Chapter 4

Granulation Using Clustering and Rough Set Theory & its Tree Representation

4.1 Introduction

Granular computing (GrC) [Zadeh1998] emerges as a new multidisciplinary study and has received much attention in recent years. Granular computing deals with representing information in the form of some aggregates (that embrace a number of individual entities) and their ensuing processing [Pedrycz2001]. The notion of granularity itself is defined and quantified. Granular computing provides a systematic natural way to analyze, understand, represent and solve the real world problems [Pedrycz2001]. With granular computing, one aims at structured thinking at the philosophical level, and structured problem solving at the practical level [Yao2005a].

In Chapter 3 a method for discretization using Rough Set Theory and clustering was proposed. Using the concepts given in the Chapter 2, this chapter presents a novel method for granulation based on clustering and Rough Set Theory which is a set theoretic approach. To present the result of the granulation process Granulation Tree (GT) has also been proposed in this chapter. Rest of the chapter is organized as follows: Section 4.2 introduce the granular computing, Section 4.3 described the proposed method for granular computing. Section 4.4 discusses the presentation of granulation result in a tree structure. Results and analysis of the
experiments are presented in Section 4.5 and finally Section 4.6 concludes the chapter.

4.2 Granular Computing

Hobbs proposed a theory of granularity based on the observation that "we look at the world under various grain sizes and abstract from it only those things that serve our present interests" [Hobbs1985]. Furthermore the ability to conceptualize the world at different levels of granularities and to switch among these granularities is fundamental to human intelligence and flexibility. This enables to map the complexities of real world into computationally tractable simpler theories [Hobbs1985].

The concept of Granular computing first introduced by L.A. Zadeh's, in 1979, appeared under the name of information granularity in paper [Zadeh1979]. The term "granular computing" was suggested by T. Y. Lin in the discussion of BISC, Special Interest Group on Granular Computing [Zadeh1998]. Granular computing is an umbrella term to cover many theories, methodologies, techniques, and tools based on core of granules for problem solving [Zadeh1997], [Yao2000]. The idea of granular computing earlier investigated in artificial intelligence through the notions of granularity and abstraction. Hobbs proposed a theory of granularity [Hobbs1985], which is similar to the formulation of theory of rough sets. The theory perceived and represented the universe of problem under various grain sizes, only to abstract those

Granulation using Clustering and Rough Set Theory & its Tree Representation 70
things that serve the present interests. Since then, the concept of Granular computing is rapidly developing with growing interest in the topic.

GrC is a relatively new term in problem solving in computer science, and may be viewed more on the conceptual rather than technical level. A number of methods and models of granular computing have been proposed and studied. Basic concepts of granular computing are granules such as subsets, classes, clusters etc of the universe. The basic notions and principles of granular computing occur under various guises in a wide variety of fields [Yao2004], [Lin2003]. Belief functions, artificial intelligence, cluster analysis, chunking, data compression, databases, decision trees, divide and conquer, interval computing, machine learning, structure programming, quantization, quotient space theory, and rough set theory are some example fields. Much research has been conducted recently in various aspects of granular computing [Yao2005]. Following are three approaches for granulation in the literature [Yao2004],

- **Zadeh's formulation**: A general framework for granular computing based on fuzzy set theory was introduced by Zadeh [Zadeh1997]. Here the granules are constructed and defined based on the concept of generalized constraints. The relationships between granules are represented in terms of either the fuzzy graphs or the fuzzy if-then rules. The associated computation method is also known as computing with words (CW) [Yager1998], [Zadeh1996].

- **Powlak’s Rough Set formulation**: With granulation of the universe, one considers the elements within a granule as a whole rather than as individuals [Zadeh1979]. The loss of information due to granulation implies that some
subsets of the universe can only be approximately described. The theory of rough sets mainly deals with the approximation aspect of information granulation [Pawlak1998].

- **Set theoretic:** Set theoretic model [Yao2000] has been proposed using a binary relation over the power set of the objects to represent granulation. Each granule represents a concept, as each element of the granule is an instance of the concept. Yao has also presented a Partition Model for Granular Computing [Yao2004a] which is basically an extensive model of set-theoretic framework.

In all the literature surveyed by the authors granular computing in general has been presented as a model similar to the ability of humans to perceive the world at different granularity and to change granularities in problem solving. However, a need to implement the general models has continued. It has also been suggested by Yao that *each field may develop its theories and methodologies in isolation* [Yao2004a]. Therefore, implementations of other existing models have not been considered for comparison. Various aspects, methods and frameworks for granular computing without separating the process of granulation from its end use have been presented in the literature. In the next section a method for granulation has been proposed. In the proposed method, concept of partitioning as given in Chapter 3 has been used.

**Information granulation**

A granule may be interpreted as one of the numerous small particles forming a larger unit. Collectively, they provide a representation of the unit with respect to a
particular level of granularity. Thus a granule may be considered as a localized view
or a specific aspect of a large unit. Information Granulation involves partitioning a
class of objects into granules i.e. clumps of objects, which are drawn together by
indistinguishability, similarity or functionality [Zadeh1997a, Lin1998]. The phrase
“drawn together by indistinguishability, similarity, or functionality” has been
developed as the concept of binary granulation [Chiang2005] as defined below.

**Binary Granulation:** Binary granulation is the association of an object $p \in U$ with the
granule $B(p) \subseteq U$ (neighborhood), where $p$ varies through all objects of the universe
$U$. This association is the mapping $U \rightarrow 2^U$, called as basic or binary granulation
(BG).

**Binary Relation:** A relation $R$ defined as $R = \{(p, v) \mid v \in B(p), p \in U\}$ is the Binary
Relation (BR) defined using the binary granulation (BG).

**Binary Neighborhood System:** The collection $\{B(p) \mid p \text{ varies through } U\}$, where
$B(p)$ the granule of $p$ also referred as neighborhood of $p$, is called the Basic
Neighborhood System or Binary Neighborhood System (BNS).

The geometric and algebraic views of binary granulation are represented by BNS and
BR respectively.

**Fine Granule:** In the absence of concepts that evaluate the quality of granules or
process granulation in the literature, a measure to determine the quality of granules is
proposed below for the work in this chapter.
"A fine granule can be measured by the type of object it contains. If a granule contains all the object belonging to one class or category then it is defined to be a fine granule, and coarse otherwise”.

4.3 Proposed Method

The proposed method comprises of two phases. In the first phase the natural partition of continuous attribute values are obtained and in the second phase granules are formed using the partitioned obtained in first phase.

Phase I:

The basic neighborhood system is implemented using the clustering technique at attribute level in the first phase. This phase implement the neighborhood system by partitioning the attribute values in to smaller interval using density based approached. A detailed discussion of partitioning method is presented in Chapter 3. The method for discretization as proposed in Chapter 3 consists of two steps: In the first step to partition the attribute values into the natural non-overlapping block using the density based clustering approach, and in the second step the partitions are refined using the Rough Set Theory. Three threshold values MaxPoint, MinPoint, and MaxLength are used in the proposed method to control the quality of the partition. To partition the continuous attributes discretization algorithm given in Figure 4.1 is used. After partitioning all the attribute granule formation process is undertaken phase II which has been discussed in the next subsection.
Step 1. for each continuous attribute $A_i \in A$

1.1 Select distinct values of $A_i$ in an array $Distinct$

1.2 Sort($Distinct$)

1.3 Call DBSCAN($Distinct$, $Eps$, $MinPts$) to produce $\{I_1, I_2, ..., I_n\}$

1.4 Refine($I_1, I_2, ..., I_n$)

Step 2. for each attribute $A_i \in A$

Partition each (sub)space based on the discretized values of $A_i$ if partition gives the fine granule than previous one.

Step 3. Stop

Figure 4.1: Discretization Algorithm

\[
\text{Refine}(I_1, I_2, ..., I_n)\\
\]

If labeled dataset then call Refine_l($I_1, I_2, ..., I_n$) to get the optimal intervals for the specified parameters MinPoint, MaxPoint and MaxLength.

Else call Refine_ul($I_1, I_2, ..., I_n$) to get the optimal number of intervals with respect to the specified parameters MinPoint, MaxPoint and MaxLength.

Figure 4.2: Refine function
The functions \( \text{Refine}_I(l_1, l_2, \ldots, l_t) \) and \( \text{Refine}_\text{ul}(l_1, l_2, \ldots, l_t) \) are defined in Chapter 3. The function \( \text{Refine}(\cdot) \) is applied attribute wise to the results of partition by DBSCAN. In this function if an interval is large then it will be break in two intervals and if an interval is small then it will be merged to nearest suitable interval. To obtain partition of a big interval, \text{cut point()} function is used to find the best suitable cut point to split a big interval to maximizing \( \text{POS(D)} \) of the attribute.

**Phase II:**

In the second phase the construction of granules is carried out. Binary granulation is used for this purpose. First choose an attribute and partition the space corresponding to the intervals of the corresponding attribute values. This step is repeated for all attribute. Partitioning of a (sub)space is continued only when there is an improvement in the quality of the granules. To measure the quality of the granules we propose parameters two parameters namely- precision of a granule and ratio of the size of partitions. Precision of a granule \( G \) is given by

\[
\text{Pres}(G) = \frac{\max \{ \| X_{c_p} \| : X_{c_p} \subseteq G, \forall y \in X_{c_p} D(y) = c_p \} }{|G|} 
\]

and,

the ratio of the partition denoted by \( R(G) \) is,

\[
R(G) = \frac{\min \{ |G_i| \}}{\max \{ |G_i| \}}, G_i's \text{ are the partitions of } G
\]

\( G \) will be split only if the value of \( R(G) \) is less than a threshold value.
4.4 Granulation Tree

"Granulation of an object A, leads to a collection of granules of A, with a granule being a clump of points (objects) drawn together by indistinguishability, similarity, proximity or functionality". Zadeh 1997.

Study of data structure and knowledge representation has been pivotal in advancement in computer science. It is very important to represent, store, access and display the output of any process for good interpretability and usability. The result of granulation must therefore be represented in such a way that the user can visualize the granules and can use them for further processing. A tree structure has been proposed in this chapter to represent results of an attribute oriented granulation scheme. Tree is one of the most efficient data structure to encapsulate the notion of hierarchy. It has good interpretability and efficient algorithms exist for dynamic processing of data objects stored in the node. In the proposed tree structure all the nodes of the tree represent information granules corresponding to the given data. The root node represents the universe. The tree is constructed by decomposing the universe based on the sequence of discrete values of the attributes, considering them one at a time. The granules get finer as path from root towards leave is traversed and coarser when traversing in the opposite direction. A granule is described by the set of ordered pair <attribute, attribute values> which is unique for each granule represented by the nodes of the granulation tree.
Granule Descriptor: A granule at level-$i$, $i \geq 1$ is described by a list of $i$ pairs $<\text{att}, \text{val}>$ or the descriptor of size $i$. Thus for a granule at level-$i$ the $j^{th}$ descriptor is given by

$$d_j^i = \{<A_k, v_{jk}>\}, \text{ for } v_{jk} \in \text{Dom}(A_k) \text{ and } 1 \leq k \leq i$$

4.4.1 Granulation Tree Construction

The root of the granulation tree represents the universe a single clump consisting of all objects. Let set of attributes $\{A_1, A_2 \ldots A_m\}$ have $n_1, n_2 \ldots n_m$ as number of discretized values respectively. From the relational algebra, the number of possible unique tuples based on these attribute is equal to $n_1 \times n_2 \ldots \times n_m$. Consider one attribute to obtain the partition of the universe into subspaces. The space is Partitioned the basis of the discrete values of the attribute. Each subspace corresponds to an attribute value and is represented as a child node. The descendents at depth-1 of the granulation tree represent the information granules of level-1. The process of granulation is continued by considering one unique attribute for all the nodes at a given level, thus recursively partitioning the subspace. The partition of each subspace at depth $i$ is represented by the set of all its child nodes at depth- $i+1$ i.e. the granules at level $i+1$. Each child node represents a finer granule than its parent. An attribute once used for granulation is not reconsidered. A granule at a node at level-$k$ may be denoted by the granule descriptor, a set of $k$ ordered pairs of $<\text{attribute}, \text{attribute values}>$ along the path from root to the node and size $|G|$. A tree of depth $k$ may represent all the granules of levels less than or equal to the granules of level-$k$. 
Therefore given a set of \( m \) attributes, a granulation tree of depth \( m \) can represent all granules of level-\( k \) of the universe, for \( 1 \leq k \leq m \). The set of granules of a given dataset represented by the all the leaf nodes is unique, irrespective of the order in which the attributes are selected.

By introducing controls on the growth of a granulation tree a control the number of granules may be imposed. For a labeled dataset a granule containing the objects of same class should not be further partitioned. But in case of unlabeled data a threshold value of average dissimilarity between the objects may be used to terminate granulation tree construction. Thus Pres may be used to measure the quality of the granules during granulation tree construction.

In the theorem 4.1 it has been proved that set of granule is same, irrespective of the order of selection of attributes considered for partitioning the space i.e. granulation process is independent of order of attribute taken for partition of the space.

**Theorem 4.1 :** For \( U \), a universe with \( m \) attributes the maximum number of granules at level \( m \) is same whatever may be the order of attribute selection for partitioning the space. Moreover the granule descriptor for each granule is unique.

**Proof:** Let \( n_1, n_2, \ldots n_m \) be the number of discretized values for the attributes \( A_1, A_2, \ldots A_m \) respectively. Therefore the number of possible unique tuples based on these attribute is equal to \( n_1 \times n_2 \times \ldots \times n_m \) irrespective of the order of attributes considered to represent the tuple.

*Granulation using Clustering and Rough Set Theory & its Tree Representation* 79
Now we show that set of granule descriptor is same. Let \( \{d_1, d_2 \ldots d_N\} \) be the set of descriptors of \( N \) granule when attributes are consider for partition in \( A_1, A_2 \ldots A_m \) order. Where \( N = n_1 \times n_2 \times \ldots \times n_m \) and \( d_i = \{<A_1, V_{i1}>, <A_2, V_{i2}> \ldots <A_m, V_{im}>\}, V_{ij} \in \text{dom}(A_j). \)

Now suppose the order of attribute has been change and it is \( A_2, A_1 \ldots A_m \) then granule represented by \( d_i \) is will be represented by the \( \{<A_2, V_{i2}>, <A_1, V_{i1}> \ldots <A_m, V_{im}>\} \). Similarly for every \( d_i \), one can have different sequence of descriptor using all \( m \) attribute in a different granulation tree, where the order of selection of attribute is altered. However the set of \( m \) descriptor will remain unique for a given granule.

### 4.4.2 Granulation Tree Representation

In computer a tree is represented by a set of node. Node contains the information about the child, parents and other key information. Granulation tree is also represented by the set of nodes. The node of the granulation contains the following information:

a. Pointer to the parent node,

b. A pair of name of attribute and attribute value corresponding to which a particular node is obtained from the parent node,

c. An array of pointer to represent child nodes, and
d. An array to store the number of objects in different classes or an integer to represent the total number of objects contained in the granule represented by that node. Array is used in case of labeled data and integer is used in case of unlabeled data. To represent the attribute and attribute value, a structure pair is defined as below:

```c
typedef struct pair
{
    int class;    /* Name of attribute */
    int noOfObject;
};
```

So the node of granulation tree of labeled data is given by

```c
struct node
{
    struct *parent;
    char att[20];
    int attval;
    pair a[];
    struct node *child[];
};
```
Similarly the node of granulation tree of unlabeled data is given by

```c
struct node
{
    struct *parent;
    char att[20];
    int attval;
    pair noOfObject;
    struct node *child[];
}
```

Following is the code for granulation tree creation:

```c
GT_Creation(Node *T, )
{
    T=(Node*) malloc( sizeof(Node));
    T->parent=NULL;
}
```

Following is the code for adding the record in the granulation tree:

```c
GT_Add_Record(Node *T, R) /* R is the record represented in vector form And value of the attribute are given in same order in which attribute are considered to form the Granulation Tree, R={v1, v2 ... vn} */
{
    for i =1 to n
        if node corresponding to the path v1 to vi exist then increment the noOfObject or corresponding pair as the case may be.
```
else
    add a node in the T corresponding to \( v_i \) as a child node of \( v_{i-1} \) with set 1 to
    noOfOject or corresponding pair as the case may be.

})

4.5 Result & Analysis

Proposed scheme for granulation has been implemented in C language. The
experiments were carried out on the following four Labeled datasets obtained from
the UC Irvine ML repository [WWW]. To check the feasibility of the approach the
following data sets, relatively small in size have been considered.

1. Iris Plants dataset (iris),

2. Johns Hopkins University Ionosphere dataset (ion),

3. Statlog Project Heart Disease dataset (hea),

4. Pima Indians Diabetes dataset (pid).

Two out of four datasets, namely iris and pid, are described by continuous
attributes where as ion and hea have both the continuous and the discrete attributes. A
detailed description of the data sets is presented in Table 3.1 in Chapter 3. The
property ‘Cont/Mix’ describes the attribute types of the data set. ‘Cont’ indicates that
all the attributes have continuous values whereas ‘Mix’ denotes that some attributes
have continuous and some have nominal values.
The result of proposed method has been presented in Table 4.1. The result has been shown under four headings-

1. **Number of Granules**: The number of granule gives the total number of granules obtained by the granulation process by constructing the granulation tree. Minimum number of granule is required provided granules are fine granules.

2. **Number of Fine Granules**: The number of granules containing the object belonging to one class indicates the quality of outcomes or the fineness of granules. Therefore higher the numbers of granules containing objects of same class better the outcome of granulation.

3. **Number of Granules having more than one type of object**: It may not be possible to obtain fine granules as result of a granulation process on a dataset. The “No. of Granules having more than one type of object” is also proposed to be a parameter to measure the performance of granulation.

4. **Number of object which causes the result 3**: Granulation may not yield only fine granules. Few objects of other type/class may be clumped in a granule. Such objects may be clumped by the granulation process or may be outliers to the clump.

Result has also shows the percentage of fine granule of total granules. The tree representation of granulation result for Iris data has been shown in the Figure 4.3.
### Table 4.1: Granulation Result

<table>
<thead>
<tr>
<th>Result</th>
<th>Datasets</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Iris</td>
</tr>
<tr>
<td>1. No. of Granules</td>
<td>23</td>
</tr>
<tr>
<td>2. No. of Fine Granules</td>
<td>18</td>
</tr>
<tr>
<td>3. No. of Granules having more than one type of object</td>
<td>5</td>
</tr>
<tr>
<td>4. No. of object which cause the result 3</td>
<td>5</td>
</tr>
</tbody>
</table>

The total number of granules for all dataset is much lesser than number of unique tuples i.e. maximum number of granules. The percentage of fine granule is more than 98% for the Ion, Hea dataset. The ratio of fine granule to total granule is 18/23, 88/89, 151/153 and 145/209 for Iris, Ion, Hea and Pid respectively. This ratio is worst for Pid. It is noteworthy that Pid data does not have normal distribution. Though the number attributes is not large but all the 8 attributes are continuous. However Ion has largest number of attribute – 34 out of which 32 are continuous and they have been descretized by the method proposed in Chapter 3. It is observed that for this dataset the percentage of fine granules is maximum.

Traversing the granulation tree it is observed that fine granules are obtained at various depth of the tree. From the granulation tree of Iris dataset having 3 classes, only 2 out of 9 granules have all the three types of object at depth 2. It can also be

---

*Granulation using Clustering and Rough Set Theory & its Tree Representation* 85
observed that coarse granule at all depths contains a small percentage of objects from a class other than class labels of the objects in majority.

![Granulation Tree](image)

**Figure 4.3:** Tree representation of Granulation of Iris Data

Two out of four dataset experimented namely Ion and Hea gives good results with more than 98% fine results. Larger the number of fine granule better the granulation process. The fine granules are obtained at all depths in the tree. From the experiment on the 4 dataset a good number of fine granules are obtained at earlier stages. The coarseness of granules appears to be caused by very few objects in the chunk. Therefore the size (height) of the granulation tree can be controlled by allowing some percentage of impurity that is percentage of objects from other class. This may also depend on the strictness of the definition of the fine granule.
The proposed granulation scheme is based mainly on the discretization process. For $N$, the number of objects it was observed that the complexity of the proposed scheme was bounded by the complexity of the discretization process which i.e. $O(N^2)$.

4.6 Conclusion

The technique proposed in this chapter for granular computing is semi-supervised using clustering. The controls defined using rough set theory concept $\text{POS}_{a}(D)$ as implemented in Chapter 3 was also employed for granulation to deal with continuous attributes, during the granule formation. In the proposed method the natural intervals of attribute values (local) are obtained which maximize the mutual class-attribute interdependence (a global phenomenon) to generate a possibly minimum number of granules. A high percentage of good granules indicate the goodness of the scheme. The parameters for evaluation measuring granules with objects from different classes may help rank the granules with respect to accuracy. This may enable determine thresholds to define the goodness of a granule.

Complexity of the proposed granulation scheme beings $O(N^2)$. The attribute selection in the algorithm is random. Some heuristic such as “fineness” definition of granules may improve the granulation process by resulting in fine granule of variable size, and reducing the height of the tree, thereby reducing the total numbers of granule.