i^I) < E(P_k^I) \longrightarrow p\left[\text{card}\left(\left|M_L^{(S_j)}(P_i^I)\right|\right) < \text{card}\left(\left|M_L^{(S_j)}(P_k^I)\right|\right)\right] > 1/2$$

$$(13.6) \qquad \text{card}\left(\left|M_L^{(S_j)}(P_i^I)\right|\right) > \text{card}\left(\left|M_L^{(S_j)}(P_k^I)\right|\right) \longrightarrow p\left[E(P_i^I) > E(P_k^I)\right] > 1/2$$

The above indicates that if more than one object satisfies P_i in S_j or, if the largest model of P_i in S_j is smaller than desired, a relaxing mutation of P_i is appropriate. Now we consider the case where $S_j = 2^{\left|M_j^{(U)}(P_i^D)\right|}$ for all j (i.e., we evaluate P_i^I in the space of all subsets of the j^{th} model of P_i^D).

Suppose for some j, $\left|M_j^{(U)}(P_i^D)\right| \cap R_j \neq R_j$. Then P_i^D will be relaxed until, for all j

$$(13.7) \qquad R_j \subset \left|M_j^{(U)}(P_i^D)\right|$$

and such that

[1] For brevity in notation, when models of features are denoted, the "—" over the super script will be omitted (e.g., $M^{(\bar{U})}(P_i)$ will be denoted by $M^{(U)}(P_i)$).

[2] If we omit "maximal", clearly the ordering is the other way around.

$$\sum_{j=1}^{n(P_i)} \text{card}\left(\left|M_j^{(U)}(P_i^D)\right| - R_j\right)$$

is as small as possible. Note that it is possible that $\text{card}\left\{R_\ell \in \bar{K} \middle| R_\ell \in S_j\right\} > 1$ ("|" means "such that"), i.e., there are several $R_\ell \subset S_j$. Consider P_i and $\bar{P}$ which is ideal feature. Let $M_j^{(U)}(P_i^D) = M_j^{(U)}(\bar{P}^D) = S_j$ (note: $R_j \subset \left|M_j^{(U)}(P_i^D)\right|$). Then it can be shown that (Rothenberg, 1973b):

(13.8) $$\text{card}\left(\overline{M^{(S_j)}(P_i^I)}\right) < \text{card}(R_\ell \in \bar{K} \middle| R_\ell \subset S_j) \longrightarrow p\left[E(P_i^I) > E(\bar{P}^I)\right] > 1/2$$

(13.9) $$\text{card}\left(\left|M_L^{(S_j)}(P_i^I)\right|\right) > \text{card}(R_j) \longrightarrow P\left[E(P_i^I) > E(\bar{P}^I)\right] > 1/2$$

(13.10) $$E(P_i^I) > E(\bar{P}^I) \longrightarrow p\left[\text{card}\left(\left|M_L^{(S_j)}(P_i^I)\right|\right) > \text{card}(R_j)\right] > 1/2$$

That is, if there are two many models of P_i^I relax P_i^I; if there are two many points to a given model of P_i^I, restrict P_i^I.

Suppose $\left|M_j^{(U)}(\bar{P}^D)\right| \neq \left|M_j^{(U)}(P_i^D)\right|$. Since we are here dealing with a size-invariant recognition system[1] (intuitively, a system wherein the magnification of a geometric figures does not alter the features which it satisfies or fails to satisfy), the probable proportion of models in a space S_j does not alter with the size of the space (after a certain minimal size). Hence (13.8) still holds (although (13.9 and (13.10) do <u>not</u>). Hence a comparison of how well (13.8) and (13.9) agree, i.e., the probability that

(13.11) $$\text{card}\left(\overline{M^{(S_j)}(P_i^I)}\right) < \text{card}(R_\ell \in \bar{K} \middle| R_\ell \subset S_j) \longleftrightarrow \text{card}\left(\left|M_L^{(S_j)}(P_i^I)\right|\right) > \text{card}(R_j)$$

indicates how well $\left|M_j^{(U)}(\bar{P}^D)\right|$ and $\left|M_j^{(U)}(P_i^D)\right|$ match. Because of (13.7), $E(\bar{P}^D)$ can only be too large. Hence a failure of (13.11) indicates that P_i^D should be restricted in accordance with (13.9), but without violating (13.7).

When P_i^D is usually composed of one place atomic predicates (i.e., "gray scales"), mutations of P_i^D are simpler than those on P_i^I and are performed first (when there is such a choice).

Other techniques similar to the above exist (Rothenberg, 1973b). Since probabalistic considerations are involved in the above, (13.8) and (13.9) should be averaged over all j (there are N such) and normalized; i.e., examine both

[1] Except at the boundary of U.

$$X = \frac{1}{N}\sum_{j=1}^{N}\left(\left[\text{card}\left(\overline{M^{(S_j)}(P_i^I)}\right) - \text{card}(R_\ell \in \bar{K} \mid R_\ell \subset S_j)\right] / \text{card}(R_\ell \in \bar{K} \mid R_\ell \subset S_j)\right)$$

and

$$Y = \frac{1}{N}\sum_{j=1}^{N}\left(\left[\text{card}\left(\left|M_L^{(S_j)}(P_i^I)\right|\right) - \text{card}(R_j)\right] / \text{card}(R_j)\right)$$

and apply the strategies previously outlined according to whether X and Y are greater than zero.

We have assumed that each R_j is randomly selected from subsets of U. This is not reasonable. However, since models of P_i^I strongly satisfy P_i^I, a region, R_j, which is not a model of P_i^I need not be a model of $\sim P_i^I$. Hence by applying our criteria for mutation choice to both formulae, P_i^I and $\sim P_i^I$ (both of which for this purpose are treated as features), it appears that we may ameliorate our assumption of the randomness of R_j (i.e., "models of P_i^I are as random as models of $\sim P_i^I$").[1] Thus the disagreement of criteria when applied to both P_i^I and $\sim P_i^I$ (e.g., both P_i and $\sim P_i$ require relaxing mutations) indicates the need for a neutral mutation. Notice that both measures of success (Section 11) are reflected in the above. (Details are discussed in Rothenberg (1973b).)

14. Modified Quantifiers

Now a more powerful technique of mutation selection is described; note that all quantified point variables in a formula P(A) range over either the set A (e.g., are restricted by "$\epsilon(x, A)$" within the scope of the quantifier) or the complement A^c of A, or range over the universal set (when no set membership atomic predicate restricts its range). Thus when, on the graph of a formula, a node containing a quantifier has no branch containing a set membership atomic predicate, the quantified variable ranges over the universal set (or its power set if the variable is a set variable). (Note that in subformula $\forall x\left(\sim\epsilon[x, A] \vee F[x, \ldots]\right)$ x ranges over A and in subformula $\forall x\left(\epsilon[x, A] \vee F[x, \ldots]\right)$, x ranges over A^c. However, when the subformula is $\exists x\left(\epsilon[x, A] \wedge F[x, \ldots]\right)$, x ranges over A, etc.)[2].

Let Γ_q denote the set over which variable x_q ranges (where q indexes the variables in a nested form — see Section 3).[3] Since we are dealing with a size invariant system (see

[1] That is, if this randomness with respect to the above criteria does not hold, it appears that in these cases, we will know it because of opposite indications.

[2] Note that

$$\forall x\left(\sim\epsilon[x, A] \vee F[x, \ldots]\right) \equiv \underset{x \in A}{\forall x}\left(F[x, \ldots]\right) \quad \text{and} \quad \exists x\left(\epsilon[x, A] \vee F[x, \ldots]\right) \equiv \underset{x \in A}{\exists x}\left(F[x, \ldots]\right)$$

etc.

[3] Γ_q always is one of A, A^c, U, 2^A, or 2^{A^c}.

previous section) we here consider the fraction of x_q in Γ_q (rather than the fraction of x_q in U) for which a formula is satisfied. (Similar remarks apply to set variables.)[1]

The following notation will be utilized: all distributive forms in the distributive set will be arbitrarily indexed and $P_i^{(k)}$ will indicate the kth such distributive form of feature P_i. It can be shown that an arbitrary indexing of quantifiers in a prenex form from which the distributive set derives (see Section 4) will result in a corresponding ordering of the quantifiers in each distributive form. This follows from the following, which are consequences of the definition of the distributive set and of distributive form (see Section 4).

(a) If, in placing a prenex formula into a distributive form, one quantifier in the prenex formula becomes more than one quantifier (using the same variable) in the distributive form (e.g., $\forall x\left(F_1(x) \wedge F_2(x)\right)$ becomes $\forall x\left(F_1(x)\right) \wedge \forall x\left(F_2(x)\right)$), this occurs in all distributive forms in the distributive set.

(b) If two or more quantifiers using distinct variables in a prenex formula become quantifiers using the same variable in a distributive form (e.g., $\forall x \exists y\left(F_1(x) \wedge F_2(y)\right)$ becomes $\forall x\left(F_1(x)\right) \wedge \exists x\left(F_2(x)\right)$), the number of quantifiers in the prenex form and in the distributive form are equal.

Note that in (a) above the number of quantifiers increases and the number of variables remains unchanged. In (b) the number of variables is reduced, but the number of quantifiers remains unchanged.

Hence all distributive formulae in a distributive set have the same number of quantifiers. All nested formulae (see Section 4) have the same number of quantifiers as the prenex formula (but possibly fewer variables). However, each of the distributive formulae may have more quantifiers than the prenex formula (even when the distributive formulae have fewer variables). Hence two or more quantifiers using the same variable in a distributive formula may have the same index (as derived from the indexing of quantifiers in the prenex formula). Such quantifiers will be abritrarily ordered. That same ordering will be maintained in all distributive formulae, each of whose quantifiers will be subscripted by this ordering.

Let $Q_r^{(k)}$ denote the r^{th} quantifier in $P_i^{(k)}$ (i.e., the k^{th} distributive form of P_i), $x_r^{(k)}$ denote the variable quantified by $Q_r^{(k)}$ and $F_r^{(k)}$ denote the subformula within its scope (i.e., its subtrees on the graph of $P_i^{(k)}$). Simple methods exist (Rothenberg, 1973b) which guarantee that the subscripting of two or more quantifiers of the same variable is consistent across all distributive forms in the sense that, if they have the same subformula within

[1] When $\Gamma_q = U - S_j$; for computational economy, x_q is allowed to range only over points in the boundary of the complement of S. Note that no formula is ever evaluated in U.

their scope, they have the same subscript. Note that each variable, $x_r^{(k)}$, is subscripted according to its quantifier's subscript, and hence the same variable may have different subscripts when separately quantified (i.e., such quantifiers are on the same level of the graph of $P_i^{(k)}$). If $x_p^{(k)}$ and $x_r^{(k)}$ are two such variables we will denote their equivalence by "$x_p^{(k)} = x_r^{(k)}$". (Note that such variables are not separately quantified in nested formulae.) $\Gamma_p^{(k)}$ will denote the set over which $x_p^{(k)}$ ranges (if $x_p^{(k)}$ is not a point variable, $\Gamma_p^{(k)}$ denotes the power set over which $x_p^{(k)}$ ranges).

Let "$Q_p^{(k)} s Q_r^{(k)}$" denote that $Q_p^{(k)}$ is a successor node of $Q_r^{(k)}$ in $P_i^{(k)}$ (i.e., $Q_r^{(k)}$ is within the scope of $Q_p^{(k)}$). "$Q_p = \forall$" and "$Q_p = \exists$" will indicate, respectively, that Q_p is a universal quantifier and that Q_p is an extential quantifier (note that the superscript is unnecessary because of the correspondence in indexing across distributive forms of a formula). Superscripts will henceforth be included only when relevant or useful for purposes of clarity. Given a region, R_q, and a feature, P_i, note that "$R_q = \left|M^{(S_j)}(P_i)\right|$" (subscript is omitted because irrelevant) denotes strong satisfaction of P_i by R_q in space, S_j, and "$R_q = \left|M^{(S_j)}(F_i)\right|$" denotes satisfaction but not necessarily strong satisfaction.[1]

Now consider a feature, $P_i(A)$, and two given regions $R_q \epsilon \bar{K}$ and $R_t \epsilon \bar{C}$. Suppose we wish to alter P_i so that R_q strongly satisfies the altered feature and R_t does not.[2] From the preceding section we see that, in general:

(a) if $R_q \neq \left|M^{(S_j)}(F_i)\right|$ (i.e., R_q fails to satisfy P_i)

a relaxing mutation is required,

(b) If $R_q = \left|M^{(S_j)}(F_i)\right|$ but $R_q \neq \left|M^{(S_j)}(P_i)\right|$ (i.e., R_q satisfies P_i but not strongly),

a restricting mutation is required,

(c) if $R_t = \left|M^{(S_j)}(P_i)\right|$

either a neutral mutation or a mutation according to (a) or (b) above is required. (Note: $R_t \epsilon \bar{C}$ — to be discussed in Section 15.)

Combinations of (a), (b) and (c) will also be discussed in the next section.

---

[1] Note that it is here assumed that $R_q \subset S_j$.

[2] As in the previous section, P_i^I and P_i^D are independently mutated (although this is not explicitly noted here).

First we consider case (a) above Let $A = R_q$ (A is the free set variable in $P_i(A)$). Note that A is no longer a free variable Arbitrarily select a distributive form, $P_i^{(k)}$. Now select p such that $Q_p = \forall$ and such that for all quantifiers $Q_r^{(\ell)}$ (in all distributive forms, $P_i^{(\ell)}$

$$\sum_{\ell} \left(\text{card}(Q_r^{(\ell)}) \,\middle|\, (Q_r^{(\ell)} \text{ s } Q_p^{(\ell)}) \wedge (Q_r^{(\ell)} = \forall) \right) \tag{14.1}$$

is maximum (i.e., such that Q_p has the maximum number of successor nodes which are universal quantifiers, summed over all distributive forms in the distributive set of P_i). If two or more quantifiers are equivalent in this respect, an arbitrary selection will suffice.

Now replace Q_p by the modified quantifier, $\bar{Q}_p$, which computes $\bar{K}_p$, which is defined as that fraction of the x_p's in Γ_p for which P_i is true (i.e., for which $R_q = \left| M^{(S_j)}(F_i) \right|$; which definition is clarified by the following procedure: every subformula of P_i is of the form $Q_p x_p \left(F_p [x_1, \ldots, x_p] \right)$ (although it may be necessary to re-index the quantifiers in the distributive form so that only variables in F_p are indexed—see Section 4). Let each element of Γ_n be denoted by $c_n(k_n)$ where k_n indexes all elements of Γ_n. Define $K\left(c_1(k_1), c_2(k_2), \ldots, c_{p-1}(k_{p-1})\right)$ as the fraction of substitutions of the elements of Γ_p for x_p for which $P_i(A)$ is true when $(x_p, \ldots, x_{p-1}) = \left(c_1(k_1), \ldots, c_{p-1}(k_{p-1})\right)$. For all $n \leq p-1$, if $Q_n = \forall$, define $K\left(c_1(k_1), \ldots, c_{n-1}(k_{n-1})\right) = \min_{k_n} \left(K\left(c_1(k_1), \ldots, c_n(k_n)\right) \right)$; if $Q_n = \exists$, then $K\left(c_1(k_1), \ldots, c_{n-1}(k_{n-1})\right) = \max_{k_n} \left(K\left(c_1(k_1), \ldots, c_n(k_n)\right) \right)$. When n = 1, by definition, $\bar{K}_p = K$. However, if when $\bar{K}_p = 0$, R_q still fails to satisfy P_i, by definition $\bar{K}_p = -\infty$.[1] Note that $\bar{K}_p$ is the same regardless of the distributive form chosen (i.e., regardless of the k in $F_p^{(k)}$). Actually, the procedure will operate on a prenex form (it is best to use nested form).

Now we replace Q_p by another modified quantifier, $\hat{Q}_p^{(k)}$, which similarly computes $\hat{K}_p^{(k)}$, which is defined as that fraction of the $x_p^{(k)}$'s in $\Gamma_p^{(k)}$ for which $F^{(k)p}$ is satisfied, averaged over all substitutions of constants for the free variables in $F^{(k)p}$. Note that the distributive form chosen is now relevant and the superscript of $\hat{K}_p^{(k)}$ is necessary. (Superscripts of $x_p^{(k)}$, $\Gamma_p^{(k)}$ and $Q_p^{(k)}$ are included for clarity.) That is, $\bar{K}_p^{(k)}$ is such that feature P_i is satisfied and $\hat{K}_p^{(k)}$ is such that subformula $F_p^{(k)}$ is satisfied. Hence always $\hat{K}_p^{(k)} \geq 0$.

Note that $\bar{Q}_p$ and $\hat{Q}_p^{(k)}$ are correctly interpreted as "for $\bar{K}_p \cdot 100$ percent of x_p" and as "for $\hat{K}_p^{(k)} \cdot 100$ percent of $x_p^{(k)}$", respectively (strictly speaking, the superscript of $x_p^{(k)}$ is unnecessary). Also note that $E(P_i)$ and $E(F_p^{(k)})$ are increased by replacing Q_p by $\bar{Q}_p$ and $\hat{Q}_p^{(k)}$, respectively. $\bar{K}_p$ and $\hat{K}_p^{(k)}$ are called the values of their respective quantifiers ($\bar{Q}_p$ and $\hat{Q}_p^{(k)}$).

[1] This can be built into the above algorithm.

Now define $\bar{D}_p = \bar{K}_p - 1$. Restore the original quantifier, Q_p, and select another universal quantifier, Q_r, according to condition (14.1). Again, replace Q_r by $\bar{Q}_r$ and $\hat{Q}_r^{(k)}$ and compute the corresponding values of $\bar{K}_r$, $\hat{K}_r^{(k)}$ and $\bar{D}_r$. Restore Q_r and repeat the procedure till all universal quantifiers in $P_i^{(k)}$ are exhausted. (This can be done in a single computation (Rothenberg, 1973b).)

Suppose for some p, $\bar{K}_p = -\infty$. Using $\bar{Q}_p$ is defined as equivalent to replacing $F_p^{(k)}$ by a tautology. A value of "1" in the numerator of $\bar{K}_p$ (i.e., $\bar{K} = 1/\mathrm{card}(\Gamma_p)$) is equivalent to replacing Q_p by an existential quantifier. When $\bar{K}_p = -\infty$ we may conclude that no relaxing mutation within $F_p^{(k)}$ will result in $R_q = \left|M^{(S_j)}(F_i)\right|$ (or, of course, in $R_q = \left|M^{(S_j)}(P_i)\right|$). (Hence modifications of quantifiers which are successors of Q_p on any tree of any distributive form in the distributive set may be avoided.) Note, however, that replacing a universal by an existential quantifier may increase $E(F_p^{(k)})$ (and hence $E(F_i)$) less than a mutation which adds a subformula within $F_p^{(k)}$. Therefore when $\bar{K}_p = 0$ (i.e. $-\infty < \bar{K}_p < 1/\mathrm{card}(\Gamma_p)$) we may not without some hazard avoid considering the modification of $Q_r \left| \exists \ell (Q_r^{(\ell)} s Q_p^{(\ell)}) \right.$.

Now consider case (b) where R_q satisfies P_i but not strongly. The procedure is similar: Again, arbitrarily select a distributive form, $P_i^{(k)}$ and let $A = R_q$. Select p such that $Q_p = \exists$ and such that (similarly to (14.1))

$$\sum_{\ell} \left(\mathrm{card}\ Q_r^{(\ell)} \left| (Q_r^{(\ell)} s Q_p^{(\ell)}) \wedge (Q_r^{(\ell)} = \exists) \right. \right) \tag{14.2}$$

is maximum.

Replace Q_p by $\bar{Q}_p$, and compute $\bar{K}_p$ which is defined as previously except that if, for all k, no replacement of $F_p^{(k)}$ by a contradiction (i.e., a formula which is always false) reqults in $R_q \neq M(F_i)$,[1] by definition, $\bar{K}_p = \infty$. We then replace Q_p by $\bar{\bar{Q}}_p$ which computes $\bar{\bar{K}}_p$ which is defined similarly to $\bar{K}_p$ except that we now insist that $R_q = M(P_i)$ (strong satisfaction) instead of $R_q = M(F_i)$ (satisfaction but not strong satisfaction). Note that $\bar{K}_p$ and $\bar{\bar{K}}_p$ are not necessarily equal when $K_p = \infty$.

Now we again replace Q_p by the modified quantifier $\hat{Q}_p^{(k)}$ which computes $\hat{K}_p^{(k)}$ which is defined (as previously) as the fraction of the $x_p^{(k)}$'s in $\Gamma_p^{(k)}$ for which $F_p^{(k)}$ is satisfied, averaged as before $\left(1/\mathrm{card}(\Gamma_p^{(k)} \leq \hat{K}_p^{(k)} \leq 1)\right)$. Note that $E(P_i)$ and $E(F_p)$ are decreased by replacing Q_p by the modified quantifiers (here, always, $\bar{K}_p \geq 1/\mathrm{card}(\Gamma_p)$).

Now define $\bar{D}_p = \bar{K}_p$. Restore the original quantifier, Q_p, select another existential quantifier according to (14.2) and repeat the above procedure until all existential quantifiers in $P_i^{(k)}$ are exhausted. (Again this can be done in a single computation (Rothenberg, 1973b).)

---

[1] The superscript of "$M^{(S_j)}(F_i)$" is henceforth omitted in such equations when $R_q \subset S_j$.

Note that when $\bar{K}_p = \infty$ or $\bar{\bar{K}}_p = \infty$, we may conclude that no restricting mutation within $F_p^{(k)}$ for any k (i.e., any distributive form) will result in $R_q = M(F_i)$ or $R_q = M(P_i)$ respectively. When $\bar{K}_p = 1$ (or $\bar{\bar{K}}_p = 1$ or $\hat{K}_p^{(k)} = 1$) we are, in effect, replacing an existential by a universal quantifier. Note, however, that replacing an existential by a universal quantifier may decrease $E(F_p^{(k)})$ less than a mutation which adds a subformula within $F_p^{(k)}$. Hence, in this case, we may not without some hazard avoid considering the modification of $Q_r \mid \exists \ell (Q_r^{(\ell)} s Q_p^{(\ell)})$.

With appropriate modifications the above procedure will operate on quantifiers of set (as well as point) variables.

We now define the degree of satisfaction, $D(P_i, R_q)$, of R_q with respect to feature, P_i, as

$$D(P_i, R_q) = \max_p (\bar{D}_p) \mid A = R_q \tag{14.3}$$

("A" is the free variable in "$P_i(A)$")

When $D(P_i, R_q)$ is negative (universal quantifiers have been modified because $R_q \neq M(F_i)$), $D(P_i, R_q)$ is called the degree of undersatisfaction of R_q with respect to P_i. When $D(P_i, R_q)$ is positive (existential quantifiers have been modified because $R_q = M(F_i)$) but $R_q \neq M(P_i)$, $D(P_i, R_q)$ is called the degree of oversatisfaction of R_q with respect to P_i.[1]

In our probelm here we wish to generate a feature, P_i, such that $R_k \in \bar{K} \longrightarrow R_k = M(P_i)$ and $R_\ell \in \bar{C} \longrightarrow R_\ell \neq M(P_i)$. Accordingly we define the satisfaction gap, $G(P_i)$, of P_i as

$$G(P_i) = \min_{R_k \in \bar{K}} \left(D(P_i, R_k)\right) - \max_{R_\ell \in \bar{C}} \left(D(P_i, R_\ell)\right) \quad . \tag{14.4}$$

Note the correspondence between $G(P_i)$ and our "measures of success" (Section 11). Although we are seeking strong satisfaction while $G(P_i)$ deals with satisfaction which is not necessarily strong (appropriate adjustements can be made in the definition of $G(P_i)$), it will be seen that $G(P_i)$ has uses other than measuring the success of P_i — see Section 17. Note also that modified quantifiers of the $F_p^{(k)}$ in P_i correspond to mutations of such subformulas both in kinds and amounts of alterations in ease of satisfaction, $E(P_i)$ and $E(F_p^{(k)})$. Also, the alterations in formulae due to quantifier modification are related by implication just as are the alterations due to mutations.

---

[1] Note the intuitive interpretation of these notions—e.g., of two concave (i.e., non-convex) regions (e.g., ⊃ and ◡), one may be "more concave" (the former) than the other. Experimentation will reveal that the above techniques preserve such intuition (e.g., ⊃ and ⊙ are approximately "equally concave")—see Rothenberg (1973b).

15. Ease of Satisfaction; Mutation

It is easily seen that, when $\hat{K}_p^{(k)}$ is evaluated for all $R_i \epsilon \bar{K}$ and all $R_j \epsilon \bar{C}$ and $\overset{*}{\hat{K}}_p^{(k)}$ is defined as an average of all such $\hat{K}_p^{(k)}$,

$$E(F_p^{(k)}) \approx \overset{*}{\hat{K}}_p^{(k)} . \qquad (\text{"}\approx\text{" means "approximately equal to"})$$

"$\approx$" is used instead of "=" in the above because the probability of such equivalence depends upon and sharply increases as the number of regions (R_i and R_j) for which $\hat{K}_p^{(k)}$ is evaluated and averaged increases.

All modified quantifiers are estimated in a single evaluation of P_i by considering all substitutions for the variables in the formula (regardless of whether the variables are quantified by universal or existential quantifiers). Such methods (which result in estimates for $E(F_p^{(k)})$ rather than exact values) are essential to reduce computation to feasible proportions. Hence, when P_i is evaluated, we obtain an estimate of the ease of satisfaction $E(F_p^{(k)})$ of all its subformulae, $F_p^{(k)}$. Some such subtrees (on the graph of P_i) become derived predicates after the simple abstraction procedure is applied. When these derived predicates are stored, their available estimated eases of satisfaction are also retained.

Each of the $\bar{K}_p$ (and $\bar{\bar{K}}_p$) are also averaged[1] for all $R_i \epsilon \bar{K}$ (in cases (a) and (b)) or for all $R_j \epsilon \bar{C}$ (in case (c) - to be discussed). Let $\overset{*}{\bar{K}}_p$ (and $\overset{*}{\bar{\bar{K}}}_p$) represent such averages. Then a mutation of subformula $F_p^{(k)}$ (of P_i) which results in another subformula, $\breve{F}_p^{(k)}$, such that $E(\breve{F}_p^{(k)}) = \overset{*}{\bar{K}}_p$ (or $\overset{*}{\bar{\bar{K}}}_p$, if appropriate) may result in transforming P_i into an ideal feature. No other mutation of $F_p^{(k)}$ can achieve this result. Hence mutations are chosen so that new subformulae ($\breve{F}_p^{(k)}$) result where, as closely as possible, $E(\breve{F}_p^{(k)}) = \overset{*}{\bar{K}}_p$. For this purpose, the $F_p^{(k)}$ in P_i are compared with the derived predicates previously extracted and, in general, in much simplified form, the procedure is:

(1) Determine whether a relaxing, restricting or neutral mutation (to be discussed) is required.

(2) Examine that distributive form, $P_i^{(k)}$, of the feature, P_i, to be mutated (usually that feature with the highest measure of success) which, when compared (by the simple abstration procedure) to other features previously generated, yields a derived predicate whose formula is the longest of those of derived predicates so obtained (k is thus chosen to facilitate the comparisons (in the abstraction procedures) of the new feature generated with previously generated features). When we evaluate $P_i(A)$ we thereby compute $\overset{*}{\bar{K}}_p$ and $\overset{*}{\hat{K}}_p^{(k)}$ for all p. At first the collapsed form (see Section 5) of $P_1^{(k)}$ is examined.

[1] More refined methods than averaging are used; see next section.

(3) Compare each subformula, $\breve{F}_p^{(k)}$, which would result from each mutation of the required type (on a corresponding subformula $F_p^{(k)}$ of P_i) with each of the derived predicates which has been stored. Comparison prior to mutation avoids the examination of many possibilities (see Rothenberg, 1973b). Also, of mutations which add subformulae within the scope of a quantifier (see Section 12) only the addition of such subformulae as are included in the list of derived and atomic predicates are considered.

(4) Select that mutation which produces a subformula, $\breve{F}_p^{(k)}$ (of the new feature), which is identical to a derived predicate whose ease of satisfaction most closely approximates that required (i.e., $E(\breve{F}_p^{(k)}) \approx K_p$; see above discussion).[1] If there are several such mutations, select that which results in the most reinforced (i.e. with the highest (reinforcement" — see Section 11) derived predicate. If none such exists, decompress (see Section 5) $P_i^{(k)}$ one layer and try again. If repeated decompression fails, try another distributive form of P_i. If this also fails:

(5) Compare the features to be altered, P_i, with each of the predicate forms (see Section 8) which have been stored. Select a predicate form (if such exists) which resembles P_i (or one of its subformulae) in the sense that, if corresponding subformulae of P_i (which must have the same free variables as the dummy predicate) replace the dummy predicates, a formula results which constitutes a mutation of P_i of the required type. If such resulting formula contains another dummy predicate, replace it by the most reinforced derived predicate (or atomic predicate) with the same free variables — see last part of Example 6 below. If a choice of predicate forms exists, select that with the highest reinforcement. If no such predicate form exists, a mutation of the required type is chosen at random and the footnote below applies.

Criteria restricting such random choice exist. The details (Rothenberg, 1973b) of the above procedures are constructed so that it is extremely unlikely that random choice will be necessary (except extremely rarely) after the initial stages of the learning procedure (note that, almost always, some mutation will better approximate the required ease of satisfaction of a subformula of P_i than another). It is, of course, possible (in some applications) to construct (from certain atomic predicates) inherently two-valued predicates (such as that a set must contain an even number of points) so that quantifier modification cannot guide the mutation procedure. In this case, techniques which develop those of Section 13 must be used and much greater use of random choice results.

---

[1] If the ease of satisfaction of a derived predicate (or subformula resulting from a possible mutation) is not available by the methods described, it may be estimated by valuations of the formula in randomly chosen small spaces (as well as by other estimating techniques— see Rothenberg, 1973b).

Example 6

Suppose we wish to relax $P_{21}(A)$, which asserts that the region substituted for A has a point which is "between"[1] no two other points in the region (i.e., a "cusp" if the region is connected). Let $D_{14}(x,y) \equiv \forall z \left[\sim\epsilon(z,A) \vee \sim D_b(y,x,z)\right]$ and let $D_b(y,x,z)$ (previously defined) be in our list of derived predicates so that the collapsed form of our chosen distributive form of $P_{21}(A)$ is

$$P_{21}(A) = \underset{Q_1}{\exists x}\left[\epsilon(x,A) \vee \underset{Q_2}{\forall y}\left(\sim\epsilon\left[y,A\right] \vee D_{14}\left[x,y\right]\right)\right]$$

Suppose we wish a relaxing mutation and our list of derived predicates contains $D_a(x,y)$ (previously defined) and

$$D_{15}(x) \equiv \forall y \left[\sim\epsilon(y,A) \vee D_a(x,y) \vee D_{14}(x,y)\right]$$

and $E\left(D_{15}(x)\right) \approx \overset{*}{K}_2$ (which corresponds to Q_2 in the distributive form above). Then our mutation (of type 1 — Section 12) yields a new feature,

$$\breve{P}(A) \equiv \exists x\left[\epsilon(x,A) \vee \forall y \left(\sim\epsilon\left[y,A\right] \vee D_a\left[x,y\right] \vee D_{14}\left[x,y\right]\right)\right]$$

Suppose we could not find $D_{15}(x)$ or any other suitable derived predicate. We then decompress $P_{21}(A)$ one layer to obtain

$$P_{21}(A) \equiv \underset{Q_1}{\exists x}\left[\epsilon(x,A) \vee \underset{Q_2}{\forall y}\left(\sim\epsilon\left[y,A\right] \vee \underset{Q_3}{\forall z}\left[\sim\epsilon(z,A) \vee \sim D_b(y,x,z)\right]\right)\right]$$

Suppose we now find among our derived predicates

$$D_{16}(x,y) \equiv \forall z \left[\sim\epsilon(z,A) \vee \sim D_b(y,x,z) \vee D_a(x,z)\right]$$

and $E\left(D_{16}(x,y)\right) \approx \overset{*}{K}_3$. The result of our mutation yields:

$$\breve{P}(A) \equiv \exists x\left[\epsilon(x,A) \vee \forall y \left(\sim\epsilon\left[y,A\right] \vee \forall z \left[\sim\epsilon(z,A) \vee \sim D_b(y,x,z) \vee D_a(x,z)\right]\right)\right]$$

(which formula has the same significance as that resulting from our first mutation). If instead of $D_{16}(x,y)$, we found among our derived predicates

$$D_{17}(x) \equiv \forall y \left(\sim\epsilon\left[y,A\right] \vee D_a\left[x,y\right] \vee \forall z \left[\sim\epsilon(z,A) \vee \sim D_b(y,x,z) \vee D_a(x,z)\right]\right)$$

and $E\left(D_{17}(x)\right) \approx \overset{*}{K}_2$, our mutation of type 5 — Section 12 yields

[1] Use definition of "betweeness" in Example 1, p. 83

$$\breve{P}(A) \equiv \exists x\left[\epsilon(x,A) \vee \forall y\left(\sim\epsilon[y,A] \vee D_a[x,y] \vee \forall z\ [\sim\epsilon(z,A) \vee D_b(y,x,z) \vee D_a(x,z)]\right)\right].$$

Suppose no suitable derived predicate is found after examining decompressed forms of the various distributive forms of P_{21}, but we discover among our predicate forms

$$S_{10}(A, K_1[x,y], K_2[x,y]) = \exists x\left[\epsilon(x,A) \vee \forall y(\sim\epsilon[y,A] \vee K_1[x,y] \vee K_2[x,y])\right] .$$

We first note that $D_{14}(x,y)$ has the same variables as $K_2(x,y)$ and hence $D_{14}(x,y)$ replaces $K_2(x,y)$. Then, if $D_a(x,y)$ is the most reinforced derived predicate with two free variables, $D_a(x,y)$ replaces $K_1(x,y)$ and our mutations will yield the same result as our first mutation in this example. (The reader may find it amusing to find models of the formulae resulting from the above mutations.) Possible mutation operations are too numerous for extensive illustration here — see Rothenberg (1973b).

We now consider case (c) of the previous section where $R_t \epsilon \bar{C}$ strongly satisfies P_i. We wish this <u>not</u> to be the case and hence we consider a relaxing mutation if $R_q \epsilon \bar{K}$ fails to satisfy P_i and a restricting mutation if R_q satisfies P_i. If R_q <u>strongly satisfies</u> P_i both strategies are unlikely to be successful. In all these cases a neutral mutation should be considered. This also applies when both R_q and R_t fail to satisfy P_i. When modified quantifiers are evaluated, in <u>all</u> cases both universal and existential quantifiers are modified and values of $\bar{K}_p$ and $\hat{K}_p^{(k)}$ are computed for all such quantifiers. Suppose a neutral mutation of P_i is appropriate, and within the scope of $Q_p^{(k)}$ (with subformula $F_p^{(k)}$) there is another quantifier, $Q_q^{(k)}$ (with subformula $F_q^{(k)}$) such that Q_p is a universal quantifier and Q_q is existential (or vice versa). Then, if $\bar{K}_p \approx \bar{K}_q$, a neutral mutation of $F_p^{(k)}$ which does not alter the ease of satisfaction of $F_p^{(k)}$ is appropriate. In general, a comparison of $\bar{K}_p$ and $\bar{K}_q$ will indicate the degree to which (if any) and in which way a neutral mutation should alter the ease of satisfaction of $F_p^{(k)}$. The extent to which such a mutation may alter the <u>overlap</u> (see Section 10) of P_i and the feature, $\breve{P}_i$ (which results from the mutation), without disturbing $E(P_i)$ (i.e., such that $E(P_i) \approx E(\breve{P}_i)$) is indicated by $\min(\bar{K}_p, \bar{K}_q)$. In place of a neutral mutation, a "composite mutation" which combines a relaxing and restricting mutation may be used. These will not be discussed here (see Rothenberg, 1973b), but some indications as to the general scheme are shown: The required overlap of P_i and P_i when a neutral or composite mutation is performed may be determined by examination of the measure of success, $\theta(P_i, \bar{K}, \bar{C})$ (see Section 10). Neutral mutations are chosen by comparison with predicate forms (somewhat as in step (5) above): If the order of variables in a dummy predicate in a predicate form (which matches P_i as in step (5)) differs from the order of variables in a corresponding subformula of P_i to be mutated, the order of variables in that subformula

is altered to match those of the dummy predicate. If a predicate form differs from P_i in the identity of a corresponding subformula (to the subformula to be mutated), the subformula of P_i is replaced by that of the predicate form.

Note that when, for some feature, P_i, $F_p^{(k)} \equiv F_q^{(k)} \wedge F_r^{(k)}$, $\hat{K}_p$ will estimate $E(F_q^{(k)} \wedge F_r^{(k)})$ while $\hat{K}_q^{(k)}$ and $\hat{K}_r^{(k)}$ will estimate $E(F_q^{(k)})$ and $E(F_r^{(k)})$ respectively. If $F_q^{(k)}$ and $F_r^{(k)}$ become derived predicates, we will be able to estimate the "overlap" of $F_q^{(k)}$ and $F_r^{(k)}$. This is useful in selecting neutral mutations of a feature which replace a subformula which is identical to $F_q^{(k)}$ by one which is identical to $F_r^{(k)}$ in such fashion as to control the overlap between the original and mutated feature.

In general, we compute a $\bar{K}_p$ and $\bar{\bar{K}}_p$ for each quantifier of a feature and for each $R_i \in \bar{K}$ and $R_j \in \bar{C}$. Hence these values of modified quantifiers should properly be denoted "$\bar{K}_p(R_q)$" and "$\bar{\bar{K}}_p(R_q)$" where R_q represents the region substituted for the free variable, A, in $P_i(A)$ when these values are computed. When selecting a mutation, properties of the distributions of $\bar{K}_p(R_q)$ and $\bar{\bar{K}}_p(R_q)$ over all $R_q \in \bar{K}$ and over all $R_q \in \bar{C}$ are considered (averages are seldom used—their use was mainly for illustrative purposes). Of particular importance is $\min(\bar{K}_p | R_q \in \bar{K})$ and $\max(\bar{K}_p | R_q \in \bar{C})$ when Q_p is universal and $\max(\bar{K} | R_q \in \bar{K})$ and $\min(\bar{K}_p | R_q \in \bar{C})$ when Q_p is existential. We are attempting to make the satisfaction gap, $G(P_i)$, as large as possible. Particular properties of the distributions of $\bar{K}_p(R_q)$ and $\bar{\bar{K}}_p(R_q)$ corollate with our measures of success and with $G(P_i)$. Although the detailed strategy for mutation choice is too detailed for description here (see Rothenberg, 1973b), many of its properties can be deduced by the reader from the preceding discussion. Note also that $\bar{\bar{K}}_p(R_q)$ differs from $\bar{K}_p(R_q)$ in that it is based upon strong satisfaction. The condition for success that $R_q \in \bar{C}$ may satisfy P_i, but not strongly, elucidates the use of $\bar{K}_p(R_q)$ in the mutation choice procedure. Also of use in selecting neutral mutations is that it has been shown (Rothenberg, 1973b) that:

(15.1) A neutral mutation on a compressed formula (for a feature) is more likely to reduce the overlap between the feature and its mutant then a similar mutation on a subformula which is exposed by decompressing the formula one (or more) layers.

16. Initial Features

The system retains three lists (of fixed maximum length) of formulae: (1) features, (2) atomic predicates and derived predicates and (3) predicate forms. Those with lowest reinforcement (except atomic predicates) are eliminated when the computer memory allocation for these lists fill (except when complete abstraction is performed and the predicate hierarchy is rebuilt).

In most applications the trainer will insert any prior knowledge he has of the problem by the direct insertion (at any time in the learning procedure) of derived predicates and/or features.

When no initial features or derived predicates are provided by the trainer, the initial features are generated as follows: an atomic predicate is selected (at random) and is combined with quantifiers and set membership primitives to form formulae which both satisfy the syntactic requirements for features and survive the elimination procedure. If a single primitive does not suffice, a conjunction or disjunction of several is used. As few primitives as possible are used. Using the primitives for Example 1, such initial features might be:

$P_{19}(A) = \exists x \exists y \left(\psi [x, y] \wedge \forall z \left[\sim \epsilon(z, A) \vee < (x, z, x, y) \right] \right)$ (strongly satisfied by circular regions)

$P_{20}(A) = \exists x \exists y \left(\psi [x, y] \wedge \forall z \left[\sim \epsilon(z, A) \vee < (z, x, z, y) \right] \right)$ (strongly satisfied by linear partitions of U)

At the start of the generation procedure $\psi(x, y, \ldots)$ is used for all variables (as above) to prevent contradictory or tautologous features or features with excessively high or low eases of satisfaction.

Consider features which are of the following compressed form wherein D(x) contains no set membership primitives:

$P_1(A) \equiv \forall x \left[\epsilon(x, A) \vee D(x) \right]$ $\quad (\equiv \forall x \left[\sim \epsilon(x, A) \rightarrow D(x) \right])$

$P_2(A) \equiv \forall x \left[\sim \epsilon(x, A) \vee D(x) \right]$ $\quad (\equiv \forall x \left[\epsilon(x, A) \rightarrow D(x) \right])$

$P_3(A) \equiv \exists x \left[\epsilon(x, A) \wedge D(x) \right]$

$P_4(A) \equiv \exists x \left[\sim \epsilon(x, A) \wedge D(x) \right]$

Notice that all realizations of $P_2(A)$ also satisfy $P_1(A)$ because of the strong satisfaction requirement.[1] $P_1(A)$, however, is strongly satisfied by all data spaces. (Note that it may <u>not</u> be satisfied by subsets of a data space and is hence not a tautology.) $P_3(A)$ is strongly satisfied by the null set or an entire data space. $P_4(A)$ is strongly satisfied either by the null set or by an entire data space with one point removed.

Clearly, at the beginning of the feature generation procedure, we wish to avoid features with very high or low ease of satisfaction (so that some measure of success is probable). Hence we choose features with compressed form similar to $P_2(A)$ when generating initial features. More refined restrictions can be deduced (see Rothen berg, 1973b)

---

[1] The above statements may be intuitively illustrated by interpreting "D(x)" as being satisfied only by points on the boundary of the data space.

In general, if a mutation results in a tautology, the tautology is restricted, and if the mutation results in a contradiction, it is relaxed.[1] In initial feature generation, the following order of mutations is used, which approximates an ordering by the probability of decreasing the overlap of a feature and its mutant:

Restricting Mutations

first: add "$\psi(x_1 \ldots x_n)$" to the formula by a conjunction (to the innermost phrase) where $\{x_1 \ldots x_n\}$ includes all bound point variables.

second: add "$\epsilon(x_i, A)$" or "$\sim\epsilon(x_i, A)$" to a phrase by a conjunction. This is one phrase at a time from the innermost phrase outward (i takes on appropriate values) — see example in Section 19.

Relaxing Mutations

first: eliminate "$\psi(x_1 \ldots x_n)$ from the formula.

second: add "$\sim\psi(x_1 \ldots x_n)$" to the innermost phrase of the formula by a disjunction (where $\{x_1 \ldots x_n\}$ includes all bound variables).

third: add "$\epsilon(x_i A)$" or "$\sim\epsilon(x_i, A)$ to a phrase by a disjunction. This is done one phrase at a time, from the innermost phrase outward.

Neutral Mutations

first: negate a subformula (from the innermost phrase outward).

second: permute the variables in a subformula.

Also, in the initial generation of features, a selection weighting function assigns somewhat higher probabilities of selection to mutations which increase the length of formulae. (Such selection weighting function later prevents the generation of formulae with an excessive number of variables.) As soon as more than one successful feature is obtained, simple and relation abstractions are performed.

Note also that initial derived predicates of any given number of free variables (when such are required by the mutation procedure and are scarce) may be generated by combining atomic or derived predicates; e.g., $D_1(x, y, z) \equiv D_a(x, y) \wedge D_a(y, z)$; $D_2(x, y) \equiv \exists z(D_b[x, y, z])$.

17. Topology[2]

Because of the use of quantified variables in features (rather than constants—i.e., the propositional calculus), each feature has many models (i.e., "generalization" — see Section

[1] This applies to tautologies and contradictions which survive the elimination procedure.

[2] The author wishes to express his gratitude to Prof. John Myhill of Leeds University and to Prof. Bernard Jaulin of the University of Paris for their suggestions that a topological approach would prove fruitful.

1, occurs). Functions may exist which map models of the same feature into each other — i.e., the data space must contain certain symmetries. In general, it is desirable that the satisfaction of formulae by regions of the data space be invariant with respect to the transformations of such regions by "translation", "rotation", "reflection", "dilation" (i e., "magnification"), and "contraction" (i.e , "reduction"). It is convenient to impose a topology on the data space so that this will hold and so that the above transformations may be defined. Also, contraction (the inverse of dilation) can vastly reduce the number of points in the data space (and each of its regions). The application of our techniques to such a space with a reduced number of points can result in enormous computation reduction and, in fact, the practicality of the procedures developed rest upon the utilization of such "reduction mappings" (to be defined). The topology described below also relates the finite spaces here employed to the infinite spaces which they approximate and with which model theory customarily deals: (See appendix (p. 124) for the motivation for the definitions below.)

$(S, n_s) \in \Gamma$ (i.e., is a graphic space iff (S = space, n_s = neighborhood system)

(a) S is a V-space (A Frechet space - see Sierpinsky, 1952) with > 1 element

(b) Each point has a minimum neighborhood

(c) Each minimum neighborhood has ≥ 2 points (a consequence of (d))

(d) ϕ and S are the only closed (or open) sets ("ϕ" here denotes the null set)

Let each point be defined as "adjacent" to all points in its minimum neighborhood. This induces a metric; i.e., the graph metric derived from the number of edges in the shortest path (along adjacent points in the graph) from one point to another. Write "$xy <_G zw$", etc. for that metric. (Note that, because of (d) above, S is connected, but not necessarily finite.)

$(S, n_s) \in \hat{\Gamma}$ (is a graphable space) iff

$$(17.1)\quad (\exists P)(\exists n_p)(\exists \phi: n_p \rightarrow n_s)\left[(P, n_p) \in \Gamma\right) \wedge \left(S = U\{\phi N \mid N \in n_p \wedge N \text{ is minimal}\}\right) \wedge (\forall N_i, N_j \in n_p)\left(N_i \subsetneq N_j \rightarrow \phi N_i \subsetneq \phi N_j\right)\Big]^{1}$$

Given an ordering, $\leq_0$, of all elements of S^2 and of P^2, we define (where ϵ is an element of $S \times S$)

---

[1] The latter condition is added in order to preserve the intuitive notion. Also, possibly converge by neighborhoods rather than minimum neighborhoods is sufficient. Note that "P" here denotes a space, not a feature (features are always subscripted, e.g., "P_i").

$$xy - zw \leq_0 \epsilon \equiv xy \leq_0 zw \vee \exists v \left[(xv \leq_0 zw \wedge vy \leq_0 \epsilon) \wedge (xy \leq_0 zv \wedge wv \leq_0 \epsilon)\right]$$

and so[1]

$$xy - zw >_0 \epsilon \equiv xy >_0 zw \wedge \forall v \left[(xv \leq_0 zw \rightarrow vy >_0 \epsilon) \wedge (xy \leq_0 zv \rightarrow wv >_0 \epsilon)\right]$$

$(P, n_p, \leq_0) \in \Gamma_\mu(\epsilon)$ (is a <u>uniform graphic space</u>) iff $(P, n_p) \in \Gamma$ and $(\forall x, y, z, u \in P)$

$$(xy - zu >_0 \epsilon \rightarrow xy >_G zu)$$

(and so $xy \leq_0 zu \wedge xy >_0 zu \rightarrow \exists v \left[(xv \leq_0 zu \wedge vy \leq_0 \epsilon) \vee (xy \leq_0 zv \wedge uv \leq_0 \epsilon)\right]$).

(P, n_p) is a <u>basis</u> for S (given $\leq_0$) iff (where δ is an element of $S \times S$): P, S satisfies (17.1) for some ϕ and

(17.2) $$(\exists \psi: S \rightarrow P)(\exists \delta)(\forall x, y, z, w \in S) \left[(xy - zw >_0 \delta) \rightarrow (\psi x, \psi y) >_G (\psi z, \psi w)\right]$$

(17.3) $\psi: S \rightarrow P \in Rd_\epsilon$ (is an <u>ϵ-reduction map</u>) iff

(a) ψ is continuous (i.e., $\psi(E') \subset \psi(E) \cup \left(\psi(E)\right)'$ for all $E \subset S$, where E' is the derived set of E)[2], and onto

(b) $S \in \Gamma_u(\epsilon)$

(c) P is a basis for S (i.e., 17.2 holds with the same ψ and with $\delta = \epsilon$)

(d) Card P < card S .

The following observations are germane in graphable spaces (<u>all references are to Sierpinsky, 1952</u>):

(a) "x is a limit point of A" means that x is adjacent to an element of A (p. 3). E' (the derived set) consists of all non-isolated points of E, plus the neighboring points of E (p. 3).

(b) Only the whole space is closed or open but relative closure and openness makes sense (pp. 4, 6, 15): $E \subset F$ is closed in F if it is separated from the rest of F by a "white band". Relative open is the same as relative closed.

(c) "Dense in itself" here means no isolated points (p. 13).

---

[1] Note that our definition can be weakened to include the case where $(xy < zw \;\; yx \leq_0 zw)$ (see Rothenberg, 1973b): $xy - zw \leq_0 \epsilon = xy \leq_0 zw \vee \exists v \left[(xy \leq_0 zw \wedge vy \leq_0 \epsilon) \vee (xy \leq_0 zv \wedge wv \leq_0 \epsilon) \vee (vy \leq_0 zw \wedge xv \leq_0 \epsilon) \vee (xy \leq_0 vw \wedge vz \leq_0 \epsilon)\right]$.

[2] Condition (a) may be deducible from the other conditions (see T22, p. 25 of Sierpinsky, 1952, and definition 17.1 here).

(d) "Scattered" means only isolated points (p. 13).

(e) Two sets are "separated" if a white band separates them (p. 16).

(f) "Connected in the topological sense coincides with the geometrical sense (p. 16).

(g) "Frontier" of A is points of A adjacent to points not in A, plus points not in A next to points in A (page 19 - hence corollary on same page).

(h) "Continuous mapping" (from one discrete space to another) means two neighboring points go into neighbors or into the same point.

(i) "Bicontinuous map" (homomorphism) means "x next to y iff f(x) next to f(y) (p. 28).

Reduction maps are continuous maps and all properties preserved under continuous maps are preserved by reduction maps; i.e., connectedness (but not disconnectedness) boundary connected (i.e., simply connected), frontiers are preserved (but boundaries are not), etc.

The transformations of "contraction" and "dilation are defined by reduction maps and their inverses. "Rotation", "translation" and "reflection" are defined (together) by a bi-unique mapping from a space S to a transformed space, P, such that (a), (b) and (c) (but not (d)) of (17.3) are satisfied. The following have been proved (Rothenberg, 1973b) where $\psi: S \rightarrow P \in Rd_{\epsilon}$ and where each feature, P_i, is constructed using the atomic predicate, "$<(x, y, z, w)$":

If $S = \left|M(P_i)\right|$, then $P = \left|M(P_i^*)\right|$ where P_i^* is the same formula as P_i except that its degree of satisfaction has been altered by a bounded amount which is a function, f, of ϵ and of the syntax of P_i — i.e., in general (see (14.1):

(17.4) $$\left|D(P_i, S) - D(P_i, P)\right| \leq f(\epsilon, n)$$ (note again that P is a space, not a feature like P_i)

where n depends upon the number of occurrences of the atomic predicate "$<(x, y, z, w)$" in the subformula of P_i within the scope of the relevant (to $D(P_i, P)$) modified quantifier. Actually f depends upon other factors as well, but minimally so when the preimage of each point $p_i \in P$ is chosen so that ϵ is minimal over all of S^4. (In this case note that if S is homogeneous, the number of points in the preimage of each $p_i \in P$ is determined by ϵ.) When S is, for example, a bounded rectangle in Euclidean space, and P is finite, $f = K/card(P)$ where K is a function of n and specific metric properties of S, and card(P) is a function of ϵ. (Note: as $\epsilon \rightarrow 0$, $f \rightarrow 0$.)

Let S be finite. Suppose $R \subset S$ and $\bar{R}$ is the image of R in P under the above reduction mapping (henceforth called the degraded image of R). Let each $p_j \in \psi(R)$ such

that the preimage of p_j contains elements both of R and S-R be called a dubious point[1] denoted by "$p_j \in d(\bar{R})$". Then

$$\left|D(P_i, R) - D(P_i, \bar{R})\right| \leq g(\epsilon, n, m) \tag{17.5}$$

where $m = \text{card}\left(p_j \,\middle|\, p_j \in d(\bar{R})\right)/\text{card}(p_j \mid p_j \in \bar{R})$ and the above remarks (in (17.4)) about f also apply to g. Larger values of m result in larger values of g. (Note: if R is a randomly chosen connected region in a uniformly sampled place, m is approximately equal to its circumference divided by its area.)

Similar (but tighter) bounds exist when ψ executes a translation, rotation or reflection.

Note the similarity between the left hand portions of the formulae (17.4) and (17.5) and the definition (14.2) of the satisfaction gap of P_i, $G(P_i)$. Hence the following application of (17.4) and (17.5) is appropriate:

Normally the data space, U, is uniformly sampled - along a square grid.[2] The image of U under a reduction map is similar, but is a smaller (i.e., fewer points) space. When $G(P_i) > 0$ we may (from $G(P_i)$) determine an ϵ (and select a corresponding reduction mapping) such that the application of P_i to the degraded regions in P still results in $G(P_i) > 0$ (see remarks following (17.4)). For such ϵ no information relative to the satisfaction of P_i is lost by application of the reduction map. Since $f \approx K/\text{card}(P)$ (see above) and card(P) decreases with ϵ, a reduction map with a small ϵ (i.e., card(P) is large) will reduce $G(P_i)$ by a small amount. However, the amount of calculation reduction that can result from the application of such a reduction mapping is enormous. In addition to uses when $G(P_i) > 0$, the evaluation of P_i in a relatively large space may be avoided by confining such evaluation to regions which are preimages of regions in smaller spaces where models of P_i have been found. (That is, the evaluation of a feature in a smaller space precedes that in a larger, the former with a lower required degree of satisfaction than the latter. Evaluation in the larger space is then performed in that restricted subspace which is the preimage of a model found in the smaller.)

Other methods of computation reduction are used. Regions of the sample space which satisfy (or fail to satisfy) certain predicates have their identifications stored together with the predicate abbreviations. Then "do loops" in formulae range only over those regions which are already known to satisfy subpredicates included within these loops. Further

---

[1] Data dependent unary atomic predicates (e.g., $\bar{D}(x)$ in Example 1) can be included by extending the definition of dubious point to include any $p_i \in \psi(R)$ whose preimage contains elements with differing values of the unary atomic predicate. Then (17.5) still applies.

[2] Note: a sampling along the centers of a covering of U by regular hexagonal tiles would permit a smaller choice of ϵ in a reduction map.

computation reduction is accomplished by the use of implications between features (these are known across all but neutral mutations — see Section 12) to avoid unnecessary testing. A similar technique may be employed to minimize search and storage when data independent primitives are related by implication ("tree searches" are used).

Note also that the data independent portions of features, P_i^I, are always evaluated in either some R_j or $\left|M_j^{(U)}(P_i^D)\right|$ (see Section 13), never in the entire data space. In this manner enormous computations are avoided. Furthermore, P_i is evaluated in S by utilizing the properties of nested normal form to remove points from the space until a model (i.e., strong satisfaction) is encountered. Roughly (see Rothenberg, 1973b), if $S \neq \left|M_j^{(U)}(P_i)\right|$ then $S = \left|M_j^{(U)}(\sim F_j)\right|$. Then points are removed, one at a time so that $\sim P$ is violated as soon as possible.

18. Hierarchy of Representation

Here the size of the problem is reduced (and its representation changed) by the use of hierarchical levels wherein objects are expressed as collections of subobjects, etc., all such objects being defined by features which are similarly hierarchically organized. At each such hierarchical level the techniques previously discussed are independently applied.

All "atomic predicates" ("primitives"), "features", "points", "objects" and "object classes" (see Section 2) will now have "first level" prefixing their names. Also the following restrictions are added to our syntax:

(18.1) (a) For each first level feature, $P_i(A)$, which begins with a string of existentially quantified set variables (e.g., "$\exists B \exists C \exists D, \ldots$"), those set variables $(B, C, D, \ldots)$ are allowed to range only over models of other features already in the descriptive basis (i.e., objects).

(b) Furthermore, the above set variables may not be identical with the region that is substituted for the free variable, A, in $P_i(A)$ above (i.e., $\sim(A = B) \wedge \sim(A = C) \wedge \sim(A = D) \ldots$).

Also the following, called the identification measure is added to our measures of success (see Section 10) and is used when generating features:

$$\bar{\theta}(P_i, \bar{K}, \bar{C}) = \gamma(P_i, \bar{K}) + \gamma(P_i, \bar{C}) \quad . \tag{18.2}$$

Whenever we advance a level (from the n^{th} to the n+1st level) the following operations are performed (see Example 7 which follows).

(18.3) (a) All n^{th} level objects, $A_1, A_2 \ldots A_r$ become n+1st level points, $p_1, p_2, \ldots, p_r$.

(b) All n^{th} level features, $P_1(A), P_2(A), \ldots P_m(A)$ become n+1st level primitives (atomic predicates), $\alpha_1(x), \alpha_2(x), \ldots \alpha_m(x)$.

(c) If $P_i(A)$ is an n^{th} level feature which begins with a string of existentially quantified set variables, i.e.,

$$P_i(A) \equiv \exists B_1, \ldots, \exists B_j \; [F(A, B_1, \ldots, B_j)]$$

then $F(A, B_1, \ldots, B_j)$ becomes an n+1st level primitive $\beta(x_1, x_2, \ldots x_{j+1})$. (all sets of n^{th} level objects which together satisfy F are known (see (18.1). Hence the values ("true" or "false") of $F(A, B_1, \ldots B_j)$ are known for all arguments and so are the corresponding values of $\beta(x_1, x_2, \ldots x_{j+1})$. This procedure permits both relations between objects and between other relations to be expressed at successive levels—see illustration below and Rothenberg (1973b).

Whenever the following conditions are met we advance a level:

(18.4) For each $R_i \in \bar{C}$ there exists a set of objects $\{A_1, A_2, \ldots A_n\}$ such that

(a) $$\bigcup_{k=1}^{n} A_k = R_i$$

and (b) $$n < \text{card}(R_i)$$

and (c) There does not exist an $R_j \in \bar{C}$ and a set of objects $\{B_1, B_2, \ldots B_m\}$ such that both $\bigcup_{\ell=1}^{m} B_\ell = R_j$ and all such B_ℓ (and n-tuples of B_ℓ) satisfy the same next level atomic predicates as the A_k (and n-tuples of A_k) in an $\{A_1, A_2, \ldots A_n\}$ satisfying condition (a) of (18.4) above. (Note that these atomic predicates include all features and relations constructed therefrom (as in (18.3) above) at the level being examined. This condition assures that there exist no $R_i \in \bar{K}$ and $R_j \in \bar{C}$ which consist of unions of indistinguishable (by features satisfied) objects.)

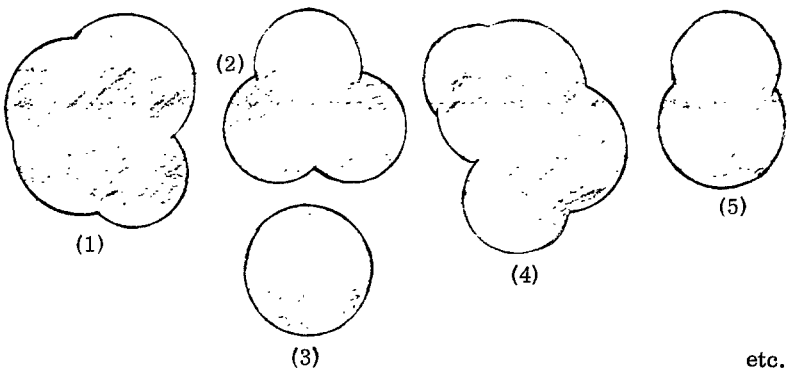
The above results in a change of hierarchical level whenever a reduction in the data to be represented results thereby. (Provision for trainer intervention when desired exists (see Rothenberg, 1973b).)

The definition of descriptive basis remains the same as in Section 2 except that all features at all levels are included. n^{th} level objects are defined in terms of n^{th} level points (i.e., n-1st level objects) exactly as first level objects are defined in terms of first level points. Similarly, the definitions of n^{th} level features, object classes, etc., are defined as before in terms of the other entities at the same level. However, the definition of n^{th} level primitive is extended to include the atomic predicate, $\epsilon^{(i,k)}(x^{(k-i)}, x^{(k)})$

wherein k and i are fixed (of course, $k-i < k \leq n+1$ where n is the hierarchical level). This atomic predicate is called recursive containment and asserts that there exists a sequence of points at the indicated (by the superscript) levels such that, if we put $x^{(k-i)}$ in front of the sequence and $x^{(k)}$ at the end, each term in the sequence is an element of the next (there are $i+1$ terms in such a sequence). (This allows statements about first level points in second level (and higher level features — see Rothenberg, 1973b).)

Example 7

Suppose we wish to distinguish photographs (from above) of any number of stacked discs from pictures of any other objects. Then the $R_i \epsilon \bar{\bar{K}}$ will consist of diagrams of the following form:



$R_j \epsilon \bar{\bar{C}}$ will consist of diagrams which cannot be formed by a finite number of overlapping circles. Suppose we have thus far generated feature $P_i(A) \equiv D_d(A) \wedge \exists x \exists y \forall z \left(\epsilon(x, A) \rightarrow \sim < [x, y, x, z]\right)$ (see Example 1, Section 1) which is strongly satisfied by dark circular regions and $P_2(A) = \exists B \exists x (\epsilon [x, A] \wedge \epsilon [x, B])$. (Note that $\bar{\theta}(P_2, \bar{\bar{K}}, \bar{\bar{C}})$, see (18.2), is very high in this example.) Note that conditions (18.4) are now met. Hence we advance a level as follows (see (18.3)): (1) All first level objects become second level points, $p_1, p_2, \ldots, p_r$. (2) $P_1(A)$ becomes a second level atomic predicate, $\alpha_1(x)$ (satisfied by second level points). (3) $P_2(A)$ is transformed by eliminating the existential quantifier at the left of its formula to form $F(A, B) = \exists x (\epsilon [x, A] \wedge \epsilon [x, B])$ which is satisfied by pairs of first level objects (which are now second level points). $F(A, B)$ now becomes a second level atomic predicate, $\beta(x, y)$. We may now, at the second level, construct the second level feature, $P_1^{(2)}(A) \equiv \forall x \left(\epsilon [x, A] \rightarrow \alpha_1(x) \wedge \forall y [\epsilon(y, A) \rightarrow \beta(x, y)]\right)$ which is satisfied only by $R_i \epsilon \bar{\bar{K}}$.

The following definitions should also be added to those of Section 2:

Description: This is a list of all features at all levels used in the descriptive basis and lists all objects strongly satisfying and satisfying each of these features and the elements of each such object.[1]

Filter predicates: (optional) These are formulas of the same form as first level features which must be satisfied by regions of the sample space called infra-objects in order for sub-regions of these infra-objects to be candidates for becoming first level objects. These may be used to substantially reduce the number of regions which must be considered for satisfaction of a first level feature. Typically, filter predicates might include formulae whose satisfaction guarantees that all points in a region have data values in a specified range and/or that such regions form connected sets. Filter predicates might also be used to eliminate the need for data dependent features.

13. Flow of the System

Features are evaluated by whether they are strongly satisfied by the required regions and perform required partitions (as initially specified). Other criteria (in the absence of initially specified objects and partitions) include feedback to a human trainer wherein objects in the description of a picture are projected on a cathode ray tube for comparison with the input picture. Also, at the request of the trainer, such objects may be successively replaced by other regions which satisfy (and strongly satisfy) the same features as these objects, so that the trainer may "see" what the system does not "see" as well as what it does "see" (see Rothenberg, 1973b)). The trainer thereby evaluates the performance of the system, which he indicates by means of positive or negative "feedback". Note again that the trainer may include any prior knowledge he has of the problem by direct insertion at any time of derived predicates and/or features. The system, in a practical application, thereby begins "learning" as if it had already "learned" all such knowledge provided by the trainer.

When feedback is absent and success has been obtained, the system nonetheless continues to compute, attempting to increase the satisfaction gap, to lessen the number of quantifiers in features and to eliminate unnecessary objects; i.e., those which are in the description, but which are not relevant to the problem as defined initially (i.e., do not correspond to any element of $\bar{K}$ or $\bar{C}$) or by the trainer.

Very briefly, the overall flow of the system is as follows (trainer interaction is not included):

---

[1] In some applications, if two features at the same level have identical data dependent expressions in their formulae and if an object which strongly satisfies one feature is properly contained in an object which strongly satisfied the other feature, the former object is omitted from the description.

(1) Initial features are either provided by the trainer or constructed as in Sec. 16.

(2) Simple and relation abstraction are performed. (When the lists are full reconstruct the predicate hierarchy and if there are still too many formulae, those with lowest reinforcement are eliminated.)

(3) That feature with the highest reinforcement is selected for mutation.

(4) The mutation is chosen as in previous sections. If the new feature meets syntactic requirements and has a higher measure of success than some other feature (and if the list of features is full), the feature with the lowest measure of success is eliminated and the new feature is added to the list of features.

(5) Is the satisfaction gap > 0? If so, perform reduction map.

(6) Is the satisfaction gap large enough to stop computing? (Predetermined by trainer.)

(7) Are the criteria for advancing a hierarchical level met? If so, advance the level, and return to step (1) at the next level. If not,

(8) Return to step (2).

The procedure is, of course, extremely complex in detail[1] — see Rothenberg (1973b).

20. Open Questions

A characterization of the semantics of predicates which can be constructed from a given set of primitives is required. A partial decision procedure which, as far as possible, avoids the generation of features which are conjunctions of contradictory predicates is needed. The generation of predicates with no finite model should also be avoided where possible (e.g., $P_i(A) \equiv \forall x \forall y \exists z \left(\epsilon[x, A] \wedge \epsilon[x, A] \longrightarrow [\epsilon(z, A) \wedge D_b(x, z, y)] \right)$ is satisfied by convex sets only when the data space is infinite—otherwise it is vacant). It would be useful to know in advance the minimum depth (of nested phrases) of the most deeply nested predicate needed to separate all inequivalent regions in a data space.[2] The replacement of "ease of satisfaction" by an easily computed measure which retains the ordering of formulae, F_i (by $E(F_i)$), when the data space is infinite is desirable.

Also significant is the question of whether the fraction consisting of the number of logically inequivalent formulae with n quantifiers divided by the total number of formulae with n quantifiers increases or decreases as n increases. This indicates the chances of skipping and not recapturing a needed feature in a "depth first" search on the tree of fea-

---

[1] Actually, a state-space search procedure is used—see Nilsson (1971).

[2] Note that a maximum must be set on the number of quantifiers in a feature, both for practical reasons and to avoid trivial solutions (i.e., no generalization).

tures and is relevant to search strategy (i.e., to proceed by large mutations which change the number of quantifiers or by small mutations?).

The extension of the system to include restricted third-order quantification (over features in lower levels of the descriptive basis) is being considered. This would enable the system to describe objects of similar kind (with respect to the descriptive basis) and would impart to it some self-descriptive power of a limited sort. However, problems arise due to the potential incompleteness of the descriptive basis at any time during the learning procedure.

Another important practical problem is the elimination of features which are trivial (as well as vacant or tautologous) in a particular application (e.g., $P_i(A) \equiv \exists x \exists y \left(\epsilon[x, A] \wedge \epsilon[y, A] \wedge \psi[x, y]\right)$). Such methods have already been derived, but the need for additional techniques will doubtless become evident during early experiments using the program.

It appears that for certain choices of primitives (such as those in the examples here) the use of Skolem functions (Schoenfeld, 1967) may result in eliminating existential quantifiers and hence in substantial simplification of the mutation procedure.

Of greatest significance, however, is that it appears that the entire system can be reduced to a first order system by (a) an appropriate choice of primitives (see Beth definability theorem (Schoenfeld, 1967; Reyes, 1969)) or (b) the development of a procedure for replacing some second order formulae by first order predicates recursively defined. For example, a second order formula asserting that "point x is connected to point y" may be replaced by

$$F(x,y) = \exists z\left([D_a(x,z) \wedge D_a(z,y)] \vee [D_a(x,z) \wedge F(z,y)]\right)$$

The computation expended in the procedures described increases rapidly as the number of primitives used increases. Hence, the choice of the smallest number of primitives which are appropriate to a problem and which result in a sufficiently expressive language is of central importance. (Note that the primitive used in our illustrations, $<(x,y,z,w)$, is extremely powerful—e.g., "$xy+yz = xz$" can be defined by using $D_b(x,y,z)$ and then we may define "$xy+vw = pq$" by using $xy+yz = xz$, $= (y,z,v,w)$ and $= (x,z,p,q)$, etc.) When many primitives are required, it is advisable to distribute them among the hierarchical levels used—i.e., so that primitives not at the first level are introduced at the second (e.g., "card (A)"), etc. Further work on this question could prove to be of considerable practical value.

It should also be noted that although this has not been done in the examples here, it appears profitable to experiment by using the system with primitives specifying (optical) spectral or holographic information.

Appendix to Section 17 (Topology)

All references are to Sierpinski's (1952) General Topology, first chapter (which deals with Frechet spaces). First we draw attention to several rather peculiar properties of Frechet spaces which are not topological spaces:

(1) In a Frecet space the sum of two closed sets need not be closed (p. 11, bottom). Hence the intersection of open sets need not be open. [Note that closed sets are sets which contain all their limit elements, p. 6 (4.), and open set is defined as the complement of a closed set, p. 11 (6.), and that the condition "every neighbourhood of an element of K is an open set" is necessarily true only when K satisfies the definition of a topological space (p. 38, middle). See also p. 12, Theorem 5.]

(2) In a Frechet space the condition that "a neighbourhood contains an element" does not imply that "the neighbourhood is a neighbourhood of that element". That is, a neighborhood may contain an element of which it is not a neighborhood (i.e., $x \in N(y)$ and $N(y) \neq N(x)$). Note also that conditions γ and δ on p. 38 may not be satisfied—i.e., distinct elements may not be contained in distinct neighborhoods.

Before attempting to explain the motivation for the axiom "The only closed sets are the null set and the universal set", we present an illustration of such a Frechet space:

Let our space, S, consist of nine points: $S = \{a,b,c,d,e,f,g,h,i\}$. For convenienc convenience, picture them in a rectangular array:

a . b . c .
d . e . f .
g . h . i .

Let the minimum neighborhood of a point consist of all points which are nearest to it in the above rectangular array (i.e., those points directly above, below, to the right of and to the left of it (if there are such points)). Thus our minimal neighborhoods are:

$$N(a) = \{a,b,d\}$$
$$N(b) = \{a,b,c,e\}$$
$$N(c) = \{b,c,f\}$$
$$N(d) = \{a,d,e,g\}$$
$$N(e) = \{b,d,e,f,h\}$$
$$N(f) = \{c,e,f,i\}$$
$$N(g) = \{d,g,h\}$$
$$N(h) = \{e,g,h,i\}$$
$$N(i) = \{f,h,i\}$$

For all x, the neighborhoods of x will be all sets which include N(x).

Note that all sets except the null and the entire set are neither open nor closed. For example, let $E = \{c, e, f, g, h, i\}$. Let E' denote its derived set of E (i.e., the set of all its limit elements). Let F denote the complement of E(i.e., $S - E$) and F' denote the derived set of F. Then $F = \{a, b, d\}$. Note that $b \in E'$. Hence $\sim(E' \subset E)$ and E is not closed. Note also that $e \in F'$. Hence $\sim(F' \subset F)$ and F is not closed.

The same argument can be repeated for all other non-empty proper subsets of S. Also, S can be made as large as we wish by adding elements to the rows and columns of the rectangular array. If the minimal neighborhoods are similarly defined all non-empty proper subsets of S will be neither open nor closed.

Observe that the imposition of the conditions on a Frechet space that "each point have a minimal neighbourhood and that ϕ and S be the only closed sets" is consistent with the definition of a minimal neighborhood of a point in a finite space as containing those points "adjacent" to that point. The conditions also prevent the shrinkage of neighborhoods to the trivial case where they contain only one point. This is as might intuitively be expected in a finite space, and the preservation of conditions (a) - (i) of Section 17 is also in accordance with intuition. Also, we are assumed that our entire space is connected so that a graph metric may be imposed (Section 17).

Note that condition δ on page 38 of Sierpinski does not hold, and hence we do not have a topological space.

Another alternative was available: The condition that "ϕ and S are the only closed sets" might have been replaced by "all sets are both open and closed". Then we could have defined the minimal neighborhoods in our above illustration non-uniquely:

$$N_1(a) = \{a, b\} \quad N_2(a) = \{a, d\}$$

$$N_1(b) = \{b, a\} \quad N_2(b) = \{b, c\} \quad N_3(b) = \{b, e\}$$

$$N_1(c) = \{c, b\} \quad N_2(c) = \{c, f\}$$

$$N_1(d) = \{d, a\} \quad N_2(d) = \{d, e\} \quad N_3(d) = \{d, g\}$$

$$N_1(e) = \{e, b\} \quad N_3(e) = \{e, d\} \quad N_3(e) = \{e, f\} \quad N_4(e) = \{e, h\}$$

etc.

Now each element would be the intersection of all its neighborhoods. But now condition β on page 38 of Sierpinski would not be satisfied and we still would not have a topological space. Of greater importance, how would we now obtain the necessary graph metric.

Also, the imposition of this type of neighborhood system on a finite space seems counterintuitive.

In summary, the purpose of the topological section of this paper is to impose conditions on the given structure $<S, \leqslant_0>$ so that it could be mapped into a structure with fewer elements, $<P, \leqslant_\epsilon>$,[1] such that if a subset, A, of S is a model of a formula, F, the image, f(A), of A (in P) is a model of a formula, $\overline{F}$, which is obtained from F by the substitution of a modified quantifier for a universal or existential quantifier in F.

[1] $xy \leq_\epsilon zw \equiv xy - zw \leq_0 \epsilon$

BIBLIOGRAPHY

Addison, J., L. Henkin and A. Tarski (1965), "Theory of Models", Proc. 1963 International Symposium at Berkeley, North Holland Co., Amsterdam.

Amarel, S. (1962), "An Approach to Automatic Theory Formation", in Principles of Self Organization, Trans. Illinois Symposium on Self Organization (Von Foerster and Zopf, Eds.), Pergamon Press, New York.

Banerji, R. (1969), Theory of Problem Solving, An Approach to Artificial Intelligence, American Elsevier Publishing Co., Inc., New York.

Bell, J. and A. Slomson (1969), Models and Ultraproducts, North Holland Co., Amsterdam.

Block, H.D., N.J. Nilsson and R.O. Duda (1964), "Determination and Detection of Features in Patterns", Computer and Information Sciences (J.T. Tou and R. Wilcox, Eds), Spartan Books, 75-110.

Bongard, M. (1970), Pattern Recognition (J.H. Hawkins, Ed.), Spartan Books

Bremermann, H.J. (unpublished manuscript), "Artificial Intelligence and Neurobiology", Department of Mathematics, University of California, Berkeley.

Ellentuck, E. (1971), "Direct Products of Relational Systems", U.S. Air Systems Command Final Report, Jan. 1, 1971 (see Rothenberg, 1971).

Fogel, L.J., A.J. Owens, and M.J. Walsh (1966), Artificial Intelligence through Simulated Evolution, John Wiley and Sons, New York.

Friedberg, R.M. (1958), "A Learning Machine, Part I", IBM J. Research and Development, 2, 2-13.

Friedberg, R.M., B. Dunham and J.H. North (1959), "A Learning Machine, Part II", IBM J. Research and Development, 3, 282-287.

Hawkins, J. (1970), "Textual Properties of Pattern Recognition", Picture Processes and and Psychopictorics (B.S. Lipkin and A. Rosenfeld, Eds), Academic Press, New York, 347-370.

Hintikka, J. (1953), "Distributive Normal Forms in the Calculus of Predicates", Acta Philos. Fennica, 6, 1.

Julesz, B. (1969), "Cluster Formation at Various Perceptual Levels", in Methodology of Pattern Recognition (S. Watanabe, Ed.), Academic Press, New York, 297-315.

Kolers, P.A. (1970), "The Role of Shape and Geometry in Picture Recognition", Picture Processing and Psychopictorics (B.S. Lipkin and A. Rosenfeld, Eds.), Academic Press, New York, 181-202.

Lee, R.C.R. (1967), "A Completeness Theorem and a Computer Program for Finding Theorems Derivable from Given Axioms", Ph.D. Dissertation, University of California at Berkeley.

Lofgren, L. (1967), "Recognition of Order and Evolutionary Systems", in Computer and Information Science - II (J. Tou, Ed.), Academic Press, New York, 165-175.

Minsky, M. (1963), "Steps Toward Artificial Intelligence", in Computers and Thought (E.A. Feigenbaum and J. Feldman, Eds.), McGraw-Hill Book Co., Inc., New York, 406-450.

Muerle, J.L. (1970), "Some Thoughts on Texture Discrimination by Computer", Picture Processing and Psychopictorics (B.S. Lipkin and A.Rosenfeld, Eds.), Academic Press, New York, 371-380.

Nilsson, N.J. (1971), Problem-solving Methods in Artificial Intelligence, McGraw-Hill Book Co., Inc., New York.

Pessin, A. and D. Rothenberg (in preparation), The Weak Second Order Predicate Calculus as a Programming Language.

Reyes, G.E. (1969), "Local Defineability Theory", Annals of Math. Logic, 1, 101.

Rosenfeld, A. and B. Lipkin (1970), "Texture Analysis", in Picture Processing and Psychopictorics (B. Lipkin and A. Rosenfeld, Eds.), Academic Press, New York, 309-347.

Rothenberg, D. (1971), "An Adaptive Linguistic Model for Pattern Representation and Recognition", Final Report, Air Force Systems Command, USAF, Directorate of Mathematical and Informational Sciences, January 1, 1971.

Rothenberg, D. (1973a), "A Pattern Recognition Model Applied to the Perception of Pitch", AFOSR Technical Report, January 1969, revised version to appear as series of articles in Mathematical Systems Theory. See also this volume.

Rothenberg, D. (1973b), "A Theory of Feature Generation", a series of four articles to be submitted to Mathematical Systems Theory (Copies available from Mathematics Department, Rutgers University, University College, New Brunswick, New Jersey).

Schoenfeld, J. (1967), Mathematical Logic, Addison Wesley, Reading, Massachusetts.

Selfridge, O.G. and U. Neisser (1963), "Pattern Recognition by Machine", in Computers and Thought (E.A. Feigenbaum and J. Feldman, Eds.), McGraw-Hill Book Co., Inc., New York, 237-250.

Sherman, R. and G.W. Ernst (1969), "Learning Patterns in Terms of Other Patterns", in Pattern Recognition, Pergamon Press, New York, (Vol. 1), 301-313.

Sierpinsky, (1952), General Topology, University of Toronto Press, Toronto.

Slagle, J., C. Chang and R. Lee (1969), "Completeness Theorems for Semantic Resolution in Consequence Finding", Proc. International Joint Conference on Artificial Intelligence, May 7-9, 1969, Washington, D.C. (D.E. Walker and J.M. Norton, Eds.), 281-285.

Solomonoff, R.J. (1964), "A Formal Theory of Inductive Inference - Part I", Information and Control, 7, 1-22.

Solomonoff, R.J. (1964), "A Formal Theory of Inductive Inference - Part II", Information and Control, 7, 224-254.

Uhr, L. and C. Vossler (1963), "A Pattern Recognition Program that Generates, Evaluates, and Adjusts its Own Operators", in Computers and Thought, (E.A. Feigenbaum and J. Feldman, Eds.), McGraw-Hill Book Co., Inc., New York, 251-268.

Wang, H. (1960), "Toward Mechanical Mathematics", IBM J. Research and Development, 4, 2-22.