Association for Information Systems
AIS Electronic Library (AISeL)

ICIS 1996 Proceedings

International Conference on Information Systems (ICIS)

December 1996

Decision Support in Knowledge Acquisition: Concept Characterization Using Genetic Algorithms

Anita Lee-Post
University of Kentucky

Follow this and additional works at: http://aisel.aisnet.org/icis1996

Recommended Citation
Lee-Post, Anita, "Decision Support in Knowledge Acquisition: Concept Characterization Using Genetic Algorithms" (1996). *ICIS 1996 Proceedings*. 12.
http://aisel.aisnet.org/icis1996/12

This material is brought to you by the International Conference on Information Systems (ICIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in ICIS 1996 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

DECISION SUPPORT IN KNOWLEDGE ACQUISITION: CONCEPT CHARACTERIZATION USING GENETIC ALGORITHMS

Anita Lee-Post
University of Kentucky

Abstract

We demonstrate the use of an unsupervised learning technique called genetic algorithms to discover the association between a concept and its key attributes in concept characterization. The resulting concept-attribute associations are important domain concepts for knowledge engineers to structure interviews with the experts or to prepare representative data for inductive inference. Examples based on the part family identification problem in manufacturing are employed to illustrate the identification capability of our technique. Preliminary results from testing the technique in a SUN SPARC station 1+ indicate that it can be exploited as a decision support tool to assist knowledge engineers in the conceptualization stage of the knowledge acquisition process.

1. INTRODUCTION

Knowledge acquisition is the process of eliciting and structuring reasoning knowledge of human experts and transferring that knowledge into forms that can be used by the inference engine of an expert system. Human involvement has long been regarded as a necessity in knowledge acquisition with the domain expert being the source of knowledge and the knowledge engineer acting as an intermediary agent between the expert and the system. Acquiring knowledge from experts in this manner becomes a bottleneck in the development of expert systems because the interaction between the knowledge engineer and the domain expert is tedious, time consuming and problematic (Feigenbaum 1978; Buchanan and Smith 1989).

On one hand, domain experts have difficulty in explaining their line of reasoning concisely and correctly, or they may state their reasoning knowledge in terms that are too broad for effective machine analysis (Waterman 1986). On the other hand, knowledge engineers often lack sufficient training and education in many behavioral and interpersonal skills that are essential to successful interaction with the experts (Mykytyn, Mykytyn and Raja 1994). More important, it takes significant time and effort for a knowledge engineer to become sufficiently familiar with the problem domain to be able to talk about the domain in terminology adequate for structuring and modeling the knowledge elicited. In fact, one of the most crucial aspects of knowledge acquisition is the conceptualization of domain knowledge because all subsequent knowledge acquisition work builds on this conceptual foundation (Buchanan, et al. 1983).

The aim of this research is to provide knowledge engineers with a decision support tool that will assist them in the conceptualization stage of the knowledge acquisition process. Such a tool is valuable in helping the knowledge engineers to cultivate a background understanding of the domain concepts so that they can communicate intelligently and effectively with the expert. The tool is also useful in reducing the complexity of the problem domain so that increasingly complex applications can be developed with a higher likelihood of success.

2. STAGES OF KNOWLEDGE ACQUISITION

The power of an expert system is derived from the knowledge it possesses (Buchanan and Mitchell 1978). Hence, special attention must be placed on the knowledge acquisition process to ensure the quality and completeness of knowledge so obtained. According to Buchanan, et al., there are five major stages to a knowledge acquisition process, namely, identification, conceptualization, formalization, implementation, and testing. Activities in the identification stage include specifying the problem characteristics, participants, technical and financial constraints, and objective of the system. In the conceptualization

stage, the key concepts and relations of the problem domain are identified and expressed explicitly to form the conceptual base of the expert system. These domain concepts are then formally represented and organized into forms that can be used by the expert system. The formalized knowledge is put together as an executable system in the implementation stage. Finally, the resulting system is tested for faults and errors, as well as fine-tuned for high performance and user satisfaction in the testing stage.

Attempts to reduce human involvement in knowledge acquisition have led to a variety of computerized techniques and tools. A survey of these can be found in Boose (1989). Such tools fall into two categories: (1) support tools for the experts to reduce their dependency on the knowledge engineers and (2) support tools for the knowledge engineers to reduce their dependency on the experts.

Sophisticated interactive programs that allow domain experts to enter their knowledge directly into an expert system belong to the first category. Representative examples are TEIRESIAS (Davis 1979), MORE (Kahn, Nowlan and McDermott 1985), ROGET (Bennett 1985), ASK (Gruber 1988), and QUANTIF (Faure, Frediani and Saitta 1993). These programs generally employ a graphical user interface or an intelligent dialog system. They are highly domain dependent. That is, a domain model needs to be constructed beforehand to provide the basic vocabulary of the domain, identify missing information, integrate new information into the knowledge base, and detect incomplete or inconsistent information. (Harandi and Lange 1990). Involvement of a knowledge engineer is crucial in this conceptualization stage.

Machine learning techniques based on induction have been the most widely used approach in developing tools of the second category. Induction algorithms such as AQ11 (Michalski and Chilausky 1980), ID3 (Quinlan 1986), and CRIS (Liang 1992) are used to produce classification rules that generalize from examples of an expert's behavior without direct involvement of the experts. A prerequisite of induction is the availability of representative historical data that adequately covers classification cases in the application area. To construct meaningful classification rules, care must be taken in selecting a set of suitable attributes on which induction is to be performed. As with the first kind of tool, knowledge acquired in the conceptual stage is imperative for successful application of these knowledge acquisition techniques.

While the conceptualization stage is especially important because all subsequent knowledge acquisition work builds on the resultant conceptual foundation, the majority of support tools are designed for use in the latter stages of knowledge acquisition (Buchanan, et al. 1983). The lack of computerized tools to use at the conceptualization stage of the knowledge acquisition process has motivated us to investigate possibilities in this direction. In this paper, we examine the use of an unsupervised learning technique called genetic algorithms to devise a tool for discovering associations between a concept and its set of attributes.

3. CONCEPT CHARACTERIZATION

In general, a concept is related to a set of attributes or descriptors that allows abstract thinking and inference of new concepts (Clancey 1985). For example, a physician has to identify the characteristics of a patient and connect them to known disease symptoms before treatment can be prescribed. Conceptualization encourages clear thinking because domain terminology is unbundled into well defined or schematically described terms.

The task of identifying the differentiating attributes or descriptors of a concept requires expertise, as pointed out by Sowa (1984):

> The sets of features. . .seem almost obvious once they are presented, but finding the right features and categories may take months or years of analysis. The proper set of categories provides a structural framework that helps to organize the detailed facts and general principles of a system. Without them, nothing seems to fit, and the resulting system is far too complex and unwieldy.

The expertise is usually expressed as practical problem-solving knowledge that allows problems to be restated in terms of stereotypes. In this paper, we demonstrate how an unsupervised learning technique can be employed to acquire such expertise.

The part family identification problem in group technology is used as an illustrative example to show the capabilities of our conceptualization support tool. By automating the identification of key attributes of a concept, we are in essence providing direct support to knowledge engineers in conceptualizing the domain expert's domain knowledge.

4. CONCEPTUALIZATION USING GENETIC ALGORITHMS

Genetic algorithms, the underlying methodology of the proposed decision tool, were developed by Holland (1975) to mimic the natural selection process in living organisms. The idea is to identify the features that account for the successful survival of an individual species over time. Genetic algorithms have been applied to solve various problems, including optimization (DeJong 1975b; Schaffer 1985), game playing (Smith 1980; Axelrod 1987), image recognition (Fitzpatrick, Greenstette, and Van Gucht 1984; Gillies 1985), communication network design (Davis and Coombs 1987), facility layout design (Tam 1992), and scheduling (Biegel and Davern 1990; Badami and Parks 1991; Holsapple, et al. 1993).

In natural selection, species that adapt well to their environment survive. These successful species are described as having a high degree of fitness. Since each species can be differentiated by its specific genetic composition embodied in its chromosome, it is the genetic makeup that accounts for the continual adaptation of a species to its changing environment. The genetic makeup that carries the successful traits for survival and adaptation is acquired and preserved from one generation to the next through three operations: (1) reproduction, (2) crossover, and (3) mutation. Reproduction allows individuals to mate with one another selectively, with the intent of passing successful traits along to the next generation. Crossover allows for an exchange of chromosomal materials between two parents to be combined in their offspring, hence synergistically increasing the degree of fitness in these offspring. Mutation allows random variations in the genetic material of the offspring so as to increase its probability of acquiring successful traits that are not passed along from its parents but through background variation. A general structure for a simple genetic algorithm is shown in Figure 1.

```
Step 1.  Let g=0, g is the generation count
Step 2.  Initialize a starting population, P(0)
             Repeat
Step 3.      Evaluate P(g)
Step 4.      Generate P(g+1) using the genetic operators
Step 5.      Set P(g)=P(g+1)
             Until a stopping criterion is met
```

Figure 1. A General Genetic Algorithm

Viewing the knowledge acquisition problem of concept characterization in the light of natural selection, features that account for the survival of an individual are analogous to the set of relevant concept attributes in conceptualizing the domain knowledge. The degree of fitness of an individual corresponds to the degree of completeness in using a given set of attributes to characterize the domain concept. As a result, using genetic algorithms to solve the concept characterization problem amounts to identifying a set of most relevant attributes leading to a complete characterization of the domain knowledge. The capabilities of our conceptualization support tool will be tested in a part family identification problem in group technology.

5. THE PART FAMILY IDENTIFICATION PROBLEM

The part family identification problem is the first step toward a group technology or cellular production layout in which dedicated machines are arranged in a repetitive production flow configuration to produce a group of similar parts (Hyer and Wemmerlov 1984). The intent is to exploit design and/or manufacturing similarities between parts such that economies of scale can be achieved in the entire manufacturing cycle (Gunasekaran et al. 1994).

The difficulty in part family identification is a lack of methods for providing identification capabilities at which humans can excel through pattern recognition, experience, and intuition (Moon 1992). Table 1 lists a wide range of attributes that can be used as a basis for forming part families.

Table 1. Part Family Identification Attributes

Manufacturing Attributes	Design Attributes
Batch size	Dimensions
Fixtures required	Length/diameter ratio
Machine required	Material type
Operation sequence	Part function
Production volume	Shape
Process	Surface finish

On one hand, design-oriented part families can be exploited to reduce part variety, avoid redesigning existing parts, and facilitate part design retrieval. However, parts with similar design attributes may not necessarily require the same manufacturing process, thus they may or may not belong to the same family (Groover and Zimmers 1984). On the other hand, manufacturing-oriented part families bring such benefits as reduced setup times, lead time, and work-in-process inventory, as well as improved productivity (Hyer 1984). The key to manufacturing-oriented part family identification is how the part is produced, not its appearance. However, part families formed based on manufacturing aspects alone are not useful for other areas of production, in particular, part design and process planning. Therefore there is a need to consider both design and manufacturing attributes in part family identification.

6. ACQUIRING THE PART FAMILY IDENTIFICATION EXPERTISE

6.1 Genetic Algorithms

The expertise needed to determine which and why certain part attributes have the part family differentiation capability is acquired through an unsupervised discovery process based on genetic algorithms. Genetic algorithms were developed by Holland to mimic the natural selection process in living organisms.

6.2 Problem Formulation

Viewing the part family identification problem in the light of natural selection, we can treat parts of different families as individuals of different species. The genetic makeup of each individual corresponds to the set of attributes of each part. Features that account for the survival of an individual are analogous to the set of relevant part attributes that serves as the key to the formation of appropriate part families. A terminology comparison between the natural selection process and the part family identification problem is summarized in Table 2. As a result, using genetic algorithms to solve the part family identification problem amounts to locating the best part family differentiating attributes (most promising path) that leads to the formation of part families (a solution) that best achieves the goal (is globally optimal or nearly so). Note that the corresponding terminologies used in genetic algorithms are in parentheses.

There are five components to any genetic algorithm applications (Davis and Steenstrup 1987):

(1) a coding scheme, which is a chromosomal representation of solutions to the problem,
(2) a way to create an initial population of solutions,
(3) an evaluation function that plays the role of the environment, rating solutions in terms of their “fitness,”

Table 2. Terminology Comparison Between Natural Selection and Part Family Identification

Natural Selection	Part Family Identification
Individual	Part
Species	Family
Genetic makeup	Part attributes
Survival traits	Part family differentiating attributes
Success in survival	Success in forming families that maximizes part similarities

(4) a set of genetic operators responsible for generating a next population of solutions,
(5) a set of instantiated parameters that the algorithm uses, such as population size, stopping rule, probability of mutation.

The subsequent sections explain how these five components are defined when applying genetic algorithms to solve the part family identification problem. A pseudocode of the part family formation procedure is given below.

The Part Family Identification Procedure

Notation:
PFDA: part family differentiating attributes.
g: the generation count to keep track of the number of times the algorithm has been run.
S: population size, which is the number of solution alternatives considered in one generation. A solution alternative is a set of PFDA, also known as a string.
K: number of 1s in a string, which is equal to the number significant attributes
N: number of part families formed using the PFDA.
Pm: probability of mutation.
P(g): g^{th} population, which is a collection of S sets of PFDA in the g^{th} generation.
f(s): fitness value of the s^{th} string, which is the sum of similarities over N part families formed by using the PFDA.

Step 1. Set g=0.
Step 2. Initialize P(g).
2.1 Generate randomly S sets of PFDA in a bit pattern format. The length of the bit pattern is equal to the number of part attributes in a classification code. The significant (insignificant) attributes are represented by 1s (0s).
Repeat
Step 3. Evaluate P(g).
3.1 For each set of PFDA in P(g) Do:
3.1.1 Use Average Link Clustering algorithm to form N part families based on the K significant attributes.
3.1.2 Compute f(s).
3.1.3 Total_f(s)=Total_f(s)+f(s).
3.2 Sort the S sets of PFDA in ascending order of f(s).
Step 4. Generate P(g+1).
4.1 For each set of PFDA in P(g) Do:
4.1.1. Convert f(s) into a number between 1 to 0 by:
4.1.1.1 Divide f(s) by Total_f(s).
4.1.1.2 Accumulate f(s)/Total_f(s) in F.
4.1.1.2 f(s)=f(s)/Total_f(s)+F.
4.1.2 Generate S random numbers between 0 to 1.

4.1.3 Select S sets of PFDA from P(g) such that their f(s) is closest to the random numbers generated. Put the PFDA selected into a storage area called mating pool.

4.1.4 For each set of PFDA in the mating pool Do:

4.1.4.1 Randomly select another sets of PFDA from the mating pool.

4.1.4.2 Generate a random number to indicate the cross over point.

4.1.4.3 Cross over the two sets of PFDA at the cross over point to form one recombined set of PFDA.

4.1.4.4 Reverse the bit pattern in the recombined set of PFDA with a probability of P_m.

4.1.4.5 Put the recombined set of PFDA in P(g+1).

Step 5. Set P(g)=P(g+1).
Until the stopping criterion is met

6.2.1 The Coding Scheme

A binary bit pattern of 1s and 0s is used as our coding scheme. This bit pattern is derived from the design and manufacturing attributes described by the part number. The significant (insignificant) part attributes are represented by 1s (0s). For example, consider the DCLASS coding system (Figure 2) developed at Brigham Young University for educational and research purposes (Bedworth, Henderson and Wolfe 1991):

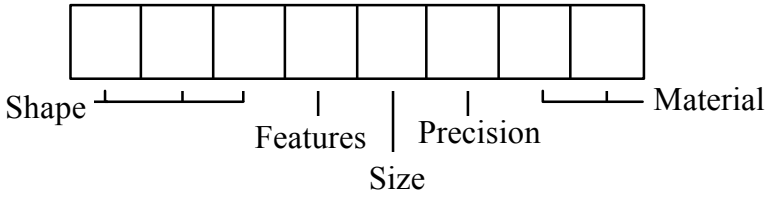

Figure 2. DCLASS Code Structure

The DCLASS coding system uses five attributes to describe a part: shape, features, size, precision and material. A binary pattern of five digits can then be used to convey which part attributes should be considered important in part family formation. Hence, a bit pattern "1 0 0 0 1" means that only shape and material are important in forming part families.

6.2.2 The Initial Population

Once the length of the bit pattern is determined by the coding scheme, we can randomly generate S strings of 1s and 0s, where S denotes the population size and is a given parameter. Each string represents a different selection of part family differentiating attributes. For example, P(0) can be:

	Bit Pattern	**Relevant Attribute(s)**
1	1 0 0 0 1	shape, material
2	1 1 0 0 1	shape, features, material
.	.	.
.	.	.
.	.	.
S	0 0 1 1 0	size, precision

6.2.3 Evaluation of Fitness

The fitness value of each string is a measure of how well the part families are formed. Since the objective of part family formation is to maximize part similarities, a natural measure of part similarities is a similarity coefficient. Hence, maximizing sum of similarities can be used as the evaluation criteria to assess the fitness of each string. This evaluation criteria is expressed mathematically as

$$\text{Max} \sum_{n=1}^{N} S_n = \frac{\sum_{i \in n,\, j \in n} S_{ij}}{C_{N_n}^2}$$

where S_{ij}: Similarity measure between part i and part j

C_{N} : Number of pairwise combinations formed in part family n, and N_n is the number of parts in family n

Our definition of S_{ij} is modified from Offodile (1992) to accommodate alphanumeric part coding, and is defined as follows:

$$S_{ij} = \frac{\sum_{k=1}^{K} S_{ijk}}{K} \qquad (1)$$

$$S_{ijk} = \begin{cases} 1 - \dfrac{|a_{ik} - a_{jk}|}{R_k} & \text{if attribute k is numeric} \\ 1 - \dfrac{\delta(a_{ik}, a_{jk})}{L_k} & \text{if attribute k is alphanumeric} \end{cases} \qquad (2)$$

where,

S_{ij} :	Similarity measure between part i and part j
S_{ijk} :	Similarity measure between part i and part j on attribute k
K :	Total number of attributes considered
a_{ik} :	Part coding for part i on attribute k
a_{jk} :	Part coding for part j on attribute k
R_k :	Range of possible part codings for all parts on attribute k
$\delta(a_{ik}, a_{jk})$:	Number of codes that are not in common between parts i and j
L_k :	Number of codes used to describe attribute k

Part families can be formed using a grouping algorithm called average link clustering (ALC) algorithm (Sneath and Sokal 1973). The steps involved in ALC are shown as follows:

1. Calculate a pairwise similarity coefficient (Sij) for each pair of parts, using equation (1).
2. Form an initial part family by grouping together parts with the largest similarity coefficient.

3. Repeat
   Calculate the average similarity coefficient (Aij) between each pair of elements using equation (3); an element may be a part family or an ungrouped part;
   Group together elements with the largest average similarity coefficient;
   until there is no ungrouped part remaining.

$$A_{ij} = \frac{\sum_{i \in G_1,\, j \in G_2} S_{ij}}{N_i \times N_j} \qquad (3)$$

where,

G_1 : Parts in part family 1

G_2 : Remaining ungrouped parts or Parts in part family 2

N_i : Number of parts in G_1

N_j : Number of parts in G_2

6.2.4 Genetic Operators

Three basic genetic operators are used here, namely, reproduction, crossover and mutation. In reproduction, S strings from P(g) will be selected probabilistically according to their fitness value to enter into a "mating pool," a term given to the place where the selected strings are stored before the crossover operator is applied. Each string in the mating pool will select randomly its partner to participate in a one point crossover. The exact locations of the crossovers will be determined randomly. With a low probability p_m, which is given as a parameter, the offsprings will mutate randomly.

6.2.5 Input Parameters

The input parameters include the stopping rule, the population size and the mutation probability. DeJong has shown that a population size of fifty is sufficient to demonstrate acceptable convergence to optimality in a non-stochastic environment. He has also suggested that a mutation probability of (1/population size) is sufficient (DeJong 1975a). These are the values we will use in our implementation.

In the case of stopping rules, two types of rules can be used:

(1) Time bounded rule which depends on the number of iterations, e.g., the algorithm is terminated after n generations

(2) Quality bounded rule which depends on the magnitude of improvements from iteration to iteration, e.g., the algorithm is terminated if the degree of improvements from one generation to the next is less than 1%.

The quality bounded stopping rule will be used in our implementation.

7. AN EXAMPLE

Consider the following DCLASS codes for five parts:

Part1	A20 1 2 3 A1
Part2	A30 2 1 3 A1
Part3	B01 1 3 2 B2
Part4	B01 1 2 3 A7
Part5	A20 2 1 2 A3

Code number B01 1 3 2 B2 provides the five types of information about Part3:

(1)	Shape (B01)	round with a concentric core
(2)	Feature (1):	has a hole and requires no heat treatment or surface finish
(3)	Size (3):	maximum dimension is 4"
(4)	Precision (2):	degree of tolerance is 0.002-0.01", surface finish is 32-125rms
(5)	Material (B2):	white cast iron.

Given a population size of five, an initial population of five strings of bit pattern can be generated randomly as:

1 0 0 0 1
1 0 0 1 1
0 1 1 0 0
0 0 1 1 0
0 1 0 1 1

A bit pattern of 1 0 0 0 1 indicates that both shape and material are relevant attributes for part family formation. The relevant part attribute list is then reduced to:

Part1	A20 A1
Part2	A30 A1
Part3	B01 B2
Part4	B01 A7
Part5	A20 A3

Based on shape and material similarity, two part families are formed: G1 consists of parts 1, 2, and 5; G2 consists of parts 3 and 4. The resulting similarity measure is 1.977. A different bit pattern will result in the formation of different part families and, hence, different measures of similarities. The essence of our part family identification technique is to use these similarity measures to guide our search for the best part family differentiating attributes (part families so formed have the maximum sum of similarities). In order to illustrate how our technique works, a trace of the part family identification procedure from generation 1 to generation 3 is provided below.

Generation 1

	Pattern	Fitness Value	Probability	No. of selection
1	1 0 0 0 1	1.977	.229	2
2	1 0 0 1 1	1.948	.226	0
3	0 1 1 0 0	0.926	.107	0
4	0 0 1 1 0	1.852	.33	1
5	0 1 0 1 1	1.928	.223	2

Mating Pool	Mate	Cross-Over pt. = 1
1 0 0 0 1	0 1 0 1 1	1 1 0 1 1
1 0 0 0 1	0 1 0 1 1	1 1 0 1 1
0 0 1 1 0	1 0 0 0 1	0 0 0 0 1
0 1 0 1 1	1 0 0 0 1	0 0 0 0 1
0 1 0 1 1	0 0 1 1 0	0 0 1 1 0

Generation 2

	Bit Pattern	Fitness Value	Probability	No. of selection
1	1 1 0 1 1	1.920	.250	2
2	1 1 0 1 1	1.920	.250	0
3	0 0 0 0 1	0.988	.129	0
4	0 0 0 0 1	0.988	.129	0
5	0 0 1 1 0	1.852	.242	3

Mating Pool	Mate	Cross-over pt. = 4
1 1 0 1 1	0 0 1 1 0	1 1 0 1 0
1 1 0 1 1	1 1 0 1 1	1 1 0 1 1
0 0 1 1 0	0 0 1 1 0	0 0 1 1 0
0 0 1 1 0	0 0 1 1 0	0 0 1 1 0
0 0 1 1 0	0 0 1 1 0	0 0 1 1 0

Generation 3

	Bit Pattern	Fitness Value	Probability	No. of selection
1	1 1 0 1 0	1.899	.203	1
2	1 1 0 1 1	1.920	.205	1
3	0 0 1 1 0	1.852	.198	1
4	0 0 1 1 0	1.852	.198	2
5	0 0 1 1 0	1.852	.198	0

Mating Pool	Mate	Cross-over pt. = 2
1 1 0 1 0	0 0 1 1 0	1 1 1 1 0
1 1 0 1 0	0 0 1 1 0	1 1 1 1 0
0 0 1 1 0	1 1 0 1 0	0 0 0 1 0
0 0 1 1 0	1 1 0 1 0	0 0 0 1 0
0 0 1 1 0	1 1 0 1 0	0 0 0 1 0

and so on.

If we stop at generation 3, the highest S_n is 1.977 in generation 1. The corresponding bit pattern is 1 0 0 0 1 indicating that the best part family differentiating attributes should be shape and material.

8. IMPLEMENTATION

The part family identification technique is implemented in C on a SUN SPARC station 1+. Test data based on the above DCLASS classification and coding scheme is used to examine the performance of the technique. Part numbers are randomly generated. The size of the test problem ranges from five parts to forty parts.

The test results in Table 3 indicate that our part family identification technique is not computationally demanding. It took only two seconds to solve a problem of forty parts. Problem 1 in Table 3 is the example we used in section 5. Our technique suggests that part families formed by using the attributes precision and material will have a sum of similarities equal to 1.983,

as opposed to 1.977 using shape and material in the above example. All five test results are confirmed to be optimal (i.e., part families formed using the part family identification attributes suggested by the algorithm exhibit the highest sum of similarities), by comparing to the results obtained from an exhaustive search using all possible combinations of part family identification attributes.

Table 3. Experimental Results

No. of Parts	Sum of Similarities	Part Family Id. Attributes	System Time	Resulting Part Families
5	1.983	Material Precision	0.7 sec	{0 1 3} {2 4}
15	5.666	Shape Feature Precision Material	0.7 sec	{10 12} {0 3 1 13} {4 6 2} {5 8} {7 14} {9 11}
20	6.687	Feature Shape Precision Material	0.7 sec	{7 9 11 2} {8 16} {4 9} {0 3 1 10} {13 14 17 18} {5 6} {12 15}
30	9.331	Shape Feature Precision Material Size	0.9 sec	{14 21 12 17 10} {6 23 7 19 12 24} {8 16} {0 3} {18 29 25 28} {2 9} {1 4} {11 20} {15 26}
40	10.161	Shape Feature Precision Material Size	2.0 sec	{14 21 34 13 17 39} {6 23 7 19 12 24 30 33} {8 16 37} {0 3} {29 38 18 36 29} {25 28 15} {2 9 35} {1 4 22 32} {11 20} {5 27} {10 31}

9. IMPLICATIONS

From the perspective of solving the part family identification problem, unlike the traditional production flow analysis that considers only one attribute, namely, the machine requirements of parts, in forming part families, our approach provides answers to why and how part families should be formed. This will enhance the understanding of the basis of part family formation with the assurance that all relevant attributes of part similarities are fully exploited.

From the perspective of knowledge acquisition, the knowledge of why and how part families should be formed can be discovered in a unsupervised fashion. This will overcome the limitations of other knowledge acquisition techniques such as neural network (Piramuthu, Kuan, and Shaw 1993) or induction (Gennari, Langley and Fisher 1989) where the availability of historical data is a prequisite.

10. CONCLUSION

In this paper, we demonstrate the feasibility of automating the identification of key attributes of a concept by using an unsupervised learning technique called genetic algorithms. While the part family identification problem is chosen to illustrate our technique, it should be clear that none of the underlying ideas are limited to a particular domain. The concept-attribute identification is the first step toward conceptualizing a problem domain. The technique presented in this paper can be incorporated into a decision support tool for knowledge engineers to structure interviews and/or prepare data for inductive inference.

11. REFERENCES

Axelrod, R. "A Evolution of Strategies in the Iterated Prisoner's Dilemma." In I. Davis, Editor, *Genetic Algorithms and Simulated Annealing*. New York: Morgan Kaufamann Publishers, Inc., 1987.

Badami, V. S., and Parks, C. M. "A Classifier Based Approach to Flow Shop Scheduling." *Computers and Industrial Engineering*, Volume 21, 1991, pp. 329-333.

Bedworth, D. D.; Henderson, M. R.; and Wolfe, P. M. *Computer-Integrated Design and Manufacturing*. New York: McGraw-Hill, Inc., 1991, pp. 196-200.

Bennet, J. S. "A Knowledge-Based System for Acquiring the Conceptual Structure of a Diagnostic Expert System." *Journal of Automated Reasoning*, Volume 1, 1985, pp. 49-74.

Biegel, J. E., and Davern, J. J. "Genetic Algorithms and Job Shop Scheduling." *Computers and Industrial Engineering*, Volume 19, 1990, pp. 81-91.

Boose, J. H. "A Survey of Knowledge Acquisition Techniques and Tools." *Knowledge Acquisition*, Volume 1, 1989, pp. 3-37.

Buchanan, B. G.; Barstow, B.; Bechtal, R.; Clancey, W.; Kulikowski, C.; Mitchell, T.; and Waterman, D. A. "Constructing an Expert System." In F. Hayes-Roth, D. A. Waterman, and D. B. Lenat, Editors, *Building Expert Systems*, Volume l. Reading, Massachusetts: Addison-Wesley Publishing Company, 1983, pp. 127-167.

Buchanan, B. G., and Mitchell, T. M. "Model-Directed Learning of Production Rules." In D. A. Waterman and F. Hayes-Roth, Editors, *Pattern-Directed Inference Systems*. New York: Academic Press, Inc., 1978.

Buchanan, B. G., and Smith, R. G. "Fundamentals of Expert Systems." In A. Barr, P. R. Cohen, and E. A. Feigenbaum, Editors, *The Handbook of Artificial Intelligence*, Volume 4. Reading, Massachusetts: Addison-Wesley Publishing Company, 1989.

Clancey, W. J. "Heuristic Classification." *Artificial Intelligence*, Volume 27, 1985, pp. 289-350.

Davis, I., and Coombs, S. "Genetic Algorithms and Communication Link Speed Design: Theoretical Considerations." *Proceedings of the Second International Conference on Genetic Algorithms*, 1987, pp. 252-256.

Davis, L., and Steenstrup, M. "Genetic Algorithms and Simulated Annealing: An Overview." In L. Davis, Editor, *Genetic Algorithms and Simulated Annealing*. New York: Morgan Kaufamann Publishers, Inc., 1987.

Davis, R. "Interactive Transfer of Expertise: Acquisition of New Inference Rules." *Artificial Intelligence*, Volume 12, 1979, pp. 121-157.

DeJong, K. A. "An Analysis of the Behavior of a Class of Genetic Adaptive Systems." Unpublished Ph.D. Dissertation, University of Michigan, USA, 1975a.

DeJong, K. A. "A Genetic-Based Global Function Optimization Technique." *Technical Report No.80-2*, University of Pittsburgh, Department of Computer Science, 1975b.

Faure, C.; Frediani, S.; and Saitta, L. "A Semiautomated Methodology for Knowledge Elicitation." *IEEE Transactions on Systems, Man, and Cybernetics*, Volume 23, 1993, pp. 346-336.

Feigenbaum, E. A. "The Art of Artificial Intelligence: Themes and Case Studies of Knowledge Engineering." *Proceedings of the 1978 National Computer Conference*, Anaheim, CA, 1978, pp. 227-240.

Fitzpatrick, J. M.; Greenstette, J. J.; and Van Gucht, D. "Image Registration by Genetic Search." *Proceedings of IEEE Southeast Conference*, 1984, pp. 460-464.

Gennari, J. H.; Langley, P.; and Fisher, D. "Models of Incremental Concept Formation." *Artificial Intelligence*, Volume 40, 1989, pp. 11-61.

Gillies, A. M. "Machine Learning Procedures for Generating Image Domain Feature Detectors." Unpublished Doctoral Dissertation, University of Michigan, Ann Arbor, 1985.

Groover, M. P., and Zimmers, Jr., E. W. *CAD/CAM: Computer-Aided Design and Manufacturing*. Englewood Cliffs, New Jersey: Prentice-Hall, 1984.

Gruber, T. R. "Acquiring Strategic Knowledge from Experts." *International Journal of Man-Machine Studies*, Volume 29, 1988, pp. 579-598.

Gunasekaran, A.; Goyal, S. K.; Virtanen, I.; and Yli-Olli, P. "An Investigation into the Application of Group Technology in Advanced Manufacturing Systems." *International Journal of Computer Integrated Manufacturing*, Volume 7, 1994, pp. 215-228.

Harandi, M. T., and Lange, R. "Model-Based Knowledge Acquisition." In H. Addi, Editor, *Knowledge Engineering: Fundamentals*. New York: McGraw-Hill, 1990, pp. 103-129.

Holland, J. H. *Adaptation in Natural and Artificial Systems*. Ann Arbor: The University of Michigan Press, 1975.

Holsapple, C. W.; Jacob, V. S.; Pakath, R.; and Zaveri, J. S. "A Genetics-Based Hybrid Scheduler for Generating Static Schedules in Flexible Manufacturing Contexts." *IEEE Transactions on Systems, Man, and Cybernetics*, Volume 23, 1993, pp. 953-972.

Hyer, N. L., and Wemmerlov, U. "Group Technology and Productivity." *Harvard Business Review*, July-August, 1984, pp. 140-149.

Kahn, G.; Nowlan, S.; and McDermott, J. "Strategies for Knowledge Acquisition." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 7, 1985, pp. 511-522.

Liang, T. P. "A Composite Approach to Inducing Knowledge for Expert Systems Design." *Management Science*, Volume 38, 1992, pp. 1-17.

Michalski, R. S., and Chilausky, R. L. "Learning by Being Told and Learning from Examples: An Experimental Comparison of the Two Methods of Knowledge Acquisition in the Context of Developing an Expert System for Soybean Disease Diagnosis." *Policy Analysis and Information Systems*, Volume 4, 1980, pp. 125-160.

Moon, Y. B. "Establishment of a Neurocomputing Model for Part Family/Machine Group Identification." *Journal of Intelligent Manufacturing*, Volume 3, 1992, pp. 173-182.

Mykytyn, Jr., P. P.; Mykytyn, K.; and Raja, M. K. "Knowledge Acquisition Skills and Traits: A Self-Assessment of Knowledge Engineers." *Information and Management*, Volume 26, 1994, pp. 95-104.

Offodile, O. F. "Application of Similarity Coefficient Method to Parts Coding and Classification Analysis in Group Technology." *Journal of Manufacturing Systems*, Volume 10, 1992, pp. 442-448.

Piramuthu, S.; Kuan, C.; and Shaw, M. J. "Learning Algorithms for Neural-Net Decision Support." *ORSA Journal on Computing*, Volume 5, 1993, pp. 361-373.

Quinlan, J. R. "Induction of Decision Tree." *Machine Learning*, Volume 1. New York: Kluwer Academic Publishers, 1986, pp. 81-106.

Schaffer, J. D. "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms." *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, 1985, pp. 93-100.

Smith, S. F. "A Learning System Based on Genetic Adaptive Algorithms." Unpublished Doctoral Dissertation, Department of Computer Science, University of Pittsburgh, 1980.

Sneath, P. H. A., and Sokal, R. R. *Numerical Taxonomy*. San Francisco: San Francisco, 1973.

Sowa, J. F. *Conceptual Structures: Information Processing in Minds and Machines*. Reading, Massachusetts: Addison-Wesley, 1984.

Tam, K. Y. "Genetic Algorithms, Function Optimization, and Facility Layout Design." *European Journal of Operational Research*, Volume 63, 1992, pp. 322-346.

Waterman, D. A. *A Guide to Expert Systems*. Reading, Massachusetts: Addison-Wesley Publishing Company, 1986.