A Logic Approach to Granular Computing

Bing Zhou

Department of Computer Science, University of Regina, Canada

3737 Wascana Parkway, Regina, Saskatchewan, Canada S4S 0A2

Phone Number: 1-306-585-5695

Fax Number: 1-306-585-5695

E-mail: zhou200b@cs.uregina.ca

Yiyu Yao

Department of Computer Science, University of Regina, Canada

3737 Wascana Parkway, Regina, Saskatchewan, Canada S4S 0A2

Phone Number: 1-306-585-5226

Fax Number: 1-306-585-4745

E-mail: yyao@cs.uregina.ca

Abstract

Granular computing is an emerging field of study that attempts to formalize and explore methods and heuristics of human problem solving with multiple levels of granularity and abstraction. A fundamental issue of granular computing is the representation and utilization of granular structures. The main objective of this paper is to examine a logic approach to address this issue. Following the classical interpretation of a concept as a pair of intension and extension, we interpret a granule as a pair of a set of objects and a logic formula describing the granule. The building blocks of granular structures are basic granules representing an elementary concept or a piece of knowledge. They are treated as atomic formulas of a logic language. Different types of granular structures can be constructed by using logic connectives. Within this logic framework, we show that rough set analysis (RSA) and formal concept analysis (FCA) can be interpreted uniformly. The two theories use multilevel granular structures, but differ in their choices of definable granules and granular structures.

1. Introduction

Cognitive science (Simon & Kaplan, 1989) and cognitive informatics (Wang, 2003) study information and knowledge processing in the abstract, in the brain, and in machines. Some of the salient features of human intelligence and problem solving are the conceptualization of a problem at multiple levels of abstraction, the representation of information and knowledge with different-sized granules, the choice of a suitable level of granularity, and the switching of views and granularity in response to changes in environments. An emerging field of study known as granular computing aims at formalizing and exploring these features (Bargiela & Pedrycz, 2002; Yao, 2004; Yao, 2006; Yao, 2007; Zadeh, 1997). The results from granular computing may shed new light on the study of cognitive informatics (Wang, 2003; Wang, 2007; Yao, 2007).

A central notion of granular computing is multilevel granular structures consisting of a family of interrelated and interacting granules. Granular computing can be considered as an umbrella term covering topics that concern granularity and has been studied either implicitly or explicitly in many fields. It focuses on problem solving by describing and representing a problem and its solution in various levels of granularity so that one can focus on things that serve a specific interest and ignore unimportant and irrelevant details. Granular computing makes use of knowledge structures and hence has a significant impact on the study of human intelligence and the design of intelligent systems.

Granular computing can be studied based on a conceptual triarchic model consisting of the philosophy of structured thinking, the methodology of structured problem solving, and the computation of structured information processing (Yao, 2001; Yao, 2004; Yao, 2006; Yao, 2007). Many concrete models of granular computing have been proposed and studied. The main objective of the paper is to make further contribution along this line by investigating a logic approach to granular computing.

We introduce a logic language $\mathcal{L}$ to study granular computing in a logic setting. The language is an extension of the decision logic language used in rough set theory (Pawlak, 1991). Instead of expressing atomic formulas by a particular concrete type of conditions, we treat them as abstract notions to be interpreted in different applications. This flexibility enables us to describe granules in different applications. The language is interpreted in the Tarski's style through the notion of a model and satisfiability (Demri & Orlowska, 1997; Pawlak, 1991; Pawlak & Skowron, 2007; Yao, 2001; Yao, 2002). The model is defined as a pair consisting of a domain and knowledge about the domain. The meaning of a formula is given by a set of objects satisfying the formula. Like the representation of a concept by a pair of intension and extension, a granule is interpreted as a pair of a set of objects of the domain and a formula of the language $\mathcal{L}$. Thus, we can study granular structures in both a set-theoretic setting and a logic setting. The basic granules are represented by atomic formulas. An individual satisfies a formula if the individual has the properties as specified by the formula.

Rough set analysis and formal concept analysis are two concrete models of granular computing for knowledge representation and data analysis (Nguyen, Skowron & Stepaniuk, 2001; Pawlak, 1991;

Pawlak & Skowron, 2007; Wille, 1982; Wille, 1992; Yao, 2004). Rough set analysis studies the object-attribute relationships in an information table and formal concept analysis studies these relationships in single-valued and many-valued formal contexts. With the introduced language, the two theories can be interpreted in a unified way. On one hand, the two theories share high-level similarities in their treatments of granular structures. On the other hand, they use different atomic formulas, definable granules and granular structures formed by definable granules.

The rest of the paper is organized as follows. Section 2 introduces the formulation of the language $\mathcal{L}$. Section 3 introduces two examples of granular structures and their properties. In Section 4 and 5, we consider the interpretations of the language $\mathcal{L}$ in both rough set analysis and formal concept analysis. Concluding remarks are given in Section 6.

2. The Logic Language $\mathcal{L}$

In this section, we introduce a logic language $\mathcal{L}$ of granular computing by adopting and modifying the decision logic language used in rough set theory (Pawlak, 1991; Yao & Liau, 2002) .

Instead of defining an atomic formula specifically as a (attribute, value) pair (Pawlak, 1991) or an (attribute, relation, value) triplet (Yao & Liau, 2002), we postpone the definition of atomic formulas to particular applications. That is, we build the language based on a set of atomic formulas without giving a concrete physical meaning of them. This allows us to apply the language to study a much wider class of granular computing models.

The language $\mathcal{L}$ is in fact a 0-order, propositional language. Its syntax can be formally defined based on the standard propositional language. In general, one may consider first-order language. For the discussion of this paper, a propositional language is sufficient. Atomic formulas are the building blocks of the language $\mathcal{L}$, denoted by $\mathcal{A}= \{p, q...\}$. Each atomic formula may be interpreted as representing one piece of basic knowledge. We assume that they are the elementary units that one uses to represent and understand a real world problem. The physical meaning of atomic formulas becomes clearer in a particular application. In general, an atomic formula corresponds to one particular property of an individual under discussion. The construction of atomic formulas is an essential step of knowledge representation. The set of atomic formulas provides a basis on which more complex knowledge can be represented. Compound formulas can be built recursively from atomic formulas by using logic connectives. If φ and ψ are formulas, then so are $(\neg\varphi)$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \rightarrow \psi)$, and $(\varphi \leftrightarrow \psi)$.

The semantics of the language $\mathcal{L}$ is given in the Tarski's style by using the notions of a model and satisfiability. The model is defined as a pair $\mathcal{M}= (D, K)$, where D is a nonempty set of individuals called the domain of $\mathcal{L}$, denoted by $D= \{x, y...\}$, and K is available knowledge about individuals of D. For example, in the decision logic used in rough set theory, the knowledge K is a set of finite attributes used to describe the set of individuals of D. The satisfiability of an atomic formula by individuals of D is viewed as the basic knowledge describable by the language $\mathcal{L}$. In general, an individual satisfies a formula if the individual has the properties as specified by the formula. For an atomic formula p, we assume that the available knowledge K enables us to decide that an individual $x \in D$ either satisfies p, written as $x \models p$, or does not satisfy p, written as $x \not\models p$. Let φ and ψ be two formulas, the satisfiability of compound formulas is defined as follows:

$$
\begin{array}{lll}
(1).\ x \models \neg\varphi & \text{iff} & x \not\models \varphi, \\
(2).\ x \models \varphi \wedge \psi & \text{iff} & x \models \varphi \text{ and } x \models \psi, \\
(3).\ x \models \varphi \vee \psi & \text{iff} & x \models \varphi \text{ or } x \models \psi, \\
(4).\ x \models \varphi \rightarrow \psi & \text{iff} & x \models \neg\varphi \vee \psi, \\
(5).\ x \models \varphi \leftrightarrow \psi & \text{iff} & x \models \varphi \rightarrow \psi \text{ and } x \models \psi \rightarrow \varphi.
\end{array}
$$

To emphasize the roles played by the set of individuals D and the set of atomic formulas A, we also rewrite the language as $\mathcal{L}(\mathcal{A}, D)$.

A fundamental difference between the language $\mathcal{L}$ and other decision logic languages is the treatment of the set of atomic formulas. In early works, atomic formulas are defined using specific forms. For example, atomic formulas can be defined in an information table based on the values of attributes (Pawlak, 1991; Yao & Liau, 2002). In this study, we treat atomic formulas as abstract notions that need to be made concrete in different applications. Many concrete examples of the language can be obtained by various definitions of atomic formulas. The construction of the set of atomic formulas and the model $\mathcal{M}$ depends on a particular application. For modeling different problems, we may choose different sets of atomic formulas and models. When a model is switched from one to another, the structures of language $\mathcal{L}$ remain the same. The flexibility in semantics interpretation enables us to describe a variety of problems.

The model $\mathcal{M}$ describes the meaning of formulas in D. If formulas are interpreted in the model, then each formula becomes meaningful and describes properties of some individuals (Pawlak, 1991). The meaning of the formula is the set of all individuals having the properties described by the formula. If φ is a formula of $\mathcal{L}$, the meaning of φ in the model $\mathcal{M}$ is the set of individuals defined by:

$$m(\varphi) = \{x \in D \mid x \models \varphi\}. \tag{1}$$

That is, $m(\varphi)$ is the set of individuals satisfying the formula φ. This establishes a correspondence between logic connectives and set-theoretic operators. The following properties hold (Pawlak, 1991):

(C1). $m(\neg\varphi) = -m(\varphi)$,
(C2). $m(\varphi \wedge \psi) = m(\varphi) \cap m(\psi)$,
(C3). $m(\varphi \vee \psi) = m(\varphi) \cup m(\psi)$,
(C4). $m(\varphi \rightarrow \psi) = -m(\varphi) \cup m(\psi)$,
(C5). $m(\varphi \leftrightarrow \psi) = (m(\varphi) \cap m(\psi)) \cup (-m(\varphi) \cap -m(\psi))$,

where $-m(\varphi) = D - m(\varphi)$ denotes the set complement of $m(\varphi)$.

In the study of concepts (Smith, 1989; Sowa, 1984; Mechelen, Hampton, Michalski & Theuns, 1993), many interpretations have been proposed and examined. The classical view regards a concept as a unit of thoughts consisting of two parts, namely, the intension and extension of the concept (Demri & Orlowska, 1997; Mechelen, Hampton, Michalski & Theuns, 1993; Wille, 1992; Yao, 2004). By using the language $\mathcal{L}$, we obtain a simple and precise representation of a concept in terms of its intension and extension. That is, a concept is defined by a pair ($m(\varphi)$, φ). The formula φ is the description of properties shared by individuals in $m(\varphi)$, and $m(\varphi)$ is the set of individuals satisfying φ. 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 in a set-theoretic setting in terms of extensions. Following the classical view of concept, we also treat a granule as a pair ($m(\varphi)$, φ) of a set of individuals $m(\varphi) \subseteq D$ and a logic formula φ. Thus, we obtain both a set-theoretic description and a logic description of granules. In subsequent discussion, we use the two descriptions interchangeably. However, it should be noted that for the same set of individuals $X \subseteq D$, one may find more than one formula in the language such that $m(\varphi)= X$.

The language $\mathcal{L}$ provides a formal method for describing and interpreting rules in data mining and machine learning (Yao & Zhou, 2007; Yao, Zhou & Chen, 2007). In many situations, one is only interested in certain types of rules. For example, rules contain only the logical connective $\wedge$. This requires us to consider a restriction of the language $\mathcal{L}$ to certain logical connectives. In this study, we consider two sub-languages of $\mathcal{L}$. One uses only the conjunctive connective $\wedge$, written as $\mathcal{L}_{\wedge}(\mathcal{A}, \mathrm{D}, \wedge)$, and the other uses only the disjunctive connective $\vee$, written as $\mathcal{L}_{\vee}(\mathcal{A}, \mathrm{D}, \vee)$.

3. Overview of Granular Computing

Granular computing is a new area of research. Its main purpose is to model, state, and solve real world problems at multiple levels of granularity (Bargiela & Pedrycz, 2002; Lin, Yao & Zadeh, 2002; Yao, 2007). The fundamental issues, principles and methodologies of granular computing can be studied based on granular structures from different perspectives (Yao, 2001; Yao, 2004; Yao, 2006; Yao, 2007).

3.1 Granular Structures

A granular structure provides structured description of a system or an application under consideration.

Basic granules are the building blocks to form a granular to form a granular structure. They represent the basic human observations of a problem in the real world. The connections of different granules form different levels of a granular structure reflet structured knowledge. A granular structure at least contains three basic components (Yao, 2004; Yao, 2006; Yao, 2007):

- internal structure of a granule;
- collective structure of a family of granules;
- hierarchical structure of a web of granules.

The introduction of the language $\mathcal{L}$ enables us to study such structures in logic terms.

In the set-theoretic settings, all possible granules form a structure known as the power set of the domain D, written as 2^D. For a concrete granular computing model, one is only interested in a subset of 2^D which is considered "meaningful" in that model, called a subsystem $G \subseteq 2^D$. The choice of G relies on the domain knowledge. For example, in rough set analysis (Pawlak, 1991; Pawlak & Skowron, 2007), the granules of G are granules that are definable by a set of attributes, and they are the unions of some equivalence classes. In formal concept analysis (Ganter & Wille, 1999; Wille, 1982), granules of G are the extensions of formal concepts. In knowledge spaces theory (Falmagne, Koppen, Villano, Doignon & Johannesen, 1990), granules of G are the feasible knowledge statements. With respect to the language $\mathcal{L}$, a granule is definable if we can find a formula whose meaning set is that granule (Yao, 2007).

The internal structure of a granule represents the characterization of the granule. Analyzing the internal structure of a granule helps us to understand why individuals are draw together. In terms of the language $\mathcal{L}$, basic granules correspond to atomic formulas, and composite granules correspond to compound formulas. Granules in the same level are formed with respect to a certain level of abstraction and collectively show a certain structure. The collective structures are related to granules in other levels. We can classify granules by the number of atomic formulas in their intensions. In the sub-language $\mathcal{L}_\wedge(\mathcal{A}, D, \wedge)$, a granule involving k atomic formulas is called a k-conjunction. A k-conjunction granule is more general than its (k + 1)-conjunctions, and more specific than its (k − 1)-conjunctions. In the sub-language $\mathcal{L}_{\vee}(\mathcal{A}, D, \vee)$., a granule involving k atomic formulas is called a k-disjunction. In this case, a k-disjunction granule is more general than its (k − 1)-disjunctions, and more specific than its (k + 1)-disjunctions.

Granules in different levels are linked by the order relations, interpreted as “more general than” or “more specific than.” Granules can be ordered based on their generalities or sizes. For example, in the set-theoretic setting, the size of a granule can be defined by its cardinality. One can define operations on granules so that smaller granules can form larger granules, and larger granules can be decomposed into smaller granules. Recall that basic granules are defined by atomic formulas of the language $\mathcal{L}$. Smaller granules may be defined by formulas where atomic formulas are connected by the conjunctive connective $\wedge$, and larger granules may be defined by formulas where atomic formulas are connected by the disjunctive connective $\vee$. Granules in a higher level can be decomposed into many smaller granules with more details in a lower level, and conversely granules in a lower level can form more abstract larger granules in a higher level. The connections of different levels form a multilevel hierarchical structure. The graph representation of a granular structure is a lattice-like line diagram.

To sum up, granular structures are the results of a structured understanding, interpretation, and representation of a real-world problem. Each granular structure represents a particular point-of-view of the problem with multiple levels of granularity. A complete understanding of the problem requires a series of granular structures that should reflect multiple views with multiple levels (Chen & Yao, Yao, 2007).

3.2 The Triarchic Model of Granular Computing

There are three different perspectives of granular computing, namely, the philosophy, the methodology and the computation perspective. These three perspectives together form a triarchic model (Yao, 2001; Yao, 2004; Yao, 2006; Yao, 2007).

From the philosophy perspective, granular computing is a way of structured thinking that focuses on modeling human perception of the reality and cognitive process. It unifies two complementary philosophical views about the complexity of real-world problems, namely, the reductionist thinking and the systems thinking. Granular computing stresses the importance of conscious effects in thinking in terms of hierarchical structures.

The methodology perspective focuses on methods and strategies for finding structured solutions. As an effective method of structured problem solving, granular computing promotes systematic approaches and practical strategies that have been used effectively by humans for solving real-world problems. An important issue is the exploration of granular structures. This involves three basic tasks: constructing granular structures, working within a particular level of the structure, and switching between levels. The methodology of granular computing is inspired by human problem solving.

The computation perspective focuses on the implementation of computer based systems. As a paradigm of structured information processing (Bargiela & Pedrycz, 2002), two related notions, namely, representations and processes (Marr, 1982), need to be discussed. A representation is a formal description and specification of entities in information systems. A process can be interpreted as the computational actions occurred in information processing. For example, information granulation and computation with granules are two basic processes of granular computing.

3.3 Two Examples of Granular Structures

Based on the order relations between granules, there are at least two ways to construct a granular structure. As examples, we briefly discuss two granular structures called ∩-closure and ∩-closure (Yao & Zhou, 2007).

Let $GS_{\cap}(\mathcal{L}_{\wedge})$ denote the ∩-closure granular structure. We can formally defined it by the sub-language $\mathcal{L}_{\wedge}(\mathcal{A}, D, \wedge)$, written as:

$$\mathrm{GS}_{\cap}(\mathcal{L}_{\wedge}) = (Def(\mathcal{L}_{\wedge}(\mathcal{A}, D, \wedge)), \cap),$$

where $Def(\mathcal{L}_{\wedge}(\mathcal{A}, D, \wedge))$ is the family of granules defined by the sub-language $\mathcal{L}_{\wedge}(\mathcal{A}, D, \wedge)$.

The process of constructing an $\cap$-closure granular structure is a top-down process, which involves dividing a larger granule into smaller and lower level granules. Each granule is labeled by the formulas of the language $\mathcal{L}_\wedge(\mathcal{A}, D, \wedge)$. At the top level, the most general granule is labeled by the formula $\top$, which is satisfied by every individual, that is, $m(\top) = D$. The next level is the elementary granules labeled by atomic formulas. The intersections of two elementary granules produce the next level of granules labeled by the conjunction of the two atomic formulas, and so on. Finally, at the bottom level, we close the structure by the most specific granule labeled by the formula $\perp$, which is not satisfied by any object of the formal context, that is, $m(\perp) = \varnothing$. In the set-theoretic setting, the $\cap$-closure granular structure is in fact a closure system that contains D and is closed under set intersections.

Let $GS_\cup(\mathcal{L}_\vee)$ denote the $\cup$-closure granular structure. We can formally define it by the sub-language $\mathcal{L}_\vee(\mathcal{A}, D, \vee)$, written as:

$$GS_\cup(\mathcal{L}_\vee) = (Def(\mathcal{L}_\vee(\mathcal{A}, D, \vee)), \cup),$$

where $Def(\mathcal{L}_\vee(\mathcal{A}, D, \vee))$ is the family of granules defined by the sub-language $\mathcal{L}_\vee(\mathcal{A}, D, \vee)$.

The process of constructing a $\cup$-closure granular structure is a bottom-up process, which involves the process of forming a larger and higher level granule with smaller and lower level granules. At the bottom level, the most specific granule is labeled by the formula $\perp$, which is not satisfied by any individual. The upper level is the elementary granules labeled by atomic formulas. The unions of two elementary granules produce the upper level of granules labeled by the disjunction of the two atomic formulas, and so on. Finally, at the top level, we close the structure by the most general granule labeled by the formula $\top$.

4. Rough Set Analysis

Rough set analysis (Pawlak, 1991; Pawlak & Skowron, 2007) studies relationships between objects and their attribute values in an information table. We use rough set analysis as a concrete granular computing model to show the usefulness and the flexibility of the language $\mathcal{L}$.

4.1 Information Tables

An information table provides a convenient way to describe a finite set of objects by a finite set of attributes. Formally, an information table can be expressed as:

$$S = (U, At, \{V_a \mid a \in At\}, \{\{R_a\} \mid a \in At\}, \{I_a \mid a \in At\}),$$

Where

U is a finite nonempty set of objects,

At is a finite nonempty set of attributes,

V_a is a nonempty set of values for $a \in At$,

$\{R_a\}$ is a family of binary relations on V_a,

$I_a: U \rightarrow V_a$ is an information function.

Each information function I_a maps an object in U to a value of V_a for an attribute $a \in At$.

The above definition of an information table considers more knowledge and information about relationships between values of attributes. Each relation R_a can represent similarity, dissimilarity, or ordering of values in V_a (Demri & Orlowska, 1997). The equality relation = is only a special case of R_a. The standard rough set theory uses the trivial equality relation on attribute values (Pawlak, 1991).

Pawlak and Skowron (Pawlak & Skowron, 2007) consider a more generalized notion of an information table. For each attribute $a \in At$, a relational structure $\mathscr{R}_a$ over V_a is introduced. A language can be defined based on the relational structures. A binary relation is a special case of relational structures.

4.2 Granules in Rough Set Analysis

The indiscernibility relation is a fundamental notion in rough set theory. It indicates why objects are draw together to form granules. By using the language $\mathcal{L}$, we can formally define an indiscernibility relation in an information table. For a subset $\mathcal{A}_0 \subseteq \mathcal{A}$, two individuals $x, y \in U$ are indistinguishable if no formula in $\mathcal{A}_0$ can distinguish them, so they can be put into the same granule. Let us define a mapping from U to $\mathcal{A}$ as follows:

$$m'(x) = \{p \in \mathcal{A} \mid x \models p\}.$$

That is, $m'(x)$ is the set of atomic formulas satisfied by x. For a subset $\mathcal{A}_0 \subseteq \mathcal{A}$, the indiscernibility relation can be defined by:

$$X \sim_{\mathcal{A}o} y \quad \text{iff} \quad m'(x) \cap \mathcal{A}_0 = m'(y) \cap \mathcal{A}_0$$

The definition of indiscernibility relation is slightly different from the conventional definition (Pawlak, 1991). One can easily show that the conventional definition is a special case with a properly chosen $\mathcal{A}_0$.

Based on the indiscernibility relation, we can construct the language $\mathcal{L}$ by using an information table as the model $\mathcal{M}$. There are at least two types of granules that can be formally defined. They represent two different types of knowledge that one can derive from an information table (Yao, Zhou & Chen, 2007).

First, we consider individuals of the domain D as objects in the universe U. The set of atomic formulas are constructed from attribute-value pair. With respect to an attribute $a \in At$ and an attribute value $v \in V_a$, an atomic formula of the language $\mathcal{L}$ is denoted by (a, R_a, v). An object $x \in U$ satisfies an atomic formula by (a, R_a, v) if the value of x on attribute a is R_a -related to the value v, that is $I_a(x)\ R_a\ v$, we write:

$$x \models (a, R_a, v) \quad \text{iff} \quad I_a(x)\ R_a\ v.$$

We rewrite the language as $\mathcal{L}\ (\{(a, R_a, v)\}, U)$. The granule corresponding to the atomic formula (a, R_a, v), namely, its meaning set, is defined as:

$$m(a, R_a, v) = \{\mathrm{x} \in U \mid I_a(x)\ R_a\ v\ \}.$$

Granules corresponding to the atomic formulas are elementary granules in an information table. Granules corresponding to the compound formulas can be formally defined according to equation (1).

A subset or a granule $X \subseteq U$ is definable in an information table if and only if there exits a formula φ

in the language $\mathcal{L}(\{(a, R_a, v)\}, U)$ such that,

$$X = m(\varphi).$$

Otherwise, it is undefinable. Accordingly, the family of granules is defined as:

$$Def(\mathcal{L}(\{(a, R_a, v)\}, U)) = \{ m(\varphi) \mid \varphi \in \mathcal{L}(\{(a, R_a, v)\}, U)\}. \tag{2}$$

In this way, the language $\mathcal{L}$ only enables us to define certain subsets of U. For an arbitrary subset of U, we may not be able to construct a formula for it. In other words, depending on the set of atomic formulas, the language $\mathcal{L}$ can only describe a restricted family of subsets of U.

Second, we consider individuals of the domain D as object pairs in $U\times U$. With respect to an attribute $a\in At$, an atomic formula of the language $\mathcal{L}$ is denoted by (a, R_a). A pair of objects $(x, y) \in U \times U$ satisfies an atomic formula (a, R_a) if the value of x is R_a-related to the value of y on the attribute a, that is, $I_a(x)\ R_a\ I_a(y)$. We write:

$$(x,y) \models (a, R_a) \text{ iff } \ I_a(x)\ R_a\ I_a(y).$$

We rewrite the language as $\mathcal{L}(\{(a, R_a)\}, U\times U)$. The granule corresponding to the atomic formula (a, R_a), i.e., the meaning set, is defined as:

$$m(a, R_a) = \{(x,y)\in U \mid I_a(x)\ R_a\ I_a(y)\}.$$

Accordingly, the granules corresponding to the compound formulas of the language $\mathcal{L}(\{(a, R_a)\}, U\times U)$ can be defined by equation (1), and the family of all definable sets or granules is defined by equation (2).

For granules that are undefinable, it is impossible to construct a formula with the set as its meaning set. In order to characterize an undefinable set, one may approximate it from below and above by two definable sets, called lower and upper approximations in rough set theory.

4.3 Granular Structures in Rough Set Analysis

The indiscernibility relation used in the standard rough set analysis is an equivalence relation on the set of objects (Pawlak, 1991; Pawlak & Skowron, 2007; Nguyen, Skowron & Stepaniuk, 2001). Let E denote an equivalence relation that partitions the universe into disjoint subsets known as equivalence classes and is denoted by U/E. Equivalence classes are basic granules of the universe which can be interpreted using atomic formulas of the language $\mathcal{L}$ with the equality relation =. That is, we consider a language $\mathcal{L}$ $(\{(a, =, v)\}, U)$ with atomic formulas of the form of $(a, =, v)$.

An object $x \in U$ satisfies an atomic formula $(a, =, v)$ if the value of x on attribute a is v, that is, $I_a(x) = v$. We write:

$$x \models (a, =, v) \text{ iff } I_a(x) = v.$$

The granule corresponding to the atomic formula $(a, =, v)$ is:

$$m(a, =, v) = \{x \in U \mid I_a(x) = v\}.$$

The granule $m(a, =, v)$ is also referred to as the block defined by the attribute-value pair (a, v) (Grzymala-Busse, 2005). The granules corresponding to the compound formulas of the $\mathcal{L}$ $(\{(a, =, v)\}, U)$ can be defined by equation (1). Equivalence classes construct a subsystem of 2^U by taking set intersections, unions and complements of equivalence classes. This subsystem is in fact an σ-algebra of subsets of the universe, written as $\sigma(U/E)$. That is, it contains the empty set $\varnothing$, the entire set U, equivalence classes and is closed under set intersection, union and complement. The partition U/E is a base of $\sigma(U/E)$. The family of granules is exactly the σ-algebra $\sigma(U/E)$ (Yao, 2007). That is:

$$Def(\mathcal{L}(\{(a, =, v)\}, U)) = \sigma(U/E).$$

For a set that is undefinable in the universe U, one can approximate it by the lower and upper approximations. For a subset of objects $X \subseteq U$, we define a pair of lower and upper approximation as (Yao, 2007):

$$\underline{apr}(X) = \cup\{Y \mid Y \in Def(\mathcal{L}(\{(a, =, v)\}, U)), Y \in X\},$$
$$\overline{apr}(X) = \cap\{Y \mid Y \in Def(\mathcal{L}(\{(a, =, v)\}, U)), X \in Y\}.$$

Table 1: An Information Table

Object	Height	Hair	Eyes	Class
O_1	short	blond	blue	+
O_2	short	blond	brown	-
O_3	tall	red	blue	+
O_4	tall	dark	blue	-
O_5	tall	dark	blue	-
O_6	tall	blond	blue	+
O_7	tall	dark	brown	-
O_8	short	blond	brown	-

This is, $\underline{apr}(X)$ is the largest definable set contained in X, and $\overline{apr}(X)$ is the smallest definable set containing X.

We can construct the $\cap$-closure and $\cup$-closure granular structures in an information table by using granules in $\sigma(U/E)$. Formally, we can rewrite the $\cap$-closure granular structure as:

$$\mathrm{GS}_{\cap}(\mathcal{L}_{\wedge}) = (Def(\mathcal{L}_{\wedge}((a, =, v), U, \wedge)), \cap),$$

Similarly, we can rewrite the $\cup$-closure granular structure as:

$$\mathrm{GS}_{\cup}(\mathcal{L}_{\vee}) = (Def(\mathcal{L}_{\vee}((a, =, v), U, \vee)), \cup).$$

Example 1 Table 1 is an information table taken from (Quinlan, 1983). Each object is described by four attributes. The column labeled by "Class" denotes an expert's classification of the objects. The possible values for three attributes {Height, Hair, Eyes} are:

$$\begin{aligned} V_{\text{Height}} &= \{\text{short, tall}\}, \\ V_{\text{Hair}} &= \{\text{blond, dark, red}\}, \\ V_{\text{Eyes}} &= \{\text{blue, brown}\}. \end{aligned}$$

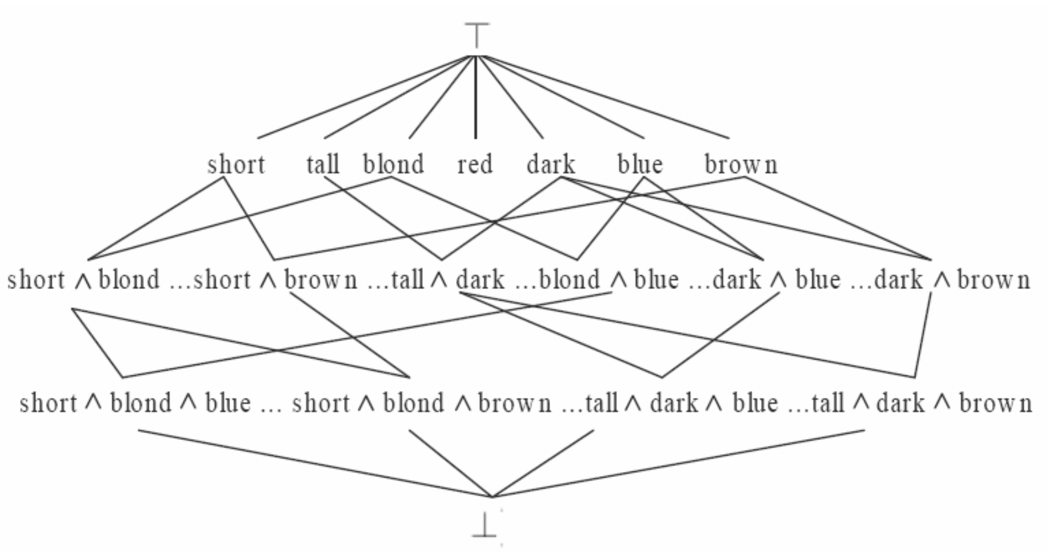
If the attribute "Height" is chosen, we can partition the universe into the following equivalence classes or elementary granules:

$$\{O_1, O_2, O_8\}, \quad \{O_3, O_4, O_5, O_6, O_7\},$$

corresponding to atomic formulas (Height, =, short) and (Height, =, tall), respectively. Similarly, the use of attribute "Hair" produces the following equivalence classes or elementary granules:

$$\{O_1, O_2, O_6, O_8\}, \{O_3\}, \{O_4, O_5, O_7\},$$

Figure 1: An example of ∩-closure granular structure



corresponding to atomic formulas (Hair, =, blond), (Hair, =, red), and (Hair, =, dark), respectively. For the attribute "Eyes", we have:

$$\{O_1, O_3, O_4, O_5, O_6\}, \{O_2, O_7, O_8\},$$

corresponding to atomic formulas (Eyes, =, blue) and (Eyes, =, brown), respectively.

Smaller granules are set intersections of elementary granules. For example, sets

$$\{O_1, O_2, O_8\} \cap \{O_1, O_2, O_6, O_8\} = \{O_1, O_2, O_8\}$$
$$\{O_3, O_4, O_5, O_6, O_7\} \cap \{O_4, O_5, O_7\} \cap \{O_2, O_7, O_8\} = \{O_7\},$$

are smaller granules with the corresponding compound formulas (Height, =, short) $\wedge$ (Hair, =, blond) and (Height, =, tall) $\wedge$ (Hair, =, dark) $\wedge$ (Eyes, =, brown), respectively.

Figure 1 draws part of the ∩-closure granular structure for Table 1. In the figure, we assume that an attribute appears at most once in each formula of. An atomic formula is simply represented by the attribute value. For example, the atomic formula (Height, =, short) is simply written as short.

Larger granules are set unions of elementary granules. For example, sets

$$\{O_1, O_2, O_8\} \cup \{O_4, O_5, O_7\} = \{O_1, O_2, O_4, O_5, O_7, O_8\},$$
$$\{O_3, O_4, O_5, O_6, O_7\} \cup \{O_3\} \cup \{O_2, O_7, O_8\} = \{O_2, O_3, O_4, O_5, O_6, O_7, O_8\},$$

are larger granules for the corresponding compound formulas (Height, =, short) $\vee$ (Hair, =, dark) and (Height, =, tall) $\vee$ (Hair, =, red) $\vee$ (Eyes, =, brown), respectively.

Figure 2: An example of ∪-closure granular structure

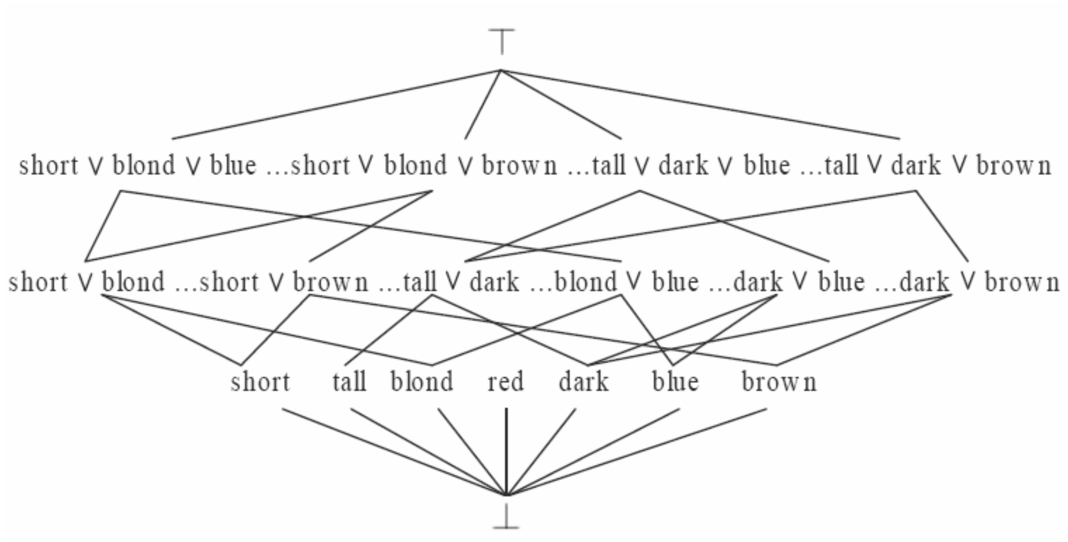

⊤
short ∨ blond ∨ blue …short ∨ blond ∨ brown …tall ∨ dark ∨ blue …tall ∨ dark ∨ brown
short ∨ blond …short ∨ brown …tall ∨ dark …blond ∨ blue …dark ∨ blue …dark ∨ brown
short tall blond red dark blue brown
⊥

Figure 2 draws part of the ∪-closure granular structure for Table 1.

5. Formal Concept Analysis

Formal concept analysis (Ganter & Wille, 1999) studies relationships between objects and their attributes in a formal context. In this section, we use formal concept analysis as another concrete granular computing model to further demonstrate the usefulness and flexibility of the language $\mathcal{L}$.

5.1 Formal Contexts

A formal context is a triple (O, A, R) consisting of two sets O and A and a binary relation $R \subseteq O \times A$ between O and A. The elements of O are called the objects, and the elements of A are called the attributes that the objects might have. If $(x, a) \in R$, we say that the object x has the attribute a; we also write it as xRa. A formal context is equivalent to a binary information table in rough set analysis.

In many situations, attributes are not just properties which objects may or may not have. Attributes such as "Size", "Prize", and "Weight" have values. A formal context with many-valued attributes can be defined as a many-valued context (O, A, V, R) consisting of sets O, A, V and a ternary relation $R \subseteq O \times A \times V$. The elements of A are many-valued attributes and the elements of V are their possible attribute values. A many-valued formal context satisfies the following condition:

$$(x, a, v_1) \in R \wedge (x, a, v_2) \in R \Rightarrow (v_1 = v_2).$$

In the language of rough set analysis, one can define a partial map from O to V with respect to an attribute a of a many-valued context (O, A, V, R). If $(x, a, v) \in R$, we write $I_a(x) = v$.

Thus, many-valued context can be represented by an information table, the rows of the table are the objects and columns are attributes. The entry in row x and column a represents the attribute value $I_a(x)$. If the attribute a does not have a value for object x, then there is no entry. A many-valued formal context can be translated into a single-valued context through a process called conceptual scaling which contains two essential steps.

In the first step, each attribute of a many-valued context is interpreted into a scale. A scale for the attribute a of a many-valued context can be defined as a single-valued context $S_a = (O_a, A_a, R_a)$. The objects of a scale are called scale values, and attributes are called scale attributes. The binary relation $R_a \subseteq O_a \times A_a$, can be interpreted differently according to different types of scales (Ganter & Wille, 1999). For better understanding, we explain three elementary scales in a simple example.

Example 2 Table 1 is an example of a many-valued context for some televisions. The scale contexts of three attributes, "Type", "Clarity", and "Price" are given in Table 2.

The first type of scale is called nominal scale. It is used to scale attributes with the values that mutually exclude each other. The attribute "Type" with the values {CRT, LCD, Plasma} in Table 2 uses this kind of scale.

The second type of scale is called ordinal scale. It is used to scale attributes with the values that are ordered and each value implies the weak ones. The attribute "Clarity" with the values {clear, very clear, extremely clear} in Table 2 uses this kind of scale.

The third type of scale is called interordinal scale. It is used to scale attributes with the values that have bipolar orderings. The attribute "Price" with the values {$\$0 \leq$ Price $< \$1000$, $\$1000 \leq$ Price $< \$3000$, $\$3000 \leq$ Price $< \$4000$} in Table 3 uses this kind of scale.

Table 2: An example of a many-valued formal context

Object	Type	Clarity	Price
tv_1	CRT	clear	\$1000
tv_2	LCD	very clear	\$2500
tv_3	Plasma	extremely clear	\$3500
tv_4	LCD	very clear	\$3900

The second step of conceptual scaling is to join the scales together to make a single-valued context. In the case of plain scaling, the object set O of the many-valued context remains unchanged, every many-valued attribute a is replaced by the scale attributes of the scale (O_a, A_a, R_a). That is, the attribute set of the derived single-valued context is the disjoint union of scale attribute sets. We can rewrite the scale attribute set A_a to $\dot{A}_a = \{a\} \times A_a$ to ensure that those scale attribute sets are disjoint. Therefore, the derived single-valued formal context from many-valued context (O, A, V, R) with respect to plain scaling can be defined as (O, B, R') with:

$$B = \cup_{a \in A} \dot{A}_a$$

and

$$x\, R'(a,b) \Leftrightarrow I_a(x) = v \wedge v\, R_a\, b$$

That is, an object $x \in O$ has the attribute value $b \in B$ with respect to the attribute $a \in A$ in the context (O, B, R') if and only if x has the attribute value v with respect to attribute a in the many-valued context (O_a, A_a, R_a), and v is R_a related to b in the scale context S_a. Table 4 shows the derived single-valued context by combining the many-valued context of Table 2 and the scale contexts of Table 3.

5.2 Granules in Formal Concept Analysis

The processes of defining granules and granular structures are different in rough set analysis and formal concept analysis. They build granules and granular structures based on two different interpretations of the notion of definability. In rough set analysis, a definable granule is the union of some equivalence classes. In formal concept analysis, one is interested in granules of objects that are extensions of formal concepts. A concept derived from a formal context is a pair of a set of objects and a set of attributes,

Table 3: Examples of scale contexts

S_{Type}	CRT	LCD	Plasma
CRT	×		
LCD		×	
Plasma			×

$S_{Clarity}$	≤ clear	≤very clear	≤extremely clear
clear	×	×	×
very clear		×	×
extremely clear			×

S_{Price}	cheap	mid-range	expensive
\$0 ≤ Price < \$1000	×		
\$1000 ≤ Price < \$3000		×	
\$3000 ≤ Price < \$4000			×

called the extension and intension of the concept. Furthermore, the extension and intension are mutually definable, that is, the intension of the extension is the extension of the intension and vice versa.

For a many-valued formal context (O, A, V, R), one first needs to transfer this many-valued context into a single-valued context (O, B, R'). The formal concepts constructed from this derived single-valued context are then interpreted as the formal concepts of the many-valued context.

Since a many-valued formal context (O, A, V, R) can be translated into a single-valued context (O, B, R'), it is sufficient to consider the construction process of formal concepts in (O, B, R'). We can construct the language $\mathcal{L}$ by using a single-valued formal context as the model $\mathcal{M}$, O as the domain D, and the set of atomic formulas is given by $\mathcal{A} = \{b \mid b \in B\}$. That is, atomic formula is denoted as the attribute value (b) or simply b. An object $x \in O$ satisfies an atomic formula b if the object has the attribute b, we write:

$$x \models b \quad \text{iff} \quad xRb.$$

We rewrite the language as $\mathcal{L}(\{b\}, O)$. In Table 2, examples of atomic formulas are (Type, CRT) and (Clarity, ≤very clear). For simplicity, we also write them as "CRT" and "≤very clear", respectively.

By using the language $\mathcal{L}$, we can formally define a formal concept in a formal context. For a set X

Table 4: Derived single-valued context

	Type			Clarity			Price		
	a	b	c	d	e	f	g	h	i
tv_1	×			×			×		
tv_2		×		×	×			×	
tv_3			×	×	×	×			×
tv_4		×		×	×				×

$\subseteq O$ of objects, we define a mapping $O \to \mathcal{A}$ as:

$$m'(X) = \{b \in \mathcal{A} \mid x \models b \quad \text{for all} \quad x \in X\},$$

that is, the set of atomic formulas satisfied by all the objects in X. Correspondingly, for a set $P \subseteq \mathcal{A}$ of atomic formulas, we define a mapping $\mathcal{A} \to O$ as:

$$\begin{aligned} m(P) &= \{x \in O \mid x \models b \quad \text{for all} \quad b \in P\} \\ &= \{ x \in O \mid x \models \wedge_{b \in P}\, b \}, \end{aligned}$$

that is, the set of objects satisfies all the atomic formulas in P. With these two mappings being defined, a formal concepts can be defined as a pair satisfying the condition:

$$(X, P) = \{X \subseteq O, P \subseteq \mathcal{A} \mid X = m(P), P = m'(X) \}.$$

The set of objects $X = extent(X, P)$ is the extension and the set of atomic formulas $P = intent(X, P)$ is the intension of the formal concept. By using logic formulas, we can rewrite a formal concept as $(X, \wedge_{b \in P}$ $b)$.

For an atomic formula $b \in \mathcal{A}$, we can derive a formal concept $(m(\{b\}), m'(m(\{b\})))$. A subset or a granule $X \subseteq O$ is definable in a formal context if and only if there exits a subset of atomic formulas $b \in \mathcal{A}$ in the language $\mathcal{L}(\{b\}, O)$ such that (X, P) is a formal concept , that is, $X = m(P)$ and $P = m'(X)$. Otherwise, it is undefinable. Let $\mathcal{B}$ (O, B, R') or simply $\mathcal{B}$ denotes the set of all formal concepts of the context, the family of definable granules is given by:

$$Def(\mathcal{L}(\{b\}, O)) = \{extent(X, P) \mid (X, P) \in \mathcal{B} \}.$$

If a formal context (O, B, R') is treated as a binary information table, one can easily observe a close relationship between rough set analysis and formal concept analysis. A definable set of objects in formal concept analysis is a definable set in rough set analysis, but the reverse in not true.

5.3 Granular Structures in Formal Concept Analysis

The family of all formal concepts forms a complete lattice called a concept lattice through which the relationships between formal concepts can be visualized (Ganter & Wille, 1999). The meet and join of the lattice are defined based on the set-theoretic operators of intersection ($\cap$), union ($\cup$) and the mappings between object set O and atomic formula set A, written as:

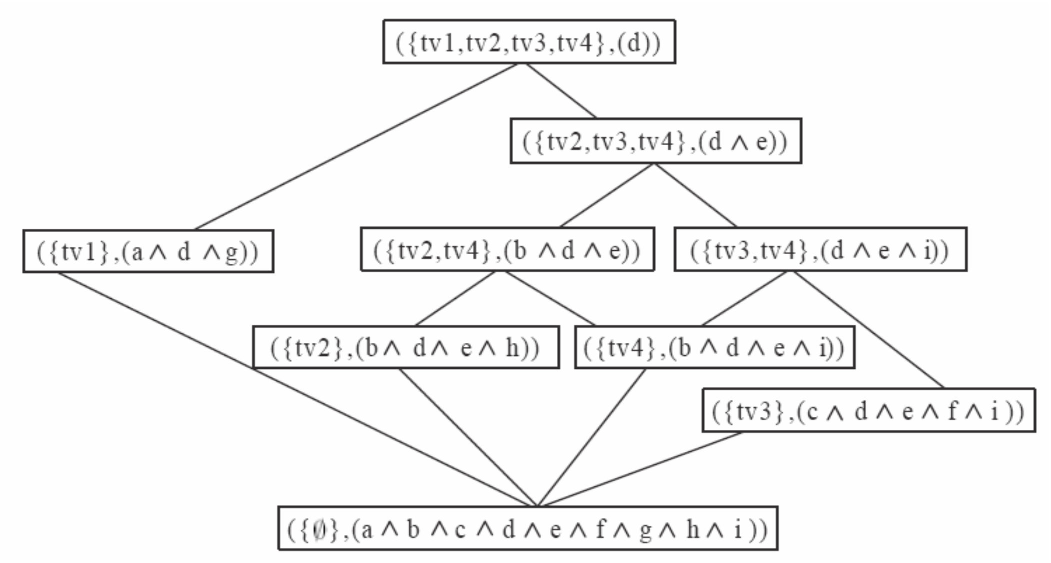
$$(X1, P1) \wedge (X2, P2) = (X1 \cap X2, m'(m(P1 \cup P2))),$$
$$(X1, P1) \vee (X2, P2) = (m(m'(X1 \cup X2)), P1 \cap P2).$$

The order relation of the concept lattice can be defined as follows. For two formal concepts ($X1$, $P1$) and ($X2$, $P2$), if $X1 \subseteq X2$ (which is equivalent to $P2 \subseteq P1$), then ($X2$, $P2$) is a superconcept of ($X1$, $P1$) and ($X1$, $P1$) is a subconcept of ($X2$, $P2$), written as $(X1, P1) \leq (X2, P2)$.

From the viewpoint of granular computing, the extensions of superconcepts are larger granules which may be decomposed into smaller granules as extensions of subconcepts. We consider a concept lattice as an $\cap$-closure granular structure which only includes granules that are extensions of formal concepts. Each granule in a concept lattice can be labeled by formulas of the language $\mathcal{L}_\wedge$.

Example 3 The process of forming a concept lattice from Table 3 can be illustrated as follows. If attribute "≤clear" is chosen, objects satisfied this attribute construct the most general granule which includes all the objects of this context, the corresponding formal concept is ({tv_1, tv_2, tv_3, tv_4}, ≤clear), where "≤clear" is an atomic formula. The second level are the granules whose intensions only include the conjunction of two atomic formulas, the corresponding formal concept is ({ tv_2, tv_3, tv_4}, ≤clear $\wedge$ ≤very clear). The third level are the granules whose intensions only include the conjunction of three atomic formulas, the corresponding formal concept is ({tv_1}, CRT $\wedge$ ≤clear $\wedge$ cheap), ({tv_2, tv_4}, LCD

Figure 3: Concept Lattice of Table 4



∧ ≤clear ∧ ≤very clear) and ({tv_3, tv_4 }, ≤clear ∧ ≤very clear ∧ expensive). The intersections of granules produce the smaller granules in the fourth level and so on. For example, the set,

$$\{tv_2, tv_4\} \cap \{tv_3, tv_4\} = \{tv_4\},$$

is a smaller granule corresponding to the formal concept ({tv_4}, LCD ∧ ≤clear ∧ ≤very clear ∧ expensive). Finally, the most specific granule is the empty set ∅ corresponding to the conjunction of all atomic formulas as its intension.

The line diagram in Figure 3 represents the concept lattice of Table 4 which includes 9 formal concepts. The intensions of each formal concept are labeled by formulas of the language $\mathcal{L}_\wedge$.

6. Conclusions

Granular computing models human problem solving with different-sized grains of knowledge. Basic granules are the elementary units of granular computing represent the basic pieces of knowledge. In order to formally describe basic granules and the construction of granular structures, we introduce a logic language $\mathcal{L}$. Atomic formulas correspond to basic granules. Other formulas of the language are recursively constructed from a set of atomic formulas. The meaning of formulas is defined in Tarski's style by using the model $\mathcal{M}$ = (D, K). Based on the knowledge K of the model, it is assumed that an individual in the domain D either satisfies a formula or does not satisfy a formula. A granule is jointly described by a pair $(m(\varphi), \varphi)$ consisting of a formula φ of the language $\mathcal{L}$ and a subset $m(\varphi)$ of the domain of $\mathcal{M}$.

Depending on particular applications, we can interpret the language by using different types of atomic formulas. We demonstrate the usefulness of the language $\mathcal{L}$ in two concrete granular computing models, namely, rough set analysis and formal concept analysis. The two theories explore different types of definability of a granule and consequently different granular structures. The notion of definability of rough set analysis is weaker than formal concept analysis.

The logic based interpretation of rough set analysis and formal concept analysis show their differences and high-level similarities. The unified study of the two theories not only demonstrates the potential of the logic approach to granular computing, but also brings more insights into data analysis using the two theories.

Acknowledgements

This study is partially supported by an NSERC Canada Discovery grant. The authors thank Professor Yingxu Wang for encouragement and anonymous reviewers for their constructive comments.

References

Bargiela, A. & Pedrycz W. (2002). Granular computing: an introduction, Kluwer Academic Publishers, Boston, 2002.

Chen, Y.H. & Yao, Y.Y. A multiview approach for intelligent data analysis based on data operators, *Information Sciences*, to appear.

Demri, S. & Orlowska, E. (1997). Logical analysis of indiscernibility, in: *Incomplete Information: Rough Set Analysis*, Orl owska, E. (Ed.), Physica Verlag, Heidelberg, pp. 347-380.

Falmagne, J.-C., Koppen, M., Villano, M., Doignon, J.-P. & Johannesen, L. (1990). Introduction to knowledge spaces: how to build, test and search them. *Psychological Review*, pp. 201-224.

Ganter, B. & Wille, R. (1999). *Formal Concept Analysis, Mathematical Foundations,* Springer, Berlin.

Grzymala-Busse, J.W. (2005). Incomplete data and generalization of indiscernibility relation, definability, and approximations, *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Proceedings of 10th International Conference, LNAI* **3641**, pp. 244-253.

Lin, T.Y., Yao, Y.Y. & Zadeh, L.A. (Eds.) (2002). *Data Mining, Rough Sets and Granular Computing,* Physica-Verlag, Heidelberg.

Marr, D. (1982). *Vision, A Computational Investigation into Human Representation and Processing of Visual Information,* W.H. Freeman and Company, San Francisco.

Nguyen, H. S., Skowron, A. & Stepaniuk, J. (2001). Granular computing: a rough set approach, *Computational Intelligence,* **17**, pp. 514-544.

Pawlak, Z. (1991). *Rough Sets - Theoretical Aspects of Reasoning about Data*, Kluwer Publishers, Boston.

Pawlak, Z. & Skowron, (2007). A. Rough sets: some extensions, *Information Science*, **177**, pp. 28-40.

Quinlan, J. R. (1983). Learning efficient classification procedures and their application to chess end-games, in: *Machine Learning: An Artificial Intelligence Approach*, Michalski, J.S. et al.(Eds.), Morgan Kaufmann, **1**, pp. 463-482.

Smith, E.E. (1989). Concepts and induction, in: *Foundations of Cognitive Science,* Posner, M.I. (Ed.), The MIT Press, Cambridge, Massachusetts, pp. 501-526.

Simon, H. A. & Kaplan, C. A. (1989). Foundations of cognitive science, in: *Foundations of Cognitive Science*, Posner, M.I. (Ed.), The MIT Press, Cambridge, Massachusetts, pp. 1-47.

Sowa, J.F. (1984). *Conceptual Structures, Information Processing in Mind and Machine,* Addison-Wesley, Reading, Massachusetts.

Van, Mechelen, I., Hampton, J., Michalski, R.S. & Theuns, P. (Eds.) (1993). *Categories and Concepts, Theoretical Views and Inductive Data Analysis*, Academic Press, New York.

Wang, Y. (2003). Cognitive informatics: a new transdisciplinary research field, *Brain and Mind: A Transdisciplinary Journal of Neuroscience and Neurophilosophy,* **4**, pp. 115-127.

Wang, Y. (2003). On cognitive informatics, *Brain and Mind: A Transdisciplinary Journal of Neuroscience and Neurophilosophy,* **4**, pp. 151-167.

Wang, Y. (2007). The theoretical framework of cognitive informatics, *The International Journal of Cognitive Informatics and Natural Intelligence*, **1**, pp. 1-27.

Wang, Y. (2007). Cognitive informatics: exploring theoretical foundations for natural intelligence, neural Informatics, autonomic computing, and agent systems, *The International Journal of of Cognitive Informatics and Natural Intelligence,* **1**, pp. 1-10.

Wille, R. (1982). Restructuring lattice theory: an approach based on hierarchies of concepts, in: *Ordered Sets*, Rival, I. (Ed), Reidel, Dordrecht-Boston, pp. 445-470.

Wille, R. (1992). Concept lattices and conceptual knowledge systems, *Computers Mathematics with Applications,* **23**, pp. 493-515.

Yao, J.T. (2007). A ten-year review of granular computing, *Proceedings of the 3rd IEEE International Conference on Granular Computing*.

Yao, Y.Y. (2001). Modeling data mining with granular computing, *Proceedings of the 25th Annual International Computer Software and Applications Conference*, pp. 638-643.

Yao, Y.Y. (2001). Information granulation and rough set approximation, *International Journal of Intelligent Systems,* **16**, pp. 87-104.

Yao, Y.Y. (2004). Granular computing, *Proceedings of The 4th Chinese National Conference on Rough Sets and Soft Computing,* **31**, pp. 1-5.

Yao, Y.Y. (2004). A comparative study of formal concept analysis and rough set theory in data analysis, *International Conference on Rough Sets and Current Trends in Computing (RSCTC'2004),* pp. 59-68.

Yao, Y.Y. (2004). A partition model of granular computing, *LNCS Transactions on Rough Sets, LNCS* **3100**, pp. 232-253.

Yao, Y.Y. (2004). Concept formation and learning: a cognitive informatics perspective, *Proceedings of the 3rd IEEE International Conference on Cognitive Informatics*.

Yao, Y.Y. (2006). Three perspectives of granular computing, *The Proceedings, International Forum on Theory of GrC from Rough Set Perspective, Journal of Nanchang Institute of Technology*, **25**, pp. 16-21.

Yao, Y.Y. (2006). Granular computing and cognitive informatics, *Proceedings of the Fifth IEEE International Conference on Cognitive Informatics*, pp. 17-18.

Yao, Y.Y. (2007). A note on definability and approximations, *Transactions on Rough Sets*, pp. 274-282.

Yao, Y.Y. (2007). The art of granular computing, *Proceeding of the International Conference on Rough Sets and Emerging Intelligent Systems Paradigms, LNAI*, **4585**, pp. 101-112.

Yao, Y.Y. & Liau, C.-J. (2002). A generalized decision logic language for granular computing, *FUZZ-IEEE'02 in The 2002 IEEE World Congress on Computational Intelligence*, pp. 1092-1097.

Yao, Y.Y. & Zhou, B. (2007). A logic language of granular computing, *Proceedings of the 6th IEEE International Conference on Cognitive Informatics*, pp. 178-185.

Yao, Y.Y., Zhou, B. & Chen, Y.H. (2007). Interpreting low and high order rules: a granular computing approach. *Proceedings of International Conference on Rough Sets and Emerging Intelligent System Paradigms (RSEISP'07), Lecture Notes in Artificial Intelligence, LNAI* **4585**, pp. 371-380.

Zadeh, L.A. (1997). Towards a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic, *Fuzzy Sets and Systems*, **19**, pp. 111-127.