FGP: A Virtual Machine for Acquiring Knowledge from Cases

Scott Fertig and David H. Gelernter
Department of Computer Science
P. 0. Box 2158 Yale Station
Yale University
New Haven, Connecticut 06520-2158

Abstract

Large case databases are numerous and packed with information. The largest of them are potentially rich sources of domain knowledge. The FGP machine is a software architecture that can make this knowledge explicit and bring it to bear on *classification* and *prediction* problems. The architecture provides much of the functionality of traditional expert systems without requiring the system builder to preprocess the database into rules, frames, or any other fixed abstraction. Implementations of the FGP machine use similarity-based reminding and the cases themselves to drive the inference engine. By having the system calculate and incorporate a measure of feature salience into its distance calculations, the FGP architecture smoothly copes with incomplete data and is particularly well-suited to weak-theory domains. We explain the model, describe a particular implementation of it, and present test-results for a classification task in three application areas.

1 Introduction

The *knowledge acquisition bottleneck* is still one of AI's major problems. It has engendered commentary from senior AI practitioners for years, sparked the current revival of interest in machine learning, and is the primary motivation behind a multi-million dollar project to assemble a massive knowledge base (on the order of 10^8 facts) [Guha and Lenat, 1990]. In recent workshops, many researchers have come to the conclusion that the effective use of large databases is our best hope to break through this bottleneck [Friesen and Golshani, 1989]. Our work is in this vein, and is an attempt to devise a domain-independent software architecture and related algorithms to extract knowledge from a large database of unstructured cases, a term we define below.

Cases are similar to feature-vector representations of data, but more general. They may be incomplete; individual cases need to have values for only an arbitrary subset of the universe of features. Cases are not assigned to fixed categories. Rather any feature in the feature-universe is a potential category; the program must be willing to direct its inference process at determining the *probable value* of any arbitrary feature for a particular case. We say *probable* because we make no assumption about consistency of cases. Two cases may have identical values along every dimension but one.

We adopt this definition because these are the characteristics observed in cases drawn from real-world domains. The question we sought to address is whether the static feature information stored in cases provides enough material to do useful inferencing and learning. The approach we choose is exemplar-based *(cf.* [Porter *et al.*, 1990]) and relies on the cases themselves to drive the inference engine. Our goals are the following: The system should "reason" on the basis of specific cases and groups of cases, and should therefore be able to cite specific precedents (including precedents that are themselves incompletely understood), to modify its behavior on the basis of every new information-providing transaction, and to subsume the functions of a conventional information-retrieval system. It should be a viable learning system, equalling or surpassing the accuracy of other purely data-driven approaches to classification and prediction problems, and do this while maintaining its availability as a real-time system. Unlike typical connectionist approaches, it should be designed to interact with a user as a case is described incrementally, or to generate line-by-line commentary on new cases. Most important, it should attempt to display not only "quantitative" but "qualitative" expertise. In our experimental definition, qualitative expertise requires (particularly on hard cases, but these are the whole point) pursuing possibly premature hypotheses, offering (occasionally brash) guesses and citing interesting precedents on the basis of a small but evocative degree of feature overlap. We are developing "simulated speculation" in an effort to approximate this behavior.

2 The FGP machine

The program we've built in an initial attempt to realize these properties is the FGP machine, after its basic operations—fetch, generalize and project. We imagine the FGP machine's database as a collection of regions in space *(cf.* the standard vector space text-retrieval model). Each element of the database corresponds to

some region. Nearby regions correspond to nearby cases. When presented with an inquiry, the machine's basic task is to add to the database a new region corresponding to the inquiry. Stationing itself on top of this new region (so to speak), the machine then looks around and reports the identities of the nearby regions—these will correspond to elements of the database that are nearby to, in other words closely related to, the subject of the inquiry. We can then inspect this list of nearby regions and "generalize"—determine which attributes tend to be shared in common by all or by most of them. We can guess that these common attributes are likely to hold true for the case being described in the inquiry as well.

Having reached whatever conclusions seem reasonable, the machine simulates "speculation." Temporarily turning aside from the inquiry in hand, it focusses on any "evocative possiblities" that may have suggested themselves during the examination of nearby regions. An "evocative possibility" is a datum that *might* be true, and that would be significant if it were. The machine's interaction with the user (see Figure 1) represents a combination of fairly safe conclusions, speculation experiments and the subsequent investigation of resulting guesses.

The system can operate interactively, but here it is working in "commentary" mode: the user presents an entire case; the system scans it element-by-element, offering comments. This case initially seems malignant (note the early mention of related cases with diagnoses of infiltrating ductal carcinoma); the fact that the mass has not changed in density and has no comet (contradicting the system's guesses, which in the nature of guesses will often be wrong) points in the other direction ("cyst" and "fed" refer to benign diagnoses); but further data, particularly the absence of a halo, tips the balance, and the system guesses that this is a malignant mass. This guess is correct, and the diagnosis was in fact infiltrating ductal carcinoma. This transcript is driven by a small collection of 67 cases, which is the only domain knowledge provided.

An FGP machine is defined in terms of a single kind of data-object and three primitive operators. These define a virtual machine in terms of which the system is programmed. We summarize the essential points in the remainder of this section; see [Fertig and Gelernter, 1989; Fertig, 1990] for further discussion.

2.1 Data-objects and databases

FGP machines run off a database of a single type of data-object, a feature tuple to which we refer generically as a r. A r consists basically of a list of attribute-value pairs; we give examples below. An FGP database consists of an unordered collection of r's. A new case for inclusion in the database is presented as a r, and a query is an incomplete r—a partial list of attribute-value pairs, with a request that the system fill in certain missing ones.

Individual cases are stored in r's; paradigm cases (cases that represent not a single object or incident, but a generalized or prototypical object or incident) are stored in formally-equivalent "heavy r's", as we discuss. The basic FGP-operations described below operate on r's individually and in groups.

The feature tuple

((name apple) (type fruit) (color red))

is a r that represents a single apple;

((name apple 100) (type fruit 100) (color (red 50) (yellow 50)))

is a r that represents one hundred different apples; half were yellow and half were red, and beyond this fact no individual apple can be distinguished. When the count field is omitted, "1" is the default. A relatively "weightier" r is a r that reflects a greater number of individual cases. We use M to represent an unordered collection of r's; an FGP database of stored cases and paradigms is an *M.L* is a list of r's ordered on their "closeness" to some other r: we explain below.

2.2 Primitive operators

The three basic FGP operations are *fetch, generalize* and *project.* They work as follows.

fetch maps a r and an M to an L: given a feature-tuple and a database of feature-tuples, it produces an ordered list of those feature-tuples in the database that are "closest to" the r mentioned in the query, *fetch* uses a two-step procedure.

First it calculates a "distance" from the new point to every point in memory[1]; all cases further away than some parameterized threshold are removed from further consideration. The distance of one case from another cannot be statically determined; two cases may be more closely related in the context of one goal than in another. Therefore *fetch's* calculation not only takes into consideration the number of shared attributes and their types, but also, in the context of a request to fill in values for missing attributes, the "evocativeness" of each with respect to the current goal—a more evocative feature is one that recalls a group of cases with a more highly focussed set of values for the goal attribute.[2] Features are weighted in the distance calculation in proportion to their evocativeness. The final distance score is the weighted sum of the individual feature distances normalized by the size (number of features) of the cases. In standard terms, the similarity metric measures the weighted euclidean distance between cases, where the weighting is done automatically and reflects the evocativeness of individual features with respect to the goal. The system comes with default feature distance metrics defined for nominal, ordinal, and dimensional (real-number valued) data, although user-defined feature metrics can be used to reflect more specialized knowledge of the domain.

Next *fetch* checks to see if there exists a well-defined group of *very-close* points among those remaining by performing a crude cluster analysis. Details can be found in

[1]This calculation needn't require that every r in *M* be examined; as a first cut, we can ignore all cases that fail to intersect the query on at least one attribute. We use hashing to determine the candidate cases (to exclude the irrelevant ones).

[2]The evocativeness of an attribute-value pair with respect to a goal attribute is inversely proportional to the entropy (disorder) of the distribution of values for the goal represented in the group of cases returned by *fetch.*

(AGE 58)
(MASS-DENSITY ISO-DENSE)
(MASS_BORDER-CQMPLETE? NO)
(MASS_TYPE_BORDER IRREGULAR)
(MASS_BORDER_DEFINED? MO)

Speculating: MASS-DENSITY-CHANGED?...
Guessing INCREASED - e.g.
case ((id 14) (age 46) (diagnosis CA_INF-DUCTAL))
case ((id 50) (age 70) (diagnosis CAJNF-DUCTAL))

Speculating: MASS-COMET?...
Guessing YES - e.g.
case 14
case ((id 40) (age 69) (diagnosis CAJNF-DUCTAL))

(MASSJLOCATION UIL)
(MASS_SIZE_CHANGED? YES)
(MASS-DENSITY-CHANGED? HO)
(MASS-COMET? NO)

Speculating: BACKGROUND-DENSITY...
Guessing DENSE - e.g.
case ((id 21) (age 61) (diagnosis cyst))
((id 47) (age 45) (diagnosis fcd))

(MASS_HALO? NO)
(BACKGROUND_DEMSITY MODERATE)

Concluding
(ARCHITECTURAL-DISTORTION? NO)

Speculating: MALIGNANT?...
Guessing YES - e.g. cases (2 6 8)

Speculating: SKIN-CHANGES...
Guessing RETRACTION - e.g. cases (2 8 28)

(SKIN-CHANGES NO)
(NIPPLE-INVERSION? NO)
(ADENOPATHY? NO)
(FAMILY-HISTORY-CANCER SISTER)
(PERSONAL-HISTORYJCANCER NO)

Closest known cases:

(19) (YES) (CAJNF-DUCTAL)
(33) (YES) (CAJNF-DUCTAL)
(26) (YES) (CA-INF-DUCTAL)
(28) (YES) (CAJNF-DUCTAL)
(18) (YES) (CA)

YES has been concluded or guessed for MALIGNANT?

Speculating: DIAGNOSIS...
CA?
CAJNF-DUCTAL?

Figure 1: Transcript of an FGP machine operating in the domain of mammography. The user's case description is in the left column, the system's commentary on the right.

[Fertig, 1990]. An ordered list of these very-close points is returned as *fetch's* value.

generalize maps an $\mathcal{L}$ to a τ: it takes an ordered list of feature-tuples and "compresses" them into a single new feature-tuple. The weighter a r and the closer it is to the top of the list, the larger the contribution its attribute-value pairs make to the combined r returned by *generalize.* Suppose we query on the r (name apple), and suppose that M holds one hundred individual apples, half red and half yellow; a *generalize* operation over a list consisting of one hundred apples, half red and half yellow, yields a single r that might look like our second example above.

project maps a r to a r: given a feature-tuple it returns a new tuple constructed from a subset of the features in the original. While *project* is a purely syntactic operation, it is used by higher level operations (see the discussion of *refocus* below) to change contexts; the system focusses on those attributes and values that are deemed "interesting"[3], temporarily ignoring other information on hand.

2.3 The basic cycle

Given this three-instruction virtual machine, how does the system operate? The basic cycle is two phase: (1) extend the current r; (2) choose a new current r, and repeat. Step one is implemented by an *extend(r)* function that is defined in terms of *fetch* and *generalize.* Step two is implemented by *refocus* which is defined in terms of all three.

To extend a r — to discover new implications given our database of cases and paradigms — we begin by executing the operation *generalize(fetch(M,* r)), where M is the database. If r, for example, describes a particular patient, $fetch(\mathcal{M}, \tau)$ will return a list of remembered r's that are close to (similar to or reminiscent of) this particular patient; executing *generalize* over this list will produce an amalgam of all these remembered cases. Any highly-focussed and sufficiently-weighty values can be classified as conclusions: if the memories examined by *generalize* mainly have a value of "blonde" for attribute "hair-color", say, the system will conclude that (hair-color blonde) is likely to characterize this case as well. It reports (hair-color blonde) to the user as a conclusion and augments the current *r* with this new attribute-value pair. The system attempts to conclude any value turned up by the *fetch-generalize* combination which hasn't yet been seen in the context of the current query. Values which contradict[4] are withdrawn; the user's input always takes precedence over system guesses. The *extend* operation is complete when all values that can be concluded have been and all contradictions removed.

[3]The system uses the evocativeness values calculated by *fetch* to determine which features are interesting.

[4]As expected, two distinct values of a boolean-typed attribute always contradict. System-concluded values of other types of attributes contradict only if this information is specified in the attribute's distance metric. See [Fertig, 1990] for more details.

2.3.1 Simulated speculation

Refocus is then invoked over the extended r. Its role is to examine a τ and refocus attention from this entire *r* to one (possibly small and conceivably unrepresentative) part of it. This element considered in isolation may serve as a seed for a new set of inferences. We call this process "simulated speculation." *refocus* may choose no, one or many data points; each chosen data point becomes the current τ in turn. The more evocative a data point with respect to the goal—the more sharply-defined the cases nearby a τ consisting only of that data point with respect to the goal attribute, in other words—the likelier target for *refocus.* The more sharply a data point stands out from the pack—by assumption it won't stand out clearly enough to qualify as a conclusion, but there are many intermediate shadings here—the likelier a *refocus* target.

Typically, the system will examine each of a small set of values associated with a particular attribute whose value, if known, would focus the search space considerably. The system performs the basic *fetch-generalize* cycle on each of these seed-tuples and is left with a set of regions in vector-space. One may be much closer to the original query than the others and may therefore be mergeable with it. The reader can see the system's behavior during several *refocus* experiments by examining the transcript shown in figure 1. *refocus* first announces the attribute *projected* to, followed by any values tentatively guessed as the result of the speculation experiment. It then gives pointers to specific cases that both have this value and also resemble the rest of the user's input.

The system can operate in various modes; in the example given the basic cycle is repeated after considering each attribute of the presented case. After all input has been considered, the system will execute the basic cycle one final time. It is then in a position to present a list of the closest cases in the database given all the input and *project* to any undetermined goal attributes in one final attempt to determine values or at least narrow the space of possibilities.

Why should simulated speculation lead to more accurate performance? As a user enters a case the system is automatically calculating a weight for individual features based on their evocativeness, or how closely they focus the search space with respect to the goal attribute. In statistical terms, we can think of the system as treating all features other than the goal as independent variables and the goal as the dependent variable. The system is automatically calculating a weight for each non-goal feature in isolation, in essence making the simplifying assumption that it is dealing with a simple bivariate distribution. Of course this isn't correct; in fact each case is better thought of as a many-internal multivariable data point in which no feature can be labelled *a priori* as *dependent.* And as in any multivariable case, correlations between any two features may be artifacts that disappear when seen in the context of additional variables. The weighting of individual features according to a bivariate model is essentially a useful approximation but not guaranteed to be accurate. It will lead to inaccu-

rate retrieval when the impact of one feature (variable) is enhanced or diminished considerably when it appears in the context of one or more others.

The heuristic approximation of feature salience by treating each feature as the independent variable in a bivariate model is a useful approximating device that fails to capture in its calculation important *combinations* of features. It is clear that we cannot calculate a salience score for each subset of all possible features. The cost is exponential and is clearly impractical for domains in which cases consist of more than a handful of features. However, we may be able to identify subsets of features that are important when considered together through the judicious use of simulated speculation.

For example, consider what happens when the system sets aside its standard processing and speculates on an undetermined goal feature. The system first collects all the values for that feature that appear among the cases currently judged to be close. Then a *fetch* is performed on each of these values and yields a profile (i.e. a *generalized T)* in which some (other) features are strongly associated with that value. Since now the mapping is one-to-many there is no danger of hidden interactions among the features masking or spuriously adding to the importance of any single feature, and those strongly associated with one goal value and not with others *are* good markers for the presence of that value. Those that appear in all or most of the retrieved profiles are bad or completely ineffective markers.

As outlined, simulated speculation can help the system adjust the feature salience measures crucial to effective retrieval of relevant cases. In so doing it improves accuracy. Simulated speculation also improves efficiency, that is how quickly it converges on a list of relevant cases and determines the values for all goal features (if possible). Similated speculation may narrow the search space directly by eliminating from further consideration, at least temporarily, potential goal values that are inconsistent with the case as currently entered. The subsets of features which are discovered to be strongly associated with potential goal values provide an automatic standard by which to judge whether the current case is consistent with that goal value.

Finally, simulated speculation provides a road map for further processing. Among the features strongly associated with a goal may be some about which nothing has been entered. Directing its attention to these, and using the same *refocus* operation, the system may be able to determine the probable values for one or more of these non-goal features that are nevertheless highly correlated with the goal. In so doing it will have moved a significant step closer to pinpointing the goal. Even if unsuccessful on its own, the system can ask the user if he knows the value for any of these correlated features and thus engage in a reasonable directed dialogue.

2.4 Higher-level operations

An interesting and useful collection of higher-level operations can be based on these primitives. We have seen one, *refocus,* which is central to our attempt to simulate speculation. Others are discussed in [Fertig and Gelernter, 1989; Fertig, 1990]. The primitive operations have proven sufficient to allow the FGP machine to assemble intelligent sequences of database retrieval out of individual comparisons, performing if necessary a large number of individual analyses in an attempt to converge on a goal. The results discussed in the next section along with the example transcript illustrate the power of this structure—simple primitives, more complicated strategies—in achieving a measure of qualitative and quantitative expertise.

3 Experiments

This section discusses experiments performed with the implemented (serial) version of the FGP system.

The experiments were conducted on case databases in three domains: descriptions of mammograms collected at Yale-New Haven hospital, Fisher's Iris plant database [Fisher, 1936], and a collection of international folk dance descriptions.[5] These datasets are described below.

We wanted to validate the FGP system performance in three ways. First, we wanted to verify that the higher-level operations such as *refocus* worked as intended. Recall that all FGP operations are built recursively from the three primitive operations *fetch, generalize,* and *project.* The behavior of *fetch* depends not only on the syntactic structure of the inputs to the operation but also on the domain information contained in the database. This semantic dependency makes testing any implementation of an FGP machine on synthetic data only partially effective. Real data from real domains must be used to exercise the various component operations in the algebra.

We also wanted to quantify the performance of the finished system. Although we do not intend for FGP systems to be used solely or even primarily as stand alone diagnostic systems, accuracy in a diagnostic task is an easily understood metric and probably the most commonly used to compare the performance of automatic classifiers and expert systems to that of human experts. For this reason, we've chosen diagnostic accuracy as the primary quantitative measure to be used in the three domains.

Measures of qualitative behavior are our third and final means of measuring the system's performance. As we've stressed, an FGP system is intended to be used as an interactive tool that can suggest plausible hypotheses and appropiate comments as information about a case is entered. This kind of behavior is hard to assess quantitatively. Nonetheless, we have made an attempt and have developed a metric to measure the system's rough *qualitative* behavior in the radiology domain.

3.1 Folk Dances

The first context in which discuss the FGP machine's accuracy is the one in which a reasonable size database and domain expert first presented themselves. Problem: Determine the nationality of a previously unseen folk dance given a list of other attributes (such as whether it is done by couples or in fours, in a circle or a line,

[5] We thank Jonathan Young for providing this database.

etc). The database has 643 cases spanning 46 different nationalities; the average number of features per case is 3.6.

As reported in [Fertig and Gelernter, 1989], we presented 50 test cases for the FGP machine to classify. The domain expert, who happened to be the creator and maintainer of the database, answered correctly 53% of the time. The prototype determined the correct nationality 57% of the time overall. Note that the classification task is not trivial given the number of possible classes and the sparseness of the training data. The performance of both the domain expert and the program is quite good although under 60% correct in an absolute sense.

We applied the current version of the FGP machine, which incorporates a working implementation of the *refocus* operation, to the same database and task to determine if "simulated speculation" could improve accuracy. We found that it did: the FGP machine was able to determine the correct nationality 65% of the time overall. The system was successful in 8 of 13 attempts to use refocus to determine the dance's nationality when it failed to draw a conclusion from the input directly. The FGP machine also used refocus to guess the values of other attributes much as seen in figure 1. These operations are harder to quantify, as many test cases are missing attributes. We were able to conclude, however, that the system guessed either a correct or consistent[6] value in 60% of these refocus experiments. The guess was clearly wrong 17% of the time.

3.2 Irises

Our second application is in the domain of plant identification. R.A. Fisher's data on three types of iris [Fisher, 1936] is often cited in the pattern recognition literature, and has become a de facto performance benchmark for new classification systems [Gates, 1972; Duda and Hart, 1973; Dasarathy, 1980]. Following the example of the previous studies, we randomly divided the Iris db into two partitions (of size 120 and 30), making sure to include equal numbers from each class in both the training and test partitions.

Table 2 gives the basic result. The FGP machine correctly classifies 23 out of 30 of the test cases (76.6%), slightly improving on the performance of Dasarathy's Neighborhood Census Rule algorithm (Table 1), the best-performing of the nearest neighbor approaches [Freeman, 1969; Gates, 1972; Dasarathy, 1980].

Dasarathy allowed his system to choose a simple "don't know" option if none of the known categories was closer than some parameterized threshold. The FGP system's ability to suggest multiple categories when unable to classify an item unambiguously is similar although often more informative. Suggesting a few possible categories is equivalent to ruling many categories out. This behavior is meant to capture an expert's ability to reserve judgement when faced with an unusual combination of facts, while still ordering past experiences in relation to the new situation.

[6] A value was judged consistent if it appeared in a majority of the cases in the database with the same nationality as the testcase.

Class	Correct	Wrong	Don't Know
Setosa	10	0	0
Versicolor	9	1	0
Virginica	3	3	4

Table 1: Frequency of correct classification from Dasarathy80

Class	Correct	Wrong	Don't Know
Setosa	10	0	0
Versicolor	8	0	2
Virginica	5	0	5

Table 2: Frequency of correct classification by FGP system

3.3 Radiology

The current prototype has also been tested against a small database of patient records, descriptions of mammograms. There were originally 88 records in the database; 20 cases were reserved for testing and 67 were used to seed the system, spanning 13 possible diagnoses (one of these 13 possibilities was the diagnosis *normal,* meaning no disease present).[7] The system was presented with the 20 test cases and asked to judge if a malignant lesion was indicated, and if so to determine a specific diagnosis. As discussed above, the system would present a short list (≤ 4) of possible diagnoses if unable to decide on one with certainty.

The domain expert was the radiologist who had compiled the database.[8] Working from the descriptions of the mammograms alone, he accurately judged the malignancy of the testcases at the 68% level. The system performed slightly worse at 63%. However the system outperformed the domain expert in producing a differential diagnosis, with the right answer being stated outright or appearing in a short list of possibilities $(<= 3)$ 70% of the time to the clinician's 60% correct performance.

3.3.1 Measuring qualitative performance

As we have discussed, we wanted the FGP system to exhibit qualitative expertise as well as quantitative accuracy. We developed the *refocus* operation to model the human propensity to set aside goal-directed thinking in the face of evocative events. We have labeled the FGP system's ability to behave similarly "simulated

[7] One record was thrown out because no diagnostic information was included.

[8] Dr. Paul Fisher of the Department of Diagnostic Radiology, Yale University School of Medicine. Dr. Fisher's clinical specialty is mammography.

[9] We present no evidence for this propensity as it is tautologically beyond question: an event could not, by definition, be evocative if it did not cause the observer to stop and take notice.

speculation." We discussed it in the context of both the folk dance experiment and figure 1. Still, we would like to *quantify* the system's qualitative performance, and in particular the usefulness of simulated speculation.

We took two approaches: First, measure the absolute improvement in classification accuracy when simulated speculation is enabled. As reported, the FGP system's accuracy improved from 57 to 65% correct in the folk dance experiment, for a 14% gain. The system's performance is identical with or without speculation when applied to the Iris test, a fact we expected given the extremely small number of features per case. Preliminary results indicate an 8% improvement in the radiology domain.[10] Note that the improvement in any domain is on precisely the most difficult cases. *Refocus* is only invoked when the direct evidence for choosing one value over another is underwhelming or nonexistent.

Second, we wanted to measure whether simulated speculation caused the system to make appropiate comments "in about the right places and at the right times." To test this, we asked the domain expert to provide us with an audio tape of his analysis of five of the x-rays chosen at random. We presented the same five to the FGP machine and then placed the transcripts side-by-side.

We then counted how often the FGP system and domain expert agreed that a particular feature indicated a malignant or benign condition in the patient. The system agreed on 18 of the 24 times the expert chose to mention that a particular feature indicated malignancy or its opposite. The system made this type of comment at a total of 16 junctures where the expert said nothing, but these comments were clearly inappropiate in 3 places only (in the opinion of the domain expert). While preliminary, we are encouraged by this concordance of opinion.

4 Related Work

The idea of retrieval based upon similarity is crucial to our system's ability to select a few relevant cases from a dense and extensive database, an idea exploited by research in case-based reasoning, analogy and information retrieval. [Fertig, 1990] provides an extensive literature review.

5 Work in Progress

Although it is built to be reasonably efficient, the current FGP machine is inadequate to handle case databases of more than 1000 records assuming 10-20 features per case. The technique is intended specifically for use with large databases—the larger the better. Generating transcripts like the ones in the figures requires repeated computations against many or all elements in the database. Standard indexing strategies are useful to some extent, but in this problem (unlike the keyword-based text retrieval problem, for example), the database is likely to be fairly homogeneous, with most features present in most cases. It's clear that if this program is to perform well, we must be able to execute operations involving large portions of large databases quickly. Parallel programming techniques explored by members of the Linda group at Yale are a promising approach (e.g. [Carriero and Gelernter, 1988], and we are currently applying these techniques to a new implementation of the FGP machine.

[10] The numbers reported above for the radiology experiment are with "speculation" enabled.

References

[Carriero and Gelernter, 1988] Nicholas Carriero and David Gelernter. Applications experience with Linda. In *Proceedings of the ACM Symposium on Parallel Programming,* pages 173-187, July 1988.

[Dasarathy, 1980] B. Dasarathy. Nosing around the neighborhood: A new system structure and classification rule for recognition in partially exposed environments. *IEEE transactions on Pattern Analysis and Machine Intelligence,* 2(1):67-71, 1980.

[Duda and Hart, 1973] R. Duda and P. Hart. *Pattern Classification and Scene Analysis.* John Wiley and Sons, 1973.

[Fertig and Gelernter, 1989] Scott J. Fertig and David H. Gelernter. Machine musing and expert databases. In Larry Kerschberg, editor, *Expert Database Systems,* pages 605-620. Benjamin Cummings, 1989.

[Fertig, 1990] Scott J. Fertig. The design, implementation, and performance of a database-driven expert system. Technical Report 851, Yale University Department of Computer Science, Aug 1990.

[Fisher, 1936] R. Fisher. The use of multiple measurements in taxonomic problems. *Annual Eugenics,* 7:179-188, 1936.

[Freeman, 1969] J. Freeman. Experiments in discrimination and classification. *Pattern Recognition,* 1:207-218, 1969.

[Friesen and Golshani, 1989] O. Friesen and F. Golshani. Databases in large AI systems. *AI Magazine,* X(4):17-19, Winter 1989.

[Gates, 1972] G. Gates. The reduced nearest neighbor rule. *IEEE Transactions on Information Theory,* pages 431-433, May 1972.

[Guha and Lenat, 1990] R. Guha and Douglas. Lenat. Cyc: A mid-term report. *AI Magazine,* XI(3):32-59, Fall 1990.

[Porter *et al.*, 1990] B. Porter, R. Bareiss, and R. Holte. Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence,* 45:229-263, 1990.