A Partition Model of Granular Computing

Y.Y. Yao

Department of Computer Science, University of Regina
Regina, Saskatchewan, Canada S4S 0A2
yyao@cs.uregina.ca; http://www.cs.uregina.ca/∼yyao

Abstract. There are two objectives of this chapter. One objective is to examine the basic principles and issues of granular computing. We focus on the tasks of granulation and computing with granules. From semantic and algorithmic perspectives, we study the construction, interpretation, and representation of granules, as well as principles and operations of computing and reasoning with granules. The other objective is to study a partition model of granular computing in a set-theoretic setting. The model is based on the assumption that a finite set of universe is granulated through a family of pairwise disjoint subsets. A hierarchy of granulations is modeled by the notion of the partition lattice. The model is developed by combining, reformulating, and reinterpreting notions and results from several related fields, including theories of granularity, abstraction and generalization (artificial intelligence), partition models of databases, coarsening and refining operations (evidential theory), set approximations (rough set theory), and the quotient space theory for problem solving.

1 Introduction

The basic ideas of granular computing, i.e., problem solving with different granularities, have been explored in many fields, such as artificial intelligence, interval analysis, quantization, rough set theory, Dempster-Shafer theory of belief functions, divide and conquer, cluster analysis, machine learning, databases, and many others [73]. There is a renewed and fast growing interest in granular computing [21, 30, 32, 33, 41, 43, 48, 50, 51, 58, 60, 70, 77].

The term "granular computing (GrC)" was first suggested by T.Y. Lin [74]. Although it may be difficult to have a precise and uncontroversial definition, we can describe granular computing from several perspectives.

We may define granular computing by examining its major components and topics. Granular computing is a label of theories, methodologies, techniques, and tools that make use of granules, i.e., groups, classes, or clusters of a universe, in the process of problem solving [60]. That is, granular computing is used as an umbrella term to cover these topics that have been studied in various fields in isolation. By examining existing studies in a unified framework of granular computing and extracting their commonalities, one may be able to develop a general theory for problem solving. Alternatively, we may define granular computing by

identifying its unique way of problem solving. Granular computing is a way of thinking that relies on our ability to perceive the real world under various grain sizes, to abstract and consider only those things that serve our present interest, and to switch among different granularities. By focusing on different levels of granularities, one can obtain various levels of knowledge, as well as inherent knowledge structure. Granular computing is essential to human problem solving, and hence has a very significant impact on the design and implementation of intelligent systems.

The ideas of granular computing have been investigated in artificial intelligence through the notions of granularity and abstraction. Hobbs proposed a theory of granularity based on the observation that "[w]e look at the world under various grain seizes and abstract from it only those things that serve our present interests" [18]. Furthermore, "[o]ur ability to conceptualize the world at different granularities and to switch among these granularities is fundamental to our intelligence and flexibility. It enables us to map the complexities of the world around us into simpler theories that are computationally tractable to reason in" [18]. Giunchigalia and Walsh proposed a theory of abstraction [14]. Abstraction can be thought of as "the process which allows people to consider what is *relevant* and to forget a lot of *irrelevant* details which would get in the way of what they are trying to do". They showed that the theory of abstraction captures and generalizes most previous work in the area.

The notions of granularity and abstraction are used in many subfields of artificial intelligence. The granulation of time and space leads naturally to temporal and spatial granularities. They play an important role in temporal and spatial reasoning [3, 4, 12, 19, 54]. Based on granularity and abstraction, many authors studied some fundamental topics of artificial intelligence, such as, for example, knowledge representation [14, 75], theorem proving [14], search [75, 76], planning [24], natural language understanding [35], intelligent tutoring systems [36], machine learning [44], and data mining [16].

Granular computing recently received much attention from computational intelligence community. The topic of fuzzy information granulation was first proposed and discussed by Zadeh in 1979 and further developed in the paper published in 1997 [71, 73]. In particular, Zadeh proposed a general framework of granular computing based on fuzzy set theory [73]. Granules are constructed and defined based on the concept of generalized constraints. Relationships between granules are represented in terms of fuzzy graphs or fuzzy if-then rules. The associated computation method is known as computing with words (CW) [72]. Although the formulation is different from the studies in artificial intelligence, the motivations and basic ideas are the same. Zadeh identified three basic concepts that underlie human cognition, namely, granulation, organization, and causation [73]. "Granulation involves decomposition of whole into parts, organization involves integration of parts into whole, and causation involves association of causes and effects." [73] Yager and Filev argued that "human beings have been developed a granular view of the world" and "...objects with which mankind perceives, measures, conceptualizes and reasons are granular" [58]. Therefore, as

pointed out by Zadeh, "[t]he theory of fuzzy information granulation (TFIG) is inspired by the ways in which humans granulate information and reason with it."[73]

The necessity of information granulation and simplicity derived from information granulation in problem solving are perhaps some of the practical reasons for the popularity of granular computing. In many situations, when a problem involves incomplete, uncertain, or vague information, it may be difficult to differentiate distinct elements and one is forced to consider granules [38–40]. In some situations, although detailed information may be available, it may be sufficient to use granules in order to have an efficient and practical solution. In fact, very precise solutions may not be required at all for many practical problems. It may also happen that the acquisition of precise information is too costly, and coarse-grained information reduces cost [73]. They suggest a basic guiding principle of fuzzy logic: "*Exploit the tolerance for imprecision, uncertainty and partial truth to achieve tractability, robustness, low solution cost and better rapport with reality*" [73]. This principle offers a more practical philosophy for real world problem solving. Instead of searching for the optimal solution, one may search for good approximate solutions. One only needs to examine the problem at a finer granulation level with more detailed information when there is a need or benefit for doing so [60].

The popularity of granular computing is also due to the theory of rough sets [38, 39]. As a concrete theory of granular computing, rough set model enables us to precisely define and analyze many notions of granular computing. The results provide an in-depth understanding of granular computing.

The objectives of this chapter are two-fold based on investigations at two levels. Sections 2 and 3 focus on a high and abstract level development of granular computing, and Section 3 deals with a low and concrete level development by concentrating on a partition model of granular computing. The main results are summarized as follows.

In Section 2, we discuss in general terms the basic principles and issues of granular computing based on related studies, such as the theory of granularity, the theory of abstraction, and their applications. The tasks of granulation and computing with granules are examined and related to existing studies. We study the construction, interpretation, and representation of granules, as well as principles and operations of computing and reasoning with granules. In Section 3, we argue that granular computing is a way of thinking. This way of thinking is demonstrated based on three problem solving domains, i.e., concept formation, top-down programming, and top-down theorem proving.

In Section 4, we study a partition model of granular computing in a set-theoretic setting. The model is based on the assumption that a finite set of universe is granulated through a family of pairwise disjoint subsets. A hierarchy of granulations is modeled by the notion of the partition lattice. Results from rough sets [38], quotient space theory [75, 76], belief functions [46], databases[27], data mining [31, 34], and power algebra [6] are reformulated, re-interpreted, refined, extended and combined for granular computing. We introduce two basic

operations called zooming-in and zooming-out operators. Zooming-in allows us to expand an element of the quotient universe into a subset of the universe, and hence reveals more detailed information. Zooming-out allows us to move to the quotient universe by ignoring some details. Computations in both universes can be connected through zooming operations.

2 Basic Issues of Granular Computing

Granular computing may be studied based on two related issues, namely granulation and computation [60]. The former deals with the construction, interpretation, and representation of granules, and the latter deals with the computing and reasoning with granules. They can be further divided with respect to algorithmic and semantic aspects [60]. The algorithmic study concerns the procedures for constructing granules and related computation, and the semantic study concerns the interpretation and physical meaningfulness of various algorithms. Studies from both aspects are necessary and important. The results from semantic study may provide not only interpretations and justifications for a particular granular computing model, but also guidelines that prevent possible misuses of the model. The results from algorithmic study may lead to efficient and effective granular computing methods and tools.

2.1 Granulations

Granulation of a universe involves dividing the universe into subsets or grouping individual objects into clusters. A granule may be viewed as a subset of the universe, which may be either fuzzy or crisp. A family of granules containing every object in the universe is called a granulation of the universe, which provides a coarse-grained view of the universe. A granulation may consist of a family of either disjoint or overlapping granules. There are many granulations of the same universe. Different views of the universe can be linked together, and a hierarchy of granulations can be established.

The notion of granulation can be studied in many different contexts. The granulation of the universe, particularly the semantics of granulation, is domain and application dependent. Nevertheless, one can still identify some domain independent issues [75]. Some of such issues are described in more detail below.

Granulation criteria. A granulation criterion deals with the semantic interpretation of granules and addresses the question of *why* two objects are put into the same granule. It is domain specific and relies on the available knowledge. In many situations, objects are usually grouped together based on their relationships, such as indistinguishability, similarity, proximity, or functionality [73]. One needs to build models to provide both semantical and operational interpretations of these notions. A model enables us to define formally and precisely various notions involved, and to study systematically the meanings and rationalities of granulation criteria.

Granulation structures. It is necessary to study granulation structures derivable from various granulations of the universe. Two structures can be observed, the structure of individual granules and structure of a granulation. Consider the case of crisp granulation. One can immediately define an order relation between granules based on the set inclusion. In general, a large granule may contain small granules, and small granules may be combined to form a large granule. The order relation can be extended to different granulations. This leads to multi-level granulations in a natural hierarchical structure. Various hierarchical granulation structures have been studied by many authors [22, 36, 54, 75, 76].

Granulation methods. From the algorithmic aspect, a granulation method addresses the problem of *how* to put two objects into the same granule. It is necessary to develop algorithms for constructing granules and granulations efficiently based on a granulation criterion. The construction process can be modeled as either top-down or bottom-up. In a top-down process, the universe is decomposed into a family of subsets, each subset can be further decomposed into smaller subsets. In a bottom-up process, a subset of objects can be grouped into a granule, and smaller granules can be further grouped into larger granules. Both processes lead naturally to a hierarchical organization of granules and granulations [22, 61].

Representation/description of granules. Another semantics related issue is the interpretation of the results of a granulation method. Once constructed, it is necessary to describe, to name and to label granules using certain languages. This can be done in several ways. One may assign a name to a granule such that an element in the granule is an instance of the named category, as being done in classification [22]. One may also construct a certain type of center as the representative of a granule, as being done in information retrieval [45, 56]. Alternatively, one may select some typical objects from a granule as its representative. For example, in many search engines, the search results are clustered into granules and a few titles and some terms can be used as the description of a granule [8, 17].

Quantitative characteristics of granules and granulations. One can associate quantitative measures to granules and granulations to capture their features. Consider again the case of crisp granulation. The cardinality of a granule, or Hartley information measure, can be used as a measure of the size or uncertainty of a granule [64]. The Shannon entropy measure can be used as a measure of the granularity of a partition [64].

These issues can be understood by examining a concrete example of granulation known as the cluster analysis [2]. This can be done by simply change granulation into clustering and granules into clusters. *Clustering structures* may be hierarchical or non-hierarchical, exclusive or overlapping. Typically, a similarity or distance function is used to define the relationships between objects. *Clustering criteria* may be defined based on the similarity or distance function, and the required cluster structures. For example, one would expect strong similarities between objects in the same cluster, and weak similarities between objects

in different clusters. Many *clustering methods* have been proposed and studied, including the families of hierarchical agglomerative, hierarchical divisive, iterative partitioning, density search, factor analytic, clumping, and graph theoretic methods [1]. Cluster analysis can be used as an exploratory tool to interpret data and find regularities from data [2]. This requires the active participation of experts to interpret the results of clustering methods and judge their significance. A good *representation* of clusters and their *quantitative characterizations* may make the task of exploration much easier.

2.2 Computing and reasoning with granules

Computing and reasoning with granules depend on the previously discussed notion of granulations. They can be similarly studied from both the semantic and algorithmic perspectives. One needs to design and interpret various methods based on the interpretation of granules and relationships between granules, as well as to define and interpret operations of granular computing.

The two level structures, the granule level and the granulation level, provide the inherent relationships that can be explored in problem solving. The granulated view summarizes available information and knowledge about the universe. As a basic task of granular computing, one can examine and explore further relationships between granules at a lower level, and relationships between granulations at a higher level. The relationships include closeness, dependency, and association of granules and granulations [43]. Such relationships may not hold fully and certain measures can be employed to quantify the degree to which the relationships hold [64]. This allows the possibility to extract, analyze and organize information and knowledge through relationships between granules and between granulations [62, 63].

The problem of computing and reasoning with granules is domain and application dependent. Some general domain independent principles and issues are listed below.

Mappings between different level of granulations. In the granulation hierarchy, the connections between different levels of granulations can be described by mappings. Giunchglia and Walsh view an abstraction as a mapping between a pair of formal systems in the development of a theory of abstraction [14]. One system is referred to as the ground space, and the other system is referred to as the abstract space. At each level of granulation, a problem is represented with respect to the granularity of the level. The mapping links different representations of the same problem at different levels of details. In general, one can classify and study different types of granulations by focusing on the properties of the mappings [14].

Granularity conversion. A basic task of granular computing is to change views with respect to different levels of granularity. As we move from one level of details to another, we need to convert the representation of a problem accordingly [12, 14]. A move to a more detailed view may reveal information that otherwise cannot be seen, and a move to a simpler view can improve the

high level understanding by omitting irrelevant details of the problem [12, 14, 18, 19, 73, 75, 76]. The change between grain-sized views may be metaphorically stated as the change between the forest and trees.

Property preservation. Granulation allows the different representations of the same problem in different levels of details. It is naturally expected that the same problem must be consistently represented [12]. A granulation and its related computing methods are meaningful only they preserve certain desired properties [14, 30, 75]. For example, Zhang and Zhang studied the "false-preserving" property, which states that if a coarse-grained space has no solution for a problem then the original fine-grained space has no solution [75, 76]. Such a property can be explored to improve the efficiency of problem solving by eliminating a more detailed study in a coarse-grained space. One may require that the structure of a solution in a coarse-grained space is similar to the solution in a fine-grained space. Such a property is used in top-down problem solving techniques. More specifically, one starts with a sketched solution and successively refines it into a full solution. In the context of hierarchical planning, one may impose similar properties, such as upward solution property, downward solution property, monotonicity property, etc. [24].

Operators. The relationship between granules at different levels and conversion of granularity can be precisely defined by operators [12, 36]. They serve as the basic build blocks of granular computing. There are at least two types of operators that can be defined. One type deals with the shift from a fine granularity to a coarse granularity. A characteristics of such an operator is that it will discard certain details, which makes distinct objects no longer differentiable. Depending on the context, many interpretations and definitions are available, such as abstraction, simplification, generalization, coarsening, zooming-out, etc. [14, 18, 19, 36, 46, 66, 75]. The other type deals with the change from a coarse granularity to a fine granularity. A characteristics of such an operator is that it will provide more details, so that a group of objects can be further classified. They can be defined and interpreted differently, such as articulation, specification, expanding, refining, zooming-in, etc. [14, 18, 19, 36, 46, 66, 75]. Other types of operators may also be defined. For example, with the granulation, one may not be able to exactly characterize an arbitrary subset of a fine-grained universe in a coarse-grained universe. This leads to the introduction of approximation operators in rough set theory [39, 59].

The notion of granulation describes our ability to perceive the real world under various grain sizes, and to abstract and consider only those things that serve our present interest. Granular computing methods describe our ability to switch among different granularities in problem solving. Detailed and domain specific methods can be developed by elaborating these issues with explicit reference to an application. For example, concrete domain specific conversion methods and operators can be defined. In spite of the differences between various methods, they are all governed by the same underlying principles of granular computing.

3 Granular Computing as a Way of Thinking

The underlying ideas of granular computing have been used either explicitly or implicitly for solving a wide diversity of problems. Their effectiveness and merits may be difficult to study and analyze based on some kind of formal proofs. They may be judged based on the powerful and yet imprecise and subjective tools of our experience, intuition, reflections and observations [28]. As pointed out by Leron [28], a good way of activating these tools is to carry out some case studies. For such a purpose, the general ideas, principles, and methodologies of granular computing are further examined with respect to several different fields in the rest of this section. It should be noted that analytical and experimental results on the effectiveness of granular computing in specific domains, though will not be discussed in this chapter, are available [20, 24, 75].

3.1 Concept formation

From philosophical point of view, granular computing can be understood as a way of thinking in terms of the notion of concepts that underlie the human knowledge.

Every concept is understood as a unit of thoughts consisting of two parts, the intension and the extension of the concept [9, 52, 53, 55, 57]. The intension (comprehension) of a concept consists of all properties or attributes that are valid for all those objects to which the concept applies. The extension of a concept is the set of objects or entities which are instances of the concept. All objects in the extension have the same properties that characterize the concept. In other words, the intension of a concept is an abstract description of common features or properties shared by elements in the extension, and the extension consists of concrete examples of the concept. A concept is thus described jointly by its intension and extension. This formulation enables us to study concepts in a logic setting in terms of intensions and also in a set-theoretic setting in terms of extensions. The description of granules characterize concepts from the intension point of view, while granules themselves characterize concepts from the extension point of view. Through the connections between extensions of concepts, one may establish relationships between concepts [62, 63].

In characterizing human knowledge, one needs to consider two topics, namely, context and hierarchy [42, 47]. Knowledge is contextual and hierarchical. A context in which concepts are formed provides meaningful interpretation of the concepts. Knowledge is organized in a tower or a partial ordering. The base-level, or first-level, concepts are the most fundamental concepts, and higher-level concepts depend on lower-level concepts. To some extent, granulation and inherent hierarchical granulation structures reflect naturally the way in which human knowledge is organized. The construction, interpretation, and description of granules and granulations are of fundamental importance in the understanding, representation, organization and synthesis of data, information, and knowledge.

3.2 Top-down programming

The top-down programming is an effective technique to deal with the complex problem of programming, which is based on the notions of structured programming and stepwise refinement [26]. The principles and characteristics of the top-down design and stepwise refinement, as discussed by Ledgard, Gueras and Nagin [26], provide a convincing demonstration that granular computing is a way of thinking.

According to Ledgard, Gueras and Nagin [26], the top-down programming approach has the following characteristics:

Design in levels. A level consists of a set of modules. At higher levels, only a brief description of a module is provided. The details of the module are to be refined, divided into smaller modules, and developed in lower levels.

Initial language independence. The high-level representations at initial levels focus on expressions that are relevant to the problem solution, without explicit reference to machine and language dependent features.

Postponement of details to lower levels. The initial levels concern critical broad issues and the structure of the problem solution. The details such as the choice of specific algorithms and data structures are postponed to lower, implementation levels.

Formalization of each level. Before proceeding to a lower level, one needs to obtain a formal and precise description of the current level. This will ensure a full understanding of the structure of the current sketched solution.

Verification of each level. The sketched solution at each level must be verified, so that errors pertinent to the current level will be detected.

Successive refinements. Top-down programming is a successive refinement process. Starting from the top level, each level is redefined, formalized, and verified until one obtains a full program.

In terms of granular computing, program modules correspond to granules, and levels of the top-down programming correspond to different granularities. One can immediately see that those characteristics also hold for granular computing in general.

3.3 Top-down theorem proving

Another demonstration of granular computing as a way of thinking is the approach of top-down theorem proving, which is used by computer systems and human experts. The PROLOG interpreter basically employs a top-down, depth-first search strategy to solve problem through theorem proving [5]. It has also been suggested that the top-down approach is effective for developing, communicating and writing mathematical proofs [13, 14, 25, 28].

PROLOG is a logic programming language widely used in artificial intelligence. It is based on the first-order predicate logic and models problem solving as theorem proving [5]. A PROLOG program consists of a set of facts and rules

in the form of Horn clauses that describe the objects and relations in a problem domain. The PROLOG interpreter answers a query, referred to as goals, by finding out whether the query is a logical consequence of the facts and rules of a PROLOG program. The inference is performed in a top-down, left to right, depth-first manner. A query is a sequence of one or more goals. At the top level, the leftmost goal is reduced to a sequence of subgoals to be tried by using a clause whose head unifies with the leftmost goal. The PROLOG interpreter then continues by trying to reduce the leftmost goal of the new sequence of goals. Eventually the lestmost goal is satisfied by a fact, and the second leftmost goal is tried in the same manner. Backtracking is used when the interpreter fails to find a unification that solves a goal, so that other clauses can be tried.

A proof found by the PROLOG interpreter can be expressed naturally in a hierarchical structure, with the proofs of subgoals as the children of a goal. In the process of reducing a goal to a sequence of subgoals, one obtains more details of the proof. The strategy can be applied to general theorem proving. This may be carried out by abstracting the goal, proving its abstracted version and then using the structure of the resulting proof to help construct the proof of the original goal [14].

By observing the systematic way of top-down programming, some authors suggest that the similar approach can be used in developing, teaching and communicating mathematical proofs [13, 28]. Leron proposed a structured method for presenting mathematical proofs [28]. The main objective is to increase the comprehensibility of mathematical presentations, and at the same time, retain their rigor. The traditional linear fashion presents a proof step-by-step from hypotheses to conclusion. In contrast, the structured method arranges the proof in levels and proceeds in a top-down manner. Like the top-down, step-wise refinement programming approach, a level consists of short autonomous modules, each embodying one major idea of the proof to be further developed in the subsequent levels. The top level is a very general description of the main line of the proof. The second level elaborates on the generalities of the top level by supplying proofs of unsubstantiated statements, details of general descriptions, and so on. For some more complicated tasks, the second level only gives brief descriptions and the details are postponed to the lower levels. The process continues by supplying more details of the higher levels until a complete proof is reached. Such a development of proofs procedure is similar to the strategy used by the PROLOG interpreter. A complicated proof task is successively divided into smaller and easier subtasks. The inherent structures of those tasks not only improve the comprehensibility of the proof, but also increase our understanding of the problem.

Lamport proposed a proof style, a refinement of natural deduction, for developing and writing structured proofs [25]. It is also based on hierarchical structuring, and divides proofs into levels. By using a numbering scheme to label various parts of a proof, one can explicitly show the structures of the proof. Furthermore, such a structure can be conveniently expressed using a computer-based hypertext system. One can concentrate on a particular level in the structure

and suppress lower level details. In principle, the top-down design and step-wise refinement strategy of programming can be applied in developing proofs to eliminate possible errors.

3.4 Granular computing approach of problem solving

In their book on research methods, Granziano and Raulin make a clear separation of research process and content [11]. They state, "... the basic processes and the systematic way of studying problems are common elements of science, regardless of each discipline's particular subject matter. It is the process and not the content that distinguishes science from other ways of knowing, and it is the content – the particular phenomena and fact of interest – that distinguishes one scientific discipline from another." [11] From the discussion of the previous examples, we can make a similar separation of the granular computing process and content (i.e., domains of applications). The systematic way of granular computing is generally applicable to different domains, and can be studied based on the basic issues and principles discussed in the last section.

In general, granular computing approach can be divided into top-down and bottom-up modes. They present two directions of switch between levels of granularities. The concept formation can be viewed as a combination of top-down and bottom-up. One can combine specific concepts to produce a general concept in a bottom-up manner, and divide a concept into more specific subconcepts in top-down manner. Top-down programming and top-down theorem proving are typical examples of top-down approaches. Independent of the modes, step-wise (successive) refinement plays an important role. One needs to fully understand all notions of a particular level before moving up or down to another level.

From the case studies, we can abstract some common features by ignoring irrelevant formulation details. It is easy to arrive at a conclusion that granular computing is a way of thinking and a philosophy for problem solving. At an abstract level, it captures and reflects our ability to solve a problem by focusing on different levels of details, and move easily from different levels at various stages. The principles of granular computing are the same and applicable to many domains.

4 A Partition Model

A partition model is developed by focusing on the basic issues of granular computing. The partition model has been studied extensively in rough set theory [39].

4.1 Granulation by partition and partition lattice

A simple granulation of the universe can be defined based on an equivalence relation or a partition. Let U denote a finite and non-empty set called the universe. Suppose $E \subseteq U \times U$ denote an equivalence relation on U, where $\times$ denotes the Cartesian product of sets. That is, E is reflective, symmetric, and transitive.

The equivalence relation E divides the set U into a family of disjoint subsets called the partition of the universe induced by E and denoted by $\pi_E = U/E$. The subsets in a partition are also called blocks. Conversely, given a partition π of the universe, one can uniquely define an equivalence relation E_π:

$$xE_\pi y \Longleftrightarrow x \text{ and } y \text{ are in the same block of the partition } \pi. \tag{1}$$

Due to the one to one relationship between equivalence relations and partitions, one may use them interchangeably.

One can define an order relation on the set of all partitions of U, or equivalently the set of all equivalence relations on U. A partition π_1 is a refinement of another partition π_2, or equivalently, π_2 is a coarsening of π_1, denoted by $\pi_1 \preceq \pi_2$, if every block of π_1 is contained in some block of π_2. In terms of equivalence relations, we have $E_{\pi_1} \subseteq E_{\pi_2}$. The refinement relation $\preceq$ is a partial order, namely, it is reflexive, antisymmetric, and transitive. It defines a partition lattice $\Pi(U)$. Given two partitions π_1 and π_2, their meet, $\pi_1 \wedge \pi_2$, is the largest partition that is a refinement of both π_1 and π_2, their join, $\pi_1 \vee \pi_2$, is the smallest partition that is a coarsening of both π_1 and π_2. The blocks of the meet are all nonempty intersections of a block from π_1 and a block from π_2. The blocks of the join are the smallest subsets which are exactly a union of blocks from π_1 and π_2. In terms of equivalence relations, for two equivalence relations R_1 and R_2, their meet is defined by $R_1 \cap R_2$, and their join is defined by $(R_1 \cup R_2)^*$, the transitive closure of the relation $R_1 \cup R_2$.

The lattice $\Pi(U)$ contains all possible partition based granulations of the universe. The refinement partial order on partitions provides a natural hierarchy of granulations. The partition model of granular computing is based on the partition lattice or subsystems of the partition lattice.

4.2 Partition lattice in an information table

Information tables provide a simple and convenient way to represent a set of objects by a finite set of attributes [39, 70]. Formally, an information table is defined as the following tuple:

$$(U, At, \{V_a \mid a \in At\}, \{I_a \mid a \in At\}), \tag{2}$$

where U is a finite set of objects called the universe, At is a finite set of attributes or features, V_a is a set of values for each attribute $a \in At$, and $I_a : U \longrightarrow V_a$ is an information function for each attribute $a \in At$. A database is an example of information tables. Information tables give a specific and concrete interpretation of equivalence relations used in granulation.

With respect to an attribute $a \in At$, an object $x \in U$ takes only one value from the domain V_a of a. Let $a(x) = I_a(x)$ denote the value of x on a. By extending to a subset of attributes $A \subseteq At$, $A(x)$ denotes the value of x on attributes A, which can be viewed as a vector with each $a(x)$, $a \in A$, as one of its components. For an attribute $a \in At$, an equivalence relation E_a is given by: for $x, y \in U$,

$$xE_a y \Longleftrightarrow a(x) = a(y). \tag{3}$$

Two objects are considered to be indiscernible, in the view of a single attribute a, if and only if they have exactly the same value. For a subset of attributes $A \subseteq At$, an equivalence relation E_A is defined by:

$$\begin{aligned} xE_Ay &\Longleftrightarrow A(x) = A(y) \\ &\Longleftrightarrow (\forall a \in A)a(x) = a(y) \\ &\Longleftrightarrow \bigcap_{a \in A} E_a. \end{aligned} \tag{4}$$

With respect to all attributes in A, x and y are indiscernible, if and only if they have the same value for every attribute in A.

The empty set $\emptyset$ produces the coarsest relation, i.e., $E_\emptyset = U \times U$. If the entire attribute set is used, one obtains the finest relation E_{At}. Moreover, if no two objects have the same description, E_{At} becomes the identity relation. The algebra $(\{E_A\}_{A \subseteq At}, \cap)$ is a lower semilattice with the zero element E_{At} [37]. The family of partitions $\Pi(At(U)) = \{\pi_{E_A} \mid A \subseteq At\}$ has been studied in databases [27]. In fact, $\Pi(At(U))$ is a lattice on its own right. While the meet of $\Pi(At(U))$ is the same as the meet of $\Pi(U)$, their joins are different [27]. The lattice $\Pi(At(U))$ can be used to develop a partition model of databases.

A useful result from the constructive definition of the equivalence relation is that one can associate a precise description with each equivalence class. This is done through the introduction of a decision logic language DL in an information table [39, 43, 65]. In the language DL, an atomic formula is given by $a = v$, where $a \in At$ and $v \in V_a$. If ϕ and ψ are formulas, then so are $\neg\phi$, $\phi \wedge \psi$, $\phi \vee \psi$, $\phi \rightarrow \psi$, and $\phi \equiv \psi$. The semantics of the language DL can be defined in the Tarski's style through the notions of a model and satisfiability. The model is an information table, which provides interpretation for symbols and formulas of DL. The satisfiability of a formula ϕ by an object x, written $x \models \phi$, is given by the following conditions:

(1) $x \models a = v$ iff $a(x) = v$,
(2) $x \models \neg\phi$ iff not $x \models \phi$,
(3) $x \models \phi \wedge \psi$ iff $x \models \phi$ and $x \models \psi$,
(4) $x \models \phi \vee \psi$ iff $x \models \phi$ or $x \models \psi$,
(5) $x \models \phi \rightarrow \psi$ iff $x \models \neg\phi \vee \psi$,
(6) $x \models \phi \equiv \psi$ iff $x \models \phi \rightarrow \psi$ and $x \models \psi \rightarrow \phi$.

If ϕ is a formula, the set $m(\phi)$ defined by:

$$m(\phi) = \{x \in U \mid x \models \phi\}, \tag{5}$$

is called the meaning of the formula ϕ. For an equivalence class of E_A, it can be described by a formula of the form, $\bigwedge_{a \in A} a = v_a$, where $v_a \in V_a$. Furthermore, $[x]_{E_A} = m(\bigwedge_{a \in A} a = a(x))$, where $a(x)$ is the value of x on attribute a.

4.3 Mappings between two universes

Given an equivalence relation E on U, we obtain a coarse-grained universe U/E called the quotient set of U. The relation E can be conveniently represented by a mapping from U to 2^U, where 2^U is the power set of U. The mapping $[\cdot]_E : U \longrightarrow 2^U$ is given by:

$$[x]_E = \{y \in U \mid xEy\}. \tag{6}$$

The equivalence class $[x]_E$ containing x plays dual roles. It is a subset of U and an element of U/E. That is, in U, $[x]_E$ is subset of objects, and in U/E, $[x]_E$ is considered to be a whole instead of many individuals [61]. In cluster analysis, one typically associates a name with a cluster such that elements of the cluster are instances of the named category or concept [22]. Lin [29], following Dubois and Prade [10], explicitly used $[x]_E$ for representing a subset of U and $Name([x]_E)$ for representing an element of U/E. In subsequent discussion, we use this convention.

With a partition or an equivalence relation, we have two views of the same universe, a coarse-grained view U/E and a detailed view U. Their relationship can be defined by a pair of mappings, $r : U/E \longrightarrow U$ and $c : U \longrightarrow U/E$. More specifically, we have:

$$\begin{aligned} &r(Name([x]_E)) = [x]_E, \\ &c(x) = Name([x]_E). \end{aligned} \tag{7}$$

A concept, represented as a subset of a universe, is described differently under different views. As we move from one view to the other, we change our perceptions and representations of the same concept. In order to achieve this, we define zooming-in and zooming-out operators based on the pair of mappings [66].

4.4 Zooming-in operator for refinement

Formally, zooming-in can be defined by an operator $\omega : 2^{U/E} \longrightarrow 2^U$. Shafer referred to the zooming-in operation as refining [46]. For a singleton subset $\{X_i\} \in 2^{U/E}$, we define [10]:

$$\omega(\{X_i\}) = [x]_E, \quad X_i = Name([x]_E). \tag{8}$$

For an arbitrary subset $X \subseteq U/E$, we have:

$$\omega(X) = \bigcup_{X_i \in X} \omega(\{X_i\}). \tag{9}$$

By zooming-in on a subset $X \subseteq U/E$, we obtain a unique subset $\omega(X) \subseteq U$. The set $\omega(X) \subseteq U$ is called the refinement of X.

The zooming-in operation has the following properties [46]:

$$\begin{aligned}
&\text{(zi1)} \quad \omega(\emptyset) = \emptyset,\\
&\text{(zi2)} \quad \omega(U/E) = U,\\
&\text{(zi3)} \quad \omega(X^c) = (\omega(X))^c,\\
&\text{(zi4)} \quad \omega(X \cap Y) = \omega(X) \cap \omega(Y),\\
&\text{(zi5)} \quad \omega(X \cup Y) = \omega(X) \cup \omega(Y),\\
&\text{(zi6)} \quad X \subseteq Y \Longleftrightarrow \omega(X) \subseteq \omega(Y),
\end{aligned}$$

where ^{c} denotes the set complement operator, the set-theoretic operators on the left hand side apply to the elements of $2^{U/E}$, and the same operators on the right hand side apply to the elements of 2^U. From these properties, it can be seen that any relationships of subsets observed under coarse-grained view would hold under the detailed view, and vice versa. For example, in addition to (zi6), we have $X \cap Y = \emptyset$ if and only if $\omega(X) \cap \omega(Y) = \emptyset$, and $X \cup Y = U/E$ if and only if $\omega(X) \cup \omega(Y) = U$. Therefore, conclusions drawn based on the coarse-grained elements in U/E can be carried over to the universe U.

4.5 Zooming-out operators for approximation

The change of views from U to U/E is called a zooming-out operation. By zooming-out, a subset of the universe is considered as a whole rather than many individuals. This leads to a loss of information. Zooming-out on a subset $A \subseteq U$ may induce an inexact representation in the coarse-grained universe U/E.

The theory of rough sets focuses on the zooming-out operation. For a subset $A \subseteq U$, we have a pair of lower and upper approximations in the coarse-grained universe [7, 10, 59]:

$$\begin{aligned}
&\underline{apr}(A) = \{Name([x]_E) \mid x \in U, [x]_E \subseteq A\},\\
&\overline{apr}(A) = \{Name([x]_E) \mid x \in U, [x]_E \cap A \neq \emptyset\}.
\end{aligned} \tag{10}$$

The expression of lower and upper approximations as subsets of U/E, rather than subsets of U, has only been considered by a few researchers in rough set community [7, 10, 30, 59, 69]. On the other hand, such notions have been considered in other contexts. Shafer [46] introduced those notions in the study of belief functions and called them the inner and outer reductions of $A \subseteq U$ in U/E. The connections between notions introduced by Pawlak in rough set theory and these introduced by Shafer in belief function theory have been pointed out by Dubois and Prade [10].

The expression of approximations in terms of elements of U/E clearly shows that representation of A in the coarse-grained universe U/E. By zooming-out, we only obtain an approximate representation. The lower and upper approximations satisfy the following properties [46, 69]:

$$\text{(zo1)} \quad \underline{apr}(\emptyset) = \overline{apr}(\emptyset) = \emptyset,$$

(zo2) $\underline{apr}(U) = \overline{apr}(U) = U/E$,

(zo3) $\underline{apr}(A) = (\overline{apr}(A^c))^c$,
$\overline{apr}(A) = (\underline{apr}(A^c))^c$;

(zo4) $\underline{apr}(A \cap B) = \underline{apr}(A) \cap \underline{apr}(B)$,
$\overline{apr}(A \cap B) \subseteq \overline{apr}(A) \cap \overline{apr}(B)$,

(zo5) $\underline{apr}(A) \cup \underline{apr}(B) \subseteq \underline{apr}(A \cup B)$,
$\overline{apr}(A \cup B) = \overline{apr}(A) \cup \overline{apr}(B)$,

(zo6) $A \subseteq B \Longrightarrow [\underline{apr}(A) \subseteq \underline{apr}(B), \overline{apr}(A) \subseteq \overline{apr}(B)]$,

(zo7) $\underline{apr}(A) \subseteq \overline{apr}(A)$.

According to properties (zo4)-(zo6), relationships between subsets of U may not be carried over to U/E through the zooming-out operation. It may happen that $A \cap B \neq \emptyset$, but $\underline{apr}(A \cap B) = \emptyset$, or $A \cup B \neq U$, but $\overline{apr}(A \cup B) = U/E$. Similarly, we may have $A \neq B$, but $\underline{apr}(A) = \underline{apr}(B)$ and $\overline{apr}(A) = \overline{apr}(B)$. Nevertheless, we can draw the following inferences:

(i1) $\underline{apr}(A) \cap \underline{apr}(B) \neq \emptyset \Longrightarrow A \cap B \neq \emptyset$,

(i2) $\overline{apr}(A) \cap \overline{apr}(B) = \emptyset \Longrightarrow A \cap B = \emptyset$,

(i3) $\underline{apr}(A) \cup \underline{apr}(B) = U/E \Longrightarrow A \cup B = U$,

(i4) $\overline{apr}(A) \cup \overline{apr}(B) \neq U/E \Longrightarrow A \cup B \neq U$.

If $\underline{apr}(A) \cap \underline{apr}(B) \neq \emptyset$, by property (zo4) we know that $\underline{apr}(A \cap B) \neq \emptyset$. We say that A and B have a non-empty overlap, and hence are related, in U/E. By (i1), A and B must have a non-empty overlap, and hence are related, in U. Similar explanations can be associated with other inference rules.

The approximation of a set can be easily extended to the approximation of a partition, also called a classification [39]. Let $\pi = \{X_1, \ldots, X_n\}$ be a partition of the universe U. Its approximations are a pair of families of sets, the family of lower approximations $\underline{apr}(\pi) = \{\underline{apr}(X_1), \ldots, \underline{apr}(X_n)\}$ and the family of upper approximations $\overline{apr}(\pi) = \{\overline{apr}(X_1), \ldots, \overline{apr}(X_n)\}$.

4.6 Classical rough set approximations by a combination of zooming-out and zooming-in

Traditionally, lower and upper approximations of a set are also subsets of the same universe. One can easily obtain the classical definition by performing a combination of zooming-out and zooming-in operators as follows [66]:

$$\begin{aligned}
\omega(\underline{apr}(A)) &= \bigcup_{X_i \in \underline{apr}(A)} \omega(\{X_i\}) \\
&= \bigcup \{[x]_E \mid x \in U, [x]_E \subseteq A\}, \\
\omega(\overline{apr}(A)) &= \bigcup_{X_i \in \overline{apr}(A)} \omega(\{X_i\}) \\
&= \bigcup \{[x]_E \mid x \in U, [x]_E \cap A \neq \emptyset\}.
\end{aligned} \tag{11}$$

For a subset $X \subseteq U/E$ we can zoom-in and obtain a subset $\omega(X) \subseteq U$, and then zoom-out to obtain a pair of subsets $\underline{apr}(\omega(X))$ and $\overline{apr}(\omega(X))$. The compositions of zooming-in and zooming-out operations have the properties [46]: for $X \subseteq U/E$ and $A \subseteq U$,

$$\begin{aligned} &\text{(zio1)} \qquad \omega(\underline{apr}(A)) \subseteq A \subseteq \omega(\overline{apr}(A)), \\ &\text{(zio2)} \qquad \underline{apr}(\omega(X)) = \overline{apr}(\omega(X)) = X. \end{aligned}$$

The composition of zooming-out and zooming-in cannot recover the original set $A \subseteq U$. The composition zooming-in and zooming-out produces the original set $X \subseteq U/E$. A connection between the zooming-in and zooming-out operations can be established. For a pair of subsets $X \subseteq U/E$ and $A \subseteq U$, we have [46]:

$$\begin{aligned} &(1) \qquad w(X) \subseteq A \Longleftrightarrow X \subseteq \underline{apr}(A), \\ &(2) \qquad A \subseteq \omega(X) \Longleftrightarrow \overline{apr}(A) \subseteq X. \end{aligned}$$

Property (1) can be understood as follows. Any subset $X \subseteq U/E$, whose refinement is a subset of A, is a subset of the lower approximation of A. Only a subset of the lower approximation of A has a refinement that is a subset of A. It follows that $\underline{apr}(A)$ is the largest subset of U/E whose refinement is contained in A, and $\overline{apr}(A)$ is the smallest subset of U/E whose refinement containing A.

4.7 Consistent computations in the two universes

Computation in the original universe is normally based on elements of U. When zooming-out to the coarse-grained universe U/E, we need to find the consistent computational methods. The zooming-in operator can be used for achieving this purpose.

Suppose $f : U \longrightarrow \Re$ is a real-valued function on U. One can lift the function f to U/E by performing set-based computations [67]. The lifted function f^+ is a set-valued function that maps an element of U/E to a subset of real numbers. More specifically, for an element $X_i \in U/E$, the value of function is given by:

$$f^+(X_i) = \{f(x) \mid x \in \omega(\{X_i\})\}. \tag{12}$$

The function f^+ can be changed into a single-valued function f_0^+ in a number of ways. For example, Zhang and Zhang [75] suggested the following methods:

$$\begin{aligned} f_0^+(X_i) &= \min f^+(X_i) = \min\{f(x) \mid x \in \omega(\{X_i\})\}, \\ f_0^+(X_i) &= \max f^+(X_i) = \max\{f(x) \mid x \in \omega(\{X_i\})\}, \\ f_0^+(X_i) &= \text{average} f^+(X_i) = \text{average}\{f(x) \mid x \in \omega(\{X_i\})\}. \end{aligned} \tag{13}$$

The minimum, maximum, and average definitions may be regarded as the most permissive, the most optimistic, and the balanced view in moving functions from U to U/E. More methods can be found in the book by Zhang and Zhang [75].

For a binary operation $\circ$ on U, a binary operation $\circ^+$ on U/E is defined by [6, 67]:

$$X_i \circ^+ X_j = \{x_i \circ x_j \mid x_i \in \omega(\{X_i\}), x_j \in \omega(\{X_j\})\}, \tag{14}$$

In general, one may lift any operation p on U to an operation p^+ on U/E, called the power operation of p. Suppose $p : U^n \longrightarrow U$ $(n \geq 1)$ is an n-ary operation on U. Its power operation $p^+ : (U/E)^n \longrightarrow 2^U$ is defined by [6]:

$$p^+(X_0, \ldots, X_{n-1}) = \{p(x_0, \ldots, x_{n-1}) \mid x_i \in \omega(\{X_i\}) \text{ for } i = 0, \ldots, n-1\}, \quad (15)$$

for any $X_0, \ldots, X_{n-1} \in U/E$. This provides a universal-algebraic construction approach. For any algebra $(U, p_1, \ldots, p_k)$ with base set U and operations $p_1, \ldots, p_k$, its quotient algebra is given by $(U/E, p_1^+, \ldots, p_k^+)$.

The power operation p^+ may carry some properties of p. For example, for a binary operation $p : U^2 \longrightarrow U$, if p is commutative and associative, p^+ is commutative and associative, respectively. If e is an identity for some operation p, the set $\{e\}$ is an identity for p^+. Many properties of p are not carried over by p^+. For instance, if a binary operation p is idempotent, i.e., $p(x, x) = x$, p^+ may not be idempotent. If a binary operation g is distributive over p, g^+ may not be distributive over p^+.

In some situations, we need to carry information from the quotient set U/E to U. This can be done through the zooming-out operators. A simple example is used to illustrate the basic idea.

Suppose $\mu : 2^{U/E} \longrightarrow [0, 1]$ is a set function on U/E. If μ satisfies the conditions:

$$\begin{aligned} &\text{(i)} && \mu(\emptyset) = 0, \\ &\text{(ii)} && \mu(U/E) = 1, \\ &\text{(iii)} && X \subseteq Y \Longrightarrow \mu(X) \leq \mu(Y), \end{aligned}$$

μ is called a fuzzy measure [23]. Examples of fuzzy measures are probability functions, possibility and necessity functions, and belief and plausibility functions. Information about subsets in U can be obtained from μ on U/E and the zooming-out operation. For a subset $A \subseteq U$, we can define a pair of inner and outer fuzzy measures [68]:

$$\begin{aligned} \underline{\mu}(A) &= \mu(\underline{apr}(A)), \\ \overline{\mu}(A) &= \mu(\overline{apr}(A)). \end{aligned} \quad (16)$$

They are fuzzy measures. If μ is a probability function, $\underline{\mu}$ and $\overline{\mu}$ are a pair of belief and plausibility functions [15, 49, 46, 68]. If μ is a belief function, $\underline{\mu}$ is a belief function, and if μ is a plausibility function, $\overline{\mu}$ is a plausibility [68].

5 Conclusion

Granular computing, as a way of thinking, has been explored in many fields. It captures and reflects our ability to perceive the world at different granularity and to change granularities in problem solving. In this chapter, the same approach is used to study the granular computing itself in two levels. In the first part of the chapter, we consider the fundamental issues of granular computing in general

terms. The objective is to present a domain-independent way of thinking without details of any specific formulation. The second part of the chapter concretizes the high level investigations by considering a partition model of granular computing. To a large extent, the model is based on the theory of rough sets. However, results from other theories, such as the quotient space theory, belief functions, databases, and power algebras, are incorporated.

In the development of different research fields, each field may develop its theories and methodologies in isolation. However, one may find that these theories and methodologies share the same or similar underlying principles and only differ in their formulation. It is evident that granular computing may be a basic principle that guides many problem solving methods.

The results of rough set theory have drawn our attention to granular computing. On the other hand, the study of rough set theory in the wide context of granular computing may result in an in-depth understanding of rough set theory.

References

1. Aldenderfer, M.S., Blashfield, R.K.: Cluster Analysis. Sage Publications, The International Professional Publishers, London (1984)
2. Anderberg, M.R.: Cluster Analysis for Applications. Academic Press, New York (1973)
3. Bettini, C., Montanari, A. (Eds.): Spatial and Temporal Granularity: Papers from the AAAI Workshop. Technical Report WS-00-08. The AAAI Press, Menlo Park, CA. (2000)
4. Bettini, C., Montanari, A.: Research issues and trends in spatial and temporal granularities. Annals of Mathematics and Artificial Intelligence **36** (2002) 1-4
5. Bratko, I.: PROLOG: Programming for Artificial Intelligence, Second edition. Addison-Wesley, New York (1990)
6. Brink, C.: Power structures. Algebra Universalis **30** (1993) 177-216
7. Bryniarski, E.: A calculus of rough sets of the first order. Bulletin of the Polish Academy of Sciences, Mathematics **37** (1989) 71-77
8. de Loupy, C., Bellot, P., El-Bèze, M., Marteau, P.F.: Query expansion and classification of retrieved documents. Proceedings of the Seventh Text REtrieval Conference (TREC-7) (1998) 382-389
9. Demri, S, Orlowska, E.: Logical analysis of indiscernibility. In: Incomplete Information: Rough Set Analysis, Orlowska, E. (Ed.). Physica-Verlag, Heidelberg (1998) 347-380
10. Dubois, D., Prade, P.: Fuzzy rough sets and rough fuzzy sets. International Journal of General Systems **17** (1990) 191-209
11. Graziano, A.M., Raulin, M.L.: Research Methods: A Process of Inquiry, 4th edition. Allyn and Bacon, Boston (2000)
12. Euzenat, J.: Granularity in relational formalisms - with application to time and space representation. Computational Intelligence **17** (2001) 703-737
13. Friske, M.: Teaching proofs: a lesson from software engineering. American Mathematical Monthly **92** (1995) 142-144
14. Giunchglia, F., Walsh, T.: A theory of abstraction. Artificial Intelligence **56** (1992) 323-390

15. Grzymala-Busse, J.W.: Rough-set and Dempster-Shafer approaches to knowledge acquisition under uncertainty – a comparison. Manuscript. Department of Computer Science, University of Kansas (1987)
16. Han, J., Cai, Y., Cercone, N.: Data-driven discovery of quantitative rules in data bases. IEEE Transactions on Knowledge and Data Engineering **5** (1993) 29-40
17. Hearst, M.A., Pedersen, J.O.: Reexamining the cluster hypothesis: Scatter/Gather on retrieval results. Proceedings of SIGIR'96 (1996) 76-84
18. Hobbs, J.R.: Granularity. Proceedings of the Ninth Internation Joint Conference on Artificial Intelligence (1985) 432-435
19. Hornsby, K.: Temporal zooming. Transactions in GIS **5** (2001) 255-272
20. Imielinski, T.: Domain abstraction and limited reasoning. Proceedings of the 10th International Joint Conference on Artificial Intelligence (1987) 997-1003
21. Inuiguchi, M., Hirano, S., Tsumoto, S. (Eds.): Rough Set Theory and Granular Computing. Springer, Berlin (2003)
22. Jardine, N., Sibson, R.: Mathematical Taxonomy. Wiley, New York (1971)
23. Klir, G.J., Folger, T.A.: Fuzzy Sets, Uncertainty, and Information. Prentice Hall, Englewood Cliffs (1988)
24. Knoblock, C.A.: Generating Abstraction Hierarchies: an Automated Approach to Reducing Search in Planning. Kluwer Academic Publishers, Boston (1993)
25. Lamport, L.: How to write a proof. American Mathematical Monthly **102** (1995) 600-608
26. Ledgard, H.F., Gueras, J.F., Nagin, P.A.: PASCAL with Style: Programming Proverbs. Hayden Book Company, Inc., Rechelle Park, New Jersey (1979)
27. Lee, T.T.: An information-theoretic analysis of relational databases – part I: data dependencies and information metric. IEEE Transactions on Software Engineering **SE-13** (1987) 1049-1061
28. Leron, U.: Structuring mathematical proofs. American Mathematical Monthly **90** (1983) 174-185
29. Lin, T.Y.: Topological and fuzzy rough sets. In: Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory, Slowinski, R. (Ed.). Kluwer Academic Publishers, Boston (1992) 287-304
30. Lin, T.Y.: Granular computing on binary relations I: data mining and neighborhood systems, II: rough set representations and belief functions. In: Rough Sets in Knowledge Discovery 1. Polkowski, L., Skowron, A. (Eds.). Physica-Verlag, Heidelberg (1998) 107-140
31. Lin, T.Y.: Generating concept hierarchies/networks: mining additional semantics in relational data. Advances in Knowledge Discovery and Data Mining, Proceedings of the 5th Pacific-Asia Conference, Lecture Notes on Artificial Intelligence 2035 (2001) 174-185
32. Lin, T.Y.: Granular computing. Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Proceedings of the 9th International Conference, Lecture Notes in Artificial Intelligence 2639 (2003) 16-24.
33. Lin. T.Y., Yao, Y.Y., Zadeh, L.A. (Eds.): Rough Sets, Granular Computing and Data Mining. Physica-Verlag, Heidelberg (2002)
34. Lin, T.Y., Zhong, N., Dong, J., Ohsuga, S.: Frameworks for mining binary relations in data. Rough sets and Current Trends in Computing, Proceedings of the 1st International Conference, Lecture Notes in Artificial Intelligence 1424 (1998) 387-393
35. Mani, I.: A theory of granularity and its application to problems of polysemy and underspecification of meaning. Principles of Knowledge Representation and Reasoning, Proceedings of the Sixth International Conference (1998) 245-255

36. McCalla, G., Greer, J., Barrie, J., Pospisil, P.: Granularity hierarchies. Computers and Mathematics with Applications **23** (1992) 363-375
37. Orlowska, E.: Logic of indiscernibility relations. Bulletin of the Polish Academy of Sciences, Mathematics **33** (1985) 475-485
38. Pawlak, Z.: Rough sets. International Journal of Computer and Information Sciences **11** (1982) 341-356.
39. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Boston (1991)
40. Pawlak, Z.: Granularity of knowledge, indiscernibility and rough sets. Proceedings of 1998 IEEE International Conference on Fuzzy Systems (1998) 106-110
41. Pedrycz, W.: Granular Computing: An Emerging Paradigm. Springer-Verlag, Berlin (2001)
42. Peikoff, L.: Objectivism: the Philosophy of Ayn Rand. Dutton, New York (1991)
43. Polkowski, L., Skowron, A.: Towards adaptive calculus of granules. Proceedings of 1998 IEEE International Conference on Fuzzy Systems (1998) 111-116
44. Saitta, L., Zucker, J.-D.: Semantic abstraction for concept representation and learning. Proceedings of the Symposium on Abstraction, Reformulation and Approximation (1998) 103-120
http://www.cs.vassar.edu/∼ellman/sara98/papers/. retrieved on December 14, 2003.
45. Salton, G., McGill, M.: Introduction to Modern Information Retrieval. McGraw Hill, New York (1983)
46. Shafer, G.: A Mathematical Theory of Evidence. Princeton University Press, Princeton (1976)
47. Simpson, S.G.: What is foundations of mathematics? (1996).
http://www.math.psu.edu/simpson/hierarchy.html. retrieved November 21, 2003.
48. Skowron, A.: Toward intelligent systems: calculi of information granules. Bulletin of International Rough Set Society **5** (2001) 9-30
49. Skowron, A., Grzymala-Busse, J.: From rough set theory to evidence theory. In: Advances in the Dempster-Shafer Theory of Evidence, Yager, R.R., Fedrizzi, M., Kacprzyk, J. (Eds.). Wiley, New York (1994) 193-236
50. Skowron, A., Stepaniuk, J.: Information granules and approximation spaces. Proceedings of 7th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (1998) 354-361
51. Skowron, A., Stepaniuk, J.: Information granules: towards foundations of granular computing. International Journal of Intelligent Systems **16** (2001) 57-85
52. Smith, E.E.: Concepts and induction. In: Foundations of Cognitive Science, Posner, M.I. (Ed.). The MIT Press, Cambridge (1989) 501-526
53. Sowa, J.F.: Conceptual Structures, Information Processing in Mind and Machine. Addison-Wesley, Reading (1984)
54. Stell, J.G., Worboys, M.F.: Stratified map spaces: a formal basis for multi-resolution spatial databases. Proceedings of the 8th International Symposium on Spatial Data Handling (1998) 180-189
55. van Mechelen, I., Hampton, J., Michalski, R.S., Theuns, P. (Eds.): Categories and Concepts: Theoretical Views and Inductive Data Analysis. Academic Press, New York (1993)
56. van Rijsbergen, C.J.: Information Retrieval. Butterworths, London (1979)
57. Wille, R.: Concept lattices and conceptual knowledge systems. Computers Mathematics with Applications **23** (1992) 493-515

58. Yager, R.R., Filev,D.: Operations for granular computing: mixing words with numbers. Proceedings of 1998 IEEE International Conference on Fuzzy Systems (1998) 123-128
59. Yao, Y.Y.: Two views of the theory of rough sets in finite universes. International Journal of Approximation Reasoning **15** (1996) 291-317
60. Yao, Y.Y.: Granular computing: basic issues and possible solutions. Proceedings of the 5th Joint Conference on Information Sciences (2000) 186-189
61. Yao, Y.Y.: Information granulation and rough set approximation. International Journal of Intelligent Systems **16** (2001) 87-104
62. Yao, Y.Y.: Modeling data mining with granular computing. Proceedings of the 25th Annual International Computer Software and Applications Conference (COMPSAC 2001) (2001) 638-643
63. Yao, Y.Y.: A step towards the foundations of data mining. In: Data Mining and Knowledge Discovery: Theory, Tools, and Technology V, Dasarathy, B.V. (Ed.). The International Society for Optical Engineering (2003) 254-263
64. Yao, Y.Y.: Probabilistic approaches to rough sets. Expert Systems **20** (2003) 287-297
65. Yao, Y.Y., Liau, C.-J.: A generalized decision logic language for granular computing. Proceedings of FUZZ-IEEE'02 in the 2002 IEEE World Congress on Computational Intelligence, (2002) 1092-1097
66. Yao, Y.Y., Liau, C.-J., Zhong, N.: Granular computing based on rough sets, quotient space theory, and belief functions. Proceedings of ISMIS'03 (2003) 152-159
67. Yao, Y.Y., Noroozi, N.: A unified framework for set-based computations. Proceedings of the 3rd International Workshop on Rough Sets and Soft Computing. The Society for Computer Simulation (1995) 252-255
68. Yao, Y.Y., Wong, S.K.M.: Representation, propagation and combination of uncertain information. International Journal of General Systems **23** (1994) 59-83
69. Yao, Y.Y., Wong, S.K.M., Lin, T.Y.: A review of rough set models. In: Rough Sets and Data Mining: Analysis for Imprecise Data, Lin, T.Y., Cercone, N. (Eds.). Kluwer Academic Publishers, Boston (1997) 47-75
70. Yao, Y.Y., Zhong, N.: Granular computing using information tables. In: Data Mining, Rough Sets and Granular Computing, Lin, T.Y., Yao, Y.Y., Zadeh, L.A. (Eds.). Physica-Verlag, Heidelberg (2002) 102-124
71. Zadeh, L.A.: Fuzzy sets and information granularity. In: Advances in Fuzzy Set Theory and Applications, Gupta, N., Ragade, R., Yager, R. (Eds.). North-Holland, Amsterdam (1979) 3-18
72. Zadeh, L.A.: Fuzzy logic = computing with words. IEEE Transactions on Fuzzy Systems **4** (1996) 103-111
73. Zadeh, L.A.: Towards a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. Fuzzy Sets and Systems **19** (1997) 111-127
74. Zadeh, L.A.: Some reflections on soft computing, granular computing and their roles in the conception, design and utilization of information/intelligent systems. Soft Computing **2** (1998) 23-25
75. Zhang, B., Zhang, L.: Theory and Applications of Problem Solving, North-Holland, Amsterdam (1992)
76. Zhang, L., Zhang, B.: The quotient space theory of problem solving. Proceedings of International Conference on Rough Sets, Fuzzy Set, Data Mining and Granular Computing, Lecture Notes in Artrifical Intelligence 2639 (2003) 11-15
77. Zhong, N., Skowron, A., Ohsuga S. (Eds.): New Directions in Rough Sets, Data Mining, and Granular-Soft Computing. Springer-Verlag, Berlin (1999)