Using the Group Genetic Algorithm for Attribute Clustering

Tzun-Pei Hong  
Department of Science and Information Engineering, National University of Kaohsiung, Kaohsiung, 811, Taiwan, R.O.C. tphong@nuk.edu.tw

Feng-Shih Lin  
Department of Computer Science and Information Engineering, National Sun Yat-sen University, Kaohsiung, 804, Taiwan, R.O.C. m983040076@student.nsysu.edu.tw

Chun-Hao Chen  
Department of Computer Science and Information Engineering, Tamkang University, Taipei, Taiwan, R.O.C. chchen@mail.tku.edu.tw

Abstract—In the past, the concept of performing the task of feature selection by attribute clustering was proposed. Hong et al. thus proposed several genetic algorithms for finding appropriate attribute clusters. In this paper, we attempt to improve the performance of the GA-based attribute-clustering process based on the grouping genetic algorithm (GGA). In our approach, the general GGA representation and operators are used to reduce the redundancy of chromosome representation for attribute clustering. At last, experiments are made to compare the efficiency of the proposed approaches and the previous ones.

Keywords—attribute clustering, feature selection, genetic algorithm, grouping genetic algorithm, data mining.

I. INTRODUCTION

In data mining and machine learning, feature selection is an important pre-processing step [5][8]. A proper subset of features can not only reduce execution time of deriving rules [2], but also improve accuracy of classification. In order to conquer the curse of high dimensionality, some feature selection techniques have been proposed [1][17]. However, finding an optimal feature subset has been shown to be an NP-hard problem [3]. In 2007, Hong and Liu proposed a feature selection approach based on the concept of feature clustering [10]. Based on the same idea, Hong and Wang [13][14] then proposed the GA-based clustering methods for attribute clustering to find approximate feature subset for classification.

However, as Falkenauer pointed out, the general GA had some weakness when solving the grouping problems [6]. Due to the encoding scheme, multiple chromosomes would map to the same attribute clustering result (feasible solution) due to the combinatorial property, thus causing a larger search space than needed. Thus, Falkenauer [6] proposed a group genetic algorithm (GGA) as a new evolution algorithm to ease the problem. GGA has the same workflow as GA, but uses different encoding schema and different operators. It has been testified that the efficiency of GGA is superior to GA in some areas especially on grouping problems [4]. In this paper, we thus propose a GGA-based attributed clustering approach.

II. REVIEW OF RELATED WORK

The purpose of feature selection is to find a proper subset of features that are relevant to the target concept. The dependency measure was used to estimate the similarity between each attributes which was proposed by Han et al. [9] and Li et al [11]. Hong and Liu used the dependency measure in their attribute clustering based feature selection approaches [10]. The attributes which provide similar contribution to the classification have high dependency to each other.

Hong and Wang [14] proposed a GA-based clustering method for attribute clustering to find approximate feature subsets for classification. They first proposed an approach which considered the average classification accuracy and the cluster balance of the attribute clusters, which was represented by chromosomes, as the fitness evaluation criteria. The fitness function adopted could get a good trade-off between accuracy and cluster balance.

Many methods of using GAs to solve grouping problems have been proposed before [12]. Some problems, however, exist for the standard GA to solve grouping problems. Two main weaknesses of GAs on grouping problems are described below. First, a standard encoding scheme of GAs is highly redundant on grouping problems. The second weakness is that the classical GA crossover operator can't assure the inheritance property of the offspring from their parents. Since the traditional GA approach has some weaknesses as mentioned above when applied to the grouping problems, Falkenauer thus proposed the grouping genetic algorithm (GGA) to improve it [7]. Pankratz employs an adaptation of GGA for Vehicle Routing Problem [15], and Rekiek applied GGA on the Handicapped person transportation problem [16]. Falkenauer's experiments results showed that GGA did better than GA on these problems [6]. Brown and Sumichrast [3] also did some empirical tests about the performance of GA and GGA in different domains. Their results also indicated GGA was superior to GA for big grouping problems.

GGA and GA have nearly the same procedure. But GGA adopts a different encoding scheme and different genetic operators. In the following paragraph, we will briefly introduce GGA's encoding scheme and genetic operators.

In Falkenauer's GGA representation, a chromosome consists of two parts, an object part and a group part. The object part stores the information about how the objects are grouped, and the group part is an ordered list of the groups. The object part is formed by a fixed-length string, with each gene in
the string representing a group label of an object. For example, consider an object part in a chromosome: ACBBA. This chromosome represents that the first and the fifth objects belong to group “A”, the third and the fourth objects belong to group “B”, and the second object belongs to group “C”. On the contrary, the group part is the main difference part from the traditional GA. It stores all the group names in a sequence of variable length. In GGA, a chromosome represents how the given objects are grouped. The group tags are just used to differentiate which objects are in the same group, and which aren’t, and hence the same group tag of objects in different chromosomes doesn’t mean any relation between the groups. An example for a complete chromosome is shown below:

ACBBA: ABC.

Different from the GA crossover, the GGA crossover is based on the groups instead of on the objects. For example, assume there are two chromosomes as:

ADBCCCAD: ACBD, and abbccddd: abdc.

First, the inserted position in the group part of the first parent and a random section of the second parent are selected. Assume the inserted position of the first parent is the bar position shown below:

A | CBD (The inserted position is between A and C).

Also assume the selected section of the second parent is the area between the two bar positions shown below:

ad | bc (The selected section is “bc”).

Then the section “bc” in the second parent will be inserted into the first parent, with the group part of the new chromosome shown as:

A bc CBD.

Then the duplicated objects in the new chromosome are checked. The chromosome will then become:

ADbbbcc: AbcD.

After this step, the new chromosome may have more or less groups than defined. The adjustment of the chromosome to make the group number equal to the desired is then executed if necessary.

Mutation adopted in GGA includes three different strategies, which are creating a new group, eliminating an existing group, and exchanging items among groups. Besides, the inversion operator is also used in GGA to change the order of the genes in the group part, in order to make some groups have better chances to be transmitted to offspring. These two operators are useful to avoid being trapped in local optimal solutions.

III. THE PROPOSED APPROACH

The purpose of this paper is to divide the whole feature set into K appropriate feature subsets, such that feature selection and replacement can be easily achieved with a good performance. Here K is a predefined constant number. We will adopt the grouping genetic algorithm (GGA) proposed by Falkenauer to find proper attribute groups [6]. The chromosome representation of the proposed approach is described below.

A. Chromosome Representation

In the proposed attribute clustering approach, each chromosome represents a possible attribute clustering result. Let a feature set A consist of n features, denoted \( \{a_1, a_2, \ldots, a_n\} \). If the objective is to select K features from A, then a partition with K attribute groups will be formed. The final feature set will include the features, with each selected from a group. Formally, let the i-th group of features be denoted \( G_i \). Then \( G_1 \cup G_2 \cup \ldots \cup G_k = A, G_i \neq \emptyset \) and \( \forall i \neq j, G_i \cap G_j = \emptyset \). An attribute chromosome will include the partition information of the K groups \( \{G_1, G_2, \ldots, G_k\} \). As mentioned above, a GGA chromosome consists of two parts: attribute part and group part. For the problem of attribute clustering, the attribute part stores the information about how the attributes are grouped, and the group part is an ordered list of the attribute groups. The attribute part is a fixed-length string. Each gene in the attribute part represents a label of an attribute. For example, consider an attribute part: ABBCAC. This chromosome represents that the first and the fourth attributes belong to group “A”, the second and the third attributes belong to group “B”, and the fifth attribute belongs to group “C”. The other part of a GGA chromosome is the group part, which is the main part of a chromosome for genetic operators of GGA to work with. The group part stores all the group names in a string of length K. An example for a chromosome is shown below:

ABBCAC: ABC.

In the above chromosome, the attribute part is located before the semicolon, and the group part is after the semicolon. In the attribute part of the example, there are five attributes partitioned into three groups. The names of the groups are recorded as a sequence after the semicolon. For three groups, the sequence can be any combination of the three symbols ‘A’, ‘B’ and ‘C’. For example, it may be “ABC”, “BAC” or other combinations. The sequence is randomly assigned when a chromosome is produced at the initial population-generation process.

In GGA, a group name is used to differentiate which attributes are in the same group and which are not. Hence, two genes with the same group name but in different chromosomes don’t have any relation. In practical implementation, we use the concept of a set instead of a string to store the group information in the attribute part. Below, a simple example is given to illustrate the above idea.

Example: Assume there are six attributes \( \{a_1, a_2, \ldots, a_6\} \) to be divided into three groups. Thus, \( K = 3 \) and \( N = 6 \). Suppose there is a chromosome: ABBAC which can be represented in Figure 1 for implementation. In this chromosome, the two attributes \( a_1 \) and \( a_3 \) are in the same group, the three attributes \( a_2, a_5 \), and \( a_6 \) are grouped together, and \( a_4 \) itself is in a group. Hence, there are three sets \( \{(1, 4), (5), (2, 3, 6)\} \) in Figure 1 to represent the attribute part. The sequence, “ABC”, is stored to represent the group part of the chromosome.
Figure 1: An example of a GGA chromosome

B. Initial Population

At the beginning, a population of chromosomes is randomly generated. Assume that we want to partition $N$ features into $K$ groups. First, $K$ empty groups $G = \{G_1, G_2, \ldots, G_K\}$ are generated. Then, the $N$ features are randomly assigned to the groups, with one feature to an arbitrary group. If there still exists an empty group after all the attributes are assigned, a group is then selected at random from the set

$$\{ G_i \| |G_i| > \frac{N}{K}, G_i \in G \},$$

and is split into two groups randomly. The above process is repeated until there are $K$ non-empty groups.

C. Fitness and Selection

In the proposed approach, we use the fitness function proposed by Hong and Wang [19] to find proper feature subsets. The fitness function consists of two factors, cluster accuracy and cluster balance. The two factors are briefly described in the following paragraphs.

The cluster accuracy is an evaluation of the average classification accuracy of all possible attribute subsets of a chromosome on the given training dataset. An attribute subset which has high classification accuracy means that we can precisely classify an object to the correct class by the attributes in the feature subset. Formally, the accuracy of a chromosome $C_i$ is defined as follows:

$$\text{accuracy}(C) = \sum_{j=1}^{NC} \text{subAccuracy}(S_j) / NC,$$

where $NC$ is the number of attribute combinations resulting from the chromosome $C_i$, and $\text{subAccuracy}(S_j)$ is the accuracy of the $p$-th possible attribute combination $S_j$, for the given training dataset.

Another evaluation factor is the group balance. This factor is used to make the groups represented by the chromosome have as close attribute numbers as possible. If a chromosome is unbalanced, a new object with missing value may not be correctly classified since no other alternative attributes can be used in the group with the missing attributes. Formally, the factor of cluster balance for a chromosome $C_i$ is defined as follows:

$$\text{balance}(C_i) = \frac{K}{\sum_{i=1}^{K} \frac{|\text{group}_i|}{N} \log \frac{|\text{group}_i|}{N}}.$$ 

where $|\text{group}_i|$ represents the attribute number in the $i$-th group. It basically follows the idea of entropy and the more balanced a chromosome is the higher the value the factor will get.

According to the above discussion, the fitness function is thus defined as follows:

$$f(C_i) = \text{accuracy}(C_i) \times [\text{balance}(C_i)]^\alpha,$$

where the parameter $\alpha$ is used to control the relative influence of the above two factors. It can be set with different values under different circumstances.

D. Crossover

The GGA crossover operator takes an arbitrarily chosen chromosome as a base chromosome, and then inserts some groups from another chromosome. It then eliminates duplicated attributes from the newly formed chromosome. Formally, assume there are two following chromosomes: $C'$ as a base chromosome and $C''$ as an inserted chromosome:

$$C' = G'_1 G'_2 \ldots G'_p G'_{p+1} \ldots G'_K,$$

$$C'' = G''_1 G''_2 \ldots G''_s G''_{s+1} \ldots G''_{s+1},$$

where $p$ is the inserted point, $s$ is the start group in the inserted segment and $m$ is the inserted segment number. The three parameters of $p$, $s$ and $m$ are generated randomly when a crossover operation is executed. The new offspring $C_{new}$ is then generated as follows:

$$C_{new} = G'_1 G'_2 \ldots G'_p G'_{p+1} \ldots G''_s \ldots G''_{s+1},$$

$$C''_i = C'_i - (G''_i \cup G''_{i+1} \cup \ldots \cup G''_{j+m-1}),$$

Besides, $G''_i$ will be eliminated if $G''_i = \phi$. Thus, after the elimination process, there will be three different situations: $|C''_{new}| < K$, $|C''_{new}| = K$ or $|C''_{new}| > K$, where $|C''_{new}|$ is the number of non-empty groups in the new chromosome. The second situation is the ideal result, and the other two needs further adjustment to fit the constraint.

If $|C''_{new}| < K$, then a new group will be randomly selected from $\{ G_i \| |G_i| > \frac{N}{K}, G_i \in G \}$, and then split into two subgroups.

This process is repeated until there are $K$ non-empty groups. Another situation is the third case where $|C''_{new}| > K$. For this case, the roulette wheel selection strategy is used to select the groups to remove. When a group is selected for removal, the attributes inside it are randomly reassigned to another group. The group with less attributes has a higher probability to be selected for removal. This operation is designed from preserving the parents' characteristics. We would like to keep the offspring similar to their parents. If a group with more attributes is removed in the above operation, then the offspring will be more different from its parents.

E. Mutation and Inversion

The mutation operator works on the object part only by randomly reassigning an attribute into another group. In addition to the mutation operator, another group genetic operator is the inversion operator. It is designed to help the crossover operator select different combination of groups to exchange between two parents. This can be done with random or purpose-driven rearrangement of the positions of groups. The groups which are close in the sequence position have more chances to transfer together. In our approach, the rearrangement is done randomly.
The Proposed Algorithm

According to the above description, we can design a GGA-based algorithm for attribute clustering as follows.

The Algorithm:

INPUT: A training dataset with $N$ attributes and the number $K$ of clusters.

OUTPUT: An appropriate attribute-clustering result.

STEP 1. Randomly generate a population of $P$ individuals, with each being a feasible attribute-clustering result.

STEP 2. Calculate the fitness value of each chromosome $C_i$ by the following substeps:

STEP 2.1. Calculate the average accuracy of all possible attribute combinations of the chromosome.

STEP 2.2. Calculate the cluster balance of the chromosome which is decided by the attribute number in the group. A chromosome has a higher balance value when its clustering result is more balanced.

STEP 2.3. Integrate the value from Steps 2.2 and 2.3 to evaluate the fitness of a chromosome by the following formula:

$$ f(C_i) = accuracy(C_i) \cdot \left[ balance(C_i) \right]^\alpha, $$

where $\alpha$ is the parameter to control the influence of the two factors.

STEP 3. Execute the GGA crossover operation.

STEP 4. Execute the GGA mutation operation.

STEP 5. Execute the GGA inversion operation.

STEP 6. Calculate the fitness values of the new chromosomes.

STEP 7. Select the chromosomes as the next generation by using the roulette wheel selection strategy.

STEP 8. Repeat Steps 3 to 7 until the termination criterion is satisfied.

STEP 9. Output the chromosome which has the best fitness value.

IV. EXPERIMENTAL RESULTS

In this section, the experimental results by the proposed algorithm for clustering attributes are described. One real-world data set, the Single Proton Emission Computed Tomography (SPECT), was used in the experiment. The characteristic of the dataset was shown in Tables 5-1. The experiments were implemented in the C++ language on an Intel(R) Core(TM)2 Duo CPU E8400 with 3.00GHz CPU and 4GB RAM.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Features</td>
<td>22</td>
</tr>
<tr>
<td>Number of Instances</td>
<td>267</td>
</tr>
<tr>
<td>Number of Classes</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 5-1: The characteristics of the SPECT dataset

A Experimental Results of the Approach

In this section, the GGA-based attribute clustering algorithm is compared with the GA-based attribute clustering. The real world dataset, SPECT dataset, was used to verify our improvement. In this experiment, the initial population size $P$ was set at 10, the mutation rate $P_m$ was set at 0.05, 18 features are sets as the input features set, and 4 irrelevant features are not considered. The objective is to select four features for classification. The experiments were run 10 times for each algorithm. Figure 2 shows the comparison between our approach and the GA-based approach. The fitness value trend shows that our approach converges faster than the GA-based approach.

![Figure 2: The fitness value trend for the SPECT dataset](image)

V. CONCLUSION AND FUTURE WORK

Since the traditional GA approach has some weaknesses when applied to the grouping problems. In this paper, we introduce the grouping genetic algorithm to find a feasible attribute clustering result to improve the performance of the GA-based attribute-clustering process. The proposed approach can speed up classification time and reduce cost by selecting a proper feature subset. Besides, the algorithm can deal with the missing-value problem because they can replace attributes with missing values by other attributes in the same clusters.

In the future, we will try to improve the performance of the fitness evaluation and take the advantage of the user’s background knowledge to improve the converge speed and result accuracy. We will also try to develop a new algorithm which can learn the cluster number and discover irrelevant features during the process.

REFERENCES


