Advanced Information and Knowledge Processing

Georg Peters *Editors*
Pawan Lingras
Dominik Ślęzak
Yiyu Yao

Advanced Information and Knowledge Processing

For further volumes:
www.springer.com/series/4738

Georg Peters • Pawan Lingras • Dominik Ślęzak •
Yiyu Yao
Editors

Rough Sets: Selected Methods and Applications in Management and Engineering

Springer

Editors
Georg Peters
Munich University of Applied Sciences
Munich, Germany

Dominik Ślęzak
University of Warsaw
Warsaw, Poland

Pawan Lingras
Department of Mathematics and
Computer Science
St. Mary's University
Halifax, Canada

Yiyu Yao
Department of Computer Science
University of Regina
Regina, Canada

ISSN 1610-3947 Advanced Information and Knowledge Processing
ISBN 978-1-4471-2759-8 e-ISBN 978-1-4471-2760-4
DOI 10.1007/978-1-4471-2760-4
Springer London Dordrecht Heidelberg New York

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

Library of Congress Control Number: 2012932960

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Rough Set Theory was introduced by Pawlak in the early 1980's. In the last quarter century it has become an important part of soft computing and has proved its relevance in many real-world applications.

While initially most of the articles on Rough Sets had been centered on theory, currently the focus of the research has shifted to practical usage of mathematical advances. A state of the art survey on Rough Sets from an application perspective is highly desirable but still missing.

The book is written for business and industry professionals who would like to evaluate the potential of Rough Sets. The intended readership includes (1) managers looking for methods to improve their businesses, (2) researchers in industrial laboratories and think tanks who are investigating new methods to enhance efficiency of their solutions, (3) researchers at universities who want to use Rough Sets to solve real-world problems and seek for guidance on how to describe their ideas in a way understandable for the industry readers.

The approach to Rough Sets presented in the following chapters differs from the most of articles in other publications on this subject. This book focuses on practical use cases backed by sound theory, in contrast to the presentation of a theory applied to a problem. Furthermore, it provides a unified view and easily accessible description of applications.

The book covers methods in data analysis, decision support, as well as management and engineering in order to show the great potential of Rough Sets in almost any domain. The number of real-world applications of Rough Sets has increased significantly and goes probably into hundreds. Hence the book can only give a sample of the selected practically relevant case studies.

The editors of the book would like to acknowledge the authors of all chapters for their excellent contributions. Special thanks go to Mr. Sebastian Widz for his great help with revising and indexing the materials.

Munich, Germany — Georg Peters
Halifax, Canada — Pawan Lingras
Warsaw, Poland — Dominik Ślęzak
Regina, Canada — Yiyu Yao

Contents

Part I Foundations of Rough Sets

An Introduction to Rough Sets 3
Yiyu Yao and Dominik Ślęzak

Part II Methods and Applications in Data Analysis

Applying Rough Set Concepts to Clustering 23
Pawan Lingras and Georg Peters

Rough Clustering Approaches for Dynamic Environments 39
Fernando Crespo, Georg Peters, and Richard Weber

Feature Selection, Classification and Rule Generation Using Rough Sets 51
Haider Banka and Sushmita Mitra

Part III Methods and Applications in Decision Support

Three-Way Decisions Using Rough Sets 79
Yiyu Yao

Rough Set Based Decision Support—Models Easy to Interpret 95
Sebastian Widz and Dominik Ślęzak

Part IV Methods and Applications in Management

Financial Series Forecasting Using Dual Rough Support Vector Regression 115
Pawan Lingras, Cory Butz, and Parag Bhalchandra

Grounding Information Technology Project Critical Success Factors Within the Organization 129
M. Gordon Hunter and Georg Peters

Workflow Management Supported by Rough Set Concepts 143
Georg Peters and Roger Tagg

Part V Methods and Applications in Engineering

Rough Natural Hazards Monitoring 163
Marek Sikora and Beata Sikora

Nearness of Associated Rough Sets 181
Sheela Ramanna and James F. Peters

Contributor's Biography 207

Index 213

Contributors

Haider Banka Department of Computer Science and Engineering, Indian School of Mines, Dhanbad, Jharkhand, India

Parag Bhalchandra School of Computational Sciences, Swami Ramanand Teerth Marathwada University, Nanded, India

Cory Butz Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada

Fernando Crespo Industrial Engineering School, Universidad de Valparaíso, Santiago, Chile

M. Gordon Hunter Faculty of Management, University of Lethbridge, Lethbridge, Alberta, Canada

Pawan Lingras Department of Mathematics and Computer Science, Saint Mary's University, Halifax, Canada; School of Computational Sciences, Swami Ramanand Teerth Marathwada University, Nanded, India

Sushmita Mitra Machine Intelligence Unit, Indian Statistical Institute, Kolkata, India

Georg Peters Department of Computer Science and Mathematics, Munich University of Applied Sciences, Munich, Germany

James F. Peters Computational Intelligence Laboratory, Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB, Canada

Sheela Ramanna Department of Applied Computer Science, University of Winnipeg, Winnipeg, MB, Canada; Computational Intelligence Laboratory, Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB, Canada

Beata Sikora Institute of Mathematics, Silesian University of Technology, Gliwice, Poland

Marek Sikora Institute of Innovative Technologies EMAG, Katowice, Poland; Institute of Informatics, Silesian University of Technology, Gliwice, Poland

Dominik Ślęzak Institute of Mathematics, University of Warsaw, Warsaw, Poland; Infobright Inc., Poland, Warsaw, Poland

Roger Tagg School of Computer and Information Science, University of South Australia, Adelaide, Australia

Richard Weber Department of Industrial Engineering, Universidad de Chile, Santiago, Chile

Sebastian Widz Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland; XPLUS SA, Warsaw, Poland

Yiyu Yao Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada

Part I
Foundations of Rough Sets

An Introduction to Rough Sets

Yiyu Yao and Dominik Ślęzak

Abstract Fundamental philosophy, concepts and notions of rough set theory (RST) are reviewed. Emphasis is on a constructive formulation and interpretation of rough set approximations. We restrict our discussions to classical RST introduced by Pawlak, with some brief references to the existing extensions. Whenever possible, we provide multiple equivalent definitions of fundamental RST notions in order to better illustrate their usefulness. We also refer to principles of RST based data analysis that can be used to mine data gathered in information tables.

1 Introduction

Rough set theory (RST) provides a mathematical formalism and means for representing and analyzing data [1–4]. Several unique features of the theory make it attractive to practitioners. RST is simple, elegant and, at the same time, flexible. It has been successfully applied in many areas, including those described in this book [5–10], as well as many others [11–14].

It is also worth mentioning that quite a few RST based data analysis software packages have been developed, including LERS,[1] ROSETTA,[2] RSES[3]/RSES-lib[4]

[1] http://lightning.eecs.ku.edu/LERS.html.

[2] http://www.lcb.uu.se/tools/rosetta/.

[3] http://logic.mimuw.edu.pl/~rses/.

[4] http://rseslib.mimuw.edu.pl/.

Y. Yao (✉)
Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada S4S 0A2
e-mail: yyao@cs.uregina.ca

D. Ślęzak
Institute of Mathematics, University of Warsaw, Banacha 2, 02-097 Warsaw, Poland
e-mail: slezak@mimuw.edu.pl

D. Ślęzak
Infobright Inc., Poland, Krzywickiego 34 pok. 219, 02-078 Warsaw, Poland
e-mail: slezak@infobright.com

G. Peters et al. (eds.), *Rough Sets: Selected Methods and Applications in Management and Engineering*, Advanced Information and Knowledge Processing,
DOI 10.1007/978-1-4471-2760-4_1, © Springer-Verlag London Limited 2012

and RoughICE.[5] One can apply them to mine data sets in a tabular form, which can be easily extracted from a relational database or derived using some extraction and transformation techniques.

RST leads toward a unique methodology of intelligent knowledge discovery. It may be useful in, for instance, categorization, approximation, concept formation and inductive learning [15–18]. In this chapter, we focus on the following aspects related to the foundations of RST:

– Description of concepts using a decision logic language.
– Discernibility of objects based on equivalence relations.
– Rough set approximations of sets and concepts.
– Dependencies between subsets of attributes.

We demonstrate basic notions and ideas through examples. A reader interested in more details is referred to the seminal book by Pawlak [2].

2 Objects and Concepts

RST uses a simple knowledge and data representation scheme called an information table (or an information system [19]). Two important notions related to the analysis of data gathered in information tables are the decision logic language and the indiscernibility of objects.

2.1 Information Tables

It is usually assumed that an information table contains a finite set of objects described by using a finite set of attributes.

Definition 1 An information table is a tuple:

$$M = (U, At, \{V_a | a \in At\}, \{I_a | a \in At\}), \tag{1}$$

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 an attribute $a \in At$, and $I_a : U \longrightarrow V_a$ is an information function, which maps objects in U into values in V_a.

The value of an object $x \in U$ on an attribute $a \in At$ is denoted by $I_a(x)$. In general, for a subset of attributes $A \subseteq At$, we use $I_A(x)$ to denote the vector of values of x on A.

The notion of an information table can be extended in many ways. As an example of such extension, consider the notion of a decision table (or a classification table) as

[5] http://www.mimuw.edu.pl/~bazan/roughice/?sLang=en.

Table 1 An information table

	a	b	c	d
o_1	0	0	1	+
o_2	0	0	1	+
o_3	0	1	0	−
o_4	1	0	0	+
o_5	1	0	1	+
o_6	1	1	2	+
o_7	1	2	1	−
o_8	1	2	1	−
o_9	1	2	1	−

a special information table in which the set of attributes is divided into two disjoint subsets called condition attributes C and decision (or classification) attributes D, namely, $At = C \cup D$ and $C \cap D = \emptyset$.

Example 1 Consider an information table illustrated by Table 1, in which rows represent objects and columns represent attributes. The set of objects is given by $U = \{o_1, o_2, \ldots, o_9\}$, the set of attributes by $At = \{a, b, c, d\}$, and the sets of attribute values are $V_a = \{0, 1\}$, $V_b = \{0, 1, 2\}$, $V_c = \{0, 1, 2\}$ and $V_d = \{-, +\}$. Each cell in the information table is a value of an object on an attribute. For instance, we have $I_a(o_1) = 0$, $I_d(o_3) = -$, and so on. We have also $I_{\{a,b\}}(o_1) = (0, 0)$, $I_{\{b,c,d\}}(o_2) = (0, 1, +)$ and so on. We can divide the set of attributes into two disjoint sets $C = \{a, b, c\}$ and $D = \{d\}$ to produce a decision table, where C is the set of three condition attributes and D is the set containing a single decision attribute.

Another possible aspect of extending the original notion of an information table relates to the values of attributes. If the information function I_a is a partial function, i.e., it is not defined or missing for certain objects, we obtain an incomplete information table [20]. If I_a maps an object to a subset of attribute values, we obtain a set-valued information table [21]. Some authors go even further and, for example, combine the set-valued and interval-valued attributes within the framework of incomplete information databases [22]. In this chapter, for the sake of clarity, we only consider the simplest case of complete information tables. However, various extensions may be useful for specific applications, without sacrificing the important advantage of clarity of RST.

2.2 Decision Logic Language

One of the key aspects of RST relates to the notion of a concept. There are many ways of interpreting concepts [4]. We adopt the view of representing a concept

jointly by a pair of intension and extension [23]. The intension (also called the description) is an intrinsic property of a concept, based on which one can determine if an object is an instance of the concept. The extension (also called the support) is the set of instances of a concept. Orłowska [24] and Pawlak [2] use a decision logic language $\mathcal{L}$ in order to define concepts in information tables. The intension and extension of a concept can be precisely defined as a formula in $\mathcal{L}$ and the meaning of this formula, respectively [17].

Definition 2 A decision logic language $\mathcal{L}$ in an information table can be defined as follows. An atomic formula is given by a descriptor $(a = v)$, where $a \in At$ and $v \in V_a$. Additional formulas of $\mathcal{L}$ are constructed recursively. If ϕ and ψ are in $\mathcal{L}$, then $\neg(\phi)$, $(\phi \wedge \psi)$, $(\phi \vee \psi)$, $(\phi \rightarrow \psi)$, and $(\phi \leftrightarrow \psi)$ are in $\mathcal{L}$.

In some applications, it is enough to use only a subset of logic operators. For example, one may use the set of operators $\{\wedge, \vee\}$ or $\{\wedge\}$. In general, usage of the decision logic language while formulating knowledge derived from data puts RST based data analysis into a wider category of symbolic machine learning techniques aimed at describing things in a user friendly fashion [1, 25].

One may also consider different forms of descriptors, depending on the types of attributes. For instance, in case of the set-valued and interval-valued attributes (see Sect. 2.1), it might be reasonable to consider inclusion or overlap instead of equality in $(a = v)$. Other examples may include descriptors with inequality operators for attributes with ordered domains of values [26, 27] or descriptors based on degrees of similarity and closeness [18, 28]. In this chapter, we restrict ourselves to the simplest equality based descriptors.

Formulas in $\mathcal{L}$ are interpreted based on the notion of satisfiability. For a formula ϕ, by $x \models \phi$ we denote that the object x satisfies ϕ.

Definition 3 The satisfiability of any formula is defined recursively as follows:

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

Definition 4 If ϕ is a formula, the set $m(\phi)$ defined by:

$$m(\phi) = \{x \in U \mid x \models \phi\}, \tag{2}$$

is called the meaning of the formula ϕ in an information table.

The meaning of a formula ϕ is indeed a set of all objects having the properties expressed by the formula ϕ. This way, a connection between formulas and subsets of U is established. The meanings of formulas can be computed recursively as well, by drawing a correspondence between the logic and set operators.

The decision logic language $\mathcal{L}$ provides a formal description of concepts. A concept in an information table is represented as a pair $(\phi, m(\phi))$, where $\phi \in \mathcal{L}$ and $m(\phi) \subseteq U$. The formula ϕ is a description of $m(\phi)$ in M, i.e., the intension of concept $(\phi, m(\phi))$, and $m(\phi)$ is the set of objects satisfying ϕ, namely, the extension of $(\phi, m(\phi))$. The outcomes of RST based learning processes can be precisely formulated based on formal representation of concepts.

Example 2 Consider the information table given in Table 1. Consider the following formulas: $a = 0$, $a = 1$, $b = 1$, $a = 0 \wedge b = 1$, $a = 0 \vee b = 1$, and $a = 1 \wedge \neg(b = 1)$. Their meaning sets are as follows:

$$
\begin{aligned}
&m(a = 0) = \{o_1, o_2, o_3\},\\
&m(b = 1) = \{o_3, o_6\},\\
&m(a = 0 \wedge b = 1) = m(a = 0) \cap m(b = 1) = \{o_3\},\\
&m(a = 1) = \{o_4, o_5, o_6, o_7, o_8, o_9\},\\
&m(a = 0 \vee b = 1) = m(a = 0) \cup m(b = 1) = \{o_1, o_2, o_3, o_6\},\\
&m(a = 1 \wedge \neg(b = 1)) = m(a = 1) \cap (m(b = 1))^c = \{o_4, o_5, o_7, o_8, o_9\}.
\end{aligned}
$$

They define the following concepts in the information table:

$$
\begin{aligned}
&(a = 0, \{o_1, o_2, o_3\}),\\
&(b = 1, \{o_3, o_6\}),\\
&(a = 0 \wedge b = 1, \{o_3\}),\\
&(a = 1, \{o_4, o_5, o_6, o_7, o_8, o_9\}),\\
&(a = 0 \vee b = 1, \{o_1, o_2, o_3, o_6\}),\\
&(a = 1 \wedge \neg(b = 1), \{o_4, o_5, o_7, o_8, o_9\}).
\end{aligned}
$$

The explicit expression of a concept as a pair of a formula and a set of objects, enables us to analyze an information table in both logic and set-theoretic terms. For example, we can refer to the concept $(a = 0, \{o_1, o_2, o_3\})$ by either the formula $a = 0$ or the set of objects $\{o_1, o_2, o_3\}$. Of course the same set of objects can be obtained using many formulas.

In RST based data analysis, one often considers only a subset of attributes $A \subseteq At$, i.e., only attributes from A are used in forming formulas of the logic language. We use $\mathcal{L}(A)$ to denote the language defined using only attributes from A. All notions introduced so far for $\mathcal{L}$ can be reformulated for $\mathcal{L}(A)$.

2.3 Indiscernibility Relations

Indiscernibility is another fundamental notion or RST. By the indiscernibility of objects, one can granulate the universe into subsets of objects.

Definition 5 For a subset of attributes $A \subseteq At$, we can define an indiscernibility relation $\mathrm{IND}(A)$ as follows:

$$\begin{aligned} x\,\mathrm{IND}(A)\,y \quad &\Longleftrightarrow \quad \forall_{a\in A}(I_a(x) = I_a(y)) \\ &\Longleftrightarrow \quad I_A(x) = I_A(y). \end{aligned} \tag{3}$$

Two objects are indiscernible with respect to a subset of attributes A if they have the same values on every $a \in A$. It can be verified that $\mathrm{IND}(A)$ is reflexive, symmetric, and transitive, namely, $\mathrm{IND}(A)$ is an equivalence relation on U. $\mathrm{IND}(A)$ induces a partition of U, denoted by $U/\mathrm{IND}(A)$ or U/A. Let

$$[x]_{\mathrm{IND}(A)} = [x]_A = \{y \in U \mid x\,\mathrm{IND}(A)\,y\} \tag{4}$$

denote the equivalence class of $\mathrm{IND}(A)$ that contains x. The partition is given by $U/A = U/\mathrm{IND}(A) = \{[x]_A \mid x \in U\}$.

The notion of indiscernibility can be extended in many ways. Extensions may be required to address the need of dealing with non-standard values (see Sect. 2.1) and non-standard descriptors (see Sect. 2.2) that are supposed to correspond to indiscernibility classes. We refer a reader to some tolerance based and fuzzy based generalizations considered in, e.g. [29, 30].

Let us focus on the simplest type of indiscernibility. The set inclusion of equivalence relations defines a partial order on the set of all partitions.

Definition 6 A partial order called refinement-coarsening relation on the set of all partitions of U is defined according to set inclusion of the corresponding equivalence relations as follows: for two relations E and E' on U,

$$U/E \preceq U/E' \quad \Longleftrightarrow \quad E \subseteq E', \tag{5}$$

i.e., each block of U/E' is the union of some blocks of U/E. Thus, U/E is called a refinement of U/E' and U/E' is called a coarsening of U/E.

Different subsets of attributes may define different equivalence relations. Equivalence relations defined by single attributes play an important role, as they can be used to construct equivalence relations defined by any subset of attributes. For a subset of attributes $A \subseteq At$ and $x \in U$, we have:

$$\mathrm{IND}(A) = \bigcap_{a\in A} \mathrm{IND}(\{a\}), \qquad [x]_A = \bigcap_{a\in A} [x]_{\{a\}}. \tag{6}$$

For two subsets of attributes $A, A' \subseteq At$ and $x \in U$, we have:

$$\mathrm{IND}(A \cup A') = \mathrm{IND}(A) \cap \mathrm{IND}(A'), \qquad [x]_{A \cup A'} = [x]_A \cap [x]_{A'}. \tag{7}$$

The partial order $\preceq$ on partitions defined by subsets of attributes is related to set-inclusion of attributes, that is, for $A, A' \subseteq At$ and $x \in U$, we have:

$$A \subseteq A' \implies U/A' \preceq U/A \wedge [x]_{A'} \subseteq [x]_A. \tag{8}$$

That is, the refinement-coarsening relation $\preceq$ is monotonic with respect to set inclusion of subsets of attributes.

Example 3 Consider again Table 1. The partitions induced by subsets of attributes $\{a\}$, $\{b\}$, $\{c\}$, $\{a, b\}$ and $\{a, b, c\}$ are given by:

$$\begin{aligned}
U/\{a\} &= \{\{o_1, o_2, o_3\}, \{o_4, o_5, o_6, o_7, o_8, o_9\}\},\\
U/\{b\} &= \{\{o_1, o_2, o_4, o_5\}, \{o_3, o_6\}, \{o_7, o_8, o_9\}\},\\
U/\{c\} &= \{\{o_1, o_2, o_5, o_7, o_8, o_9\}, \{o_3, o_4\}, \{o_6\}\},\\
U/\{a, b\} &= \{\{o_1, o_2\}, \{o_3\}, \{o_4, o_5\}, \{o_6\}, \{o_7, o_8, o_9\}\},\\
U/\{a, b, c\} &= \{\{o_1, o_2\}, \{o_3\}, \{o_4\}, \{o_5\}, \{o_6\}, \{o_7, o_8, o_9\}\}.
\end{aligned}$$

It can be verified that $U/\{a, b, c\} \preceq U/\{a, b\} \preceq U/\{a\}$. For example, for object o_1, we have:

$$\begin{aligned}
[o_1]_{\{a,b,c\}} &= [o_1]_{\{a\}} \cap [o_1]_{\{b\}} \cap [o_1]_{\{c\}}\\
&= \{o_1, o_2, o_3\} \cap \{o_1, o_2, o_4, o_5\} \cap \{o_1, o_2, o_5, o_7, o_8, o_9\}\\
&= \{o_1, o_2\}.
\end{aligned}$$

2.4 Definable Sets

For a given formula, one can obtain a unique subset of U as its meaning set. In contrast, one may not find a formula that produces a given subset of U. As already mentioned, it may also happen that some subsets of U have multiple representations in $\mathcal{L}$.

Example 4 In Table 1, the subset of objects $\{o_1, o_2, o_3\}$ is the meaning set of several formulas, such as: $a = 0$, $a = 0 \wedge b = 0 \vee a = 0 \wedge b = 1$, and $a = 0 \wedge c = 1 \vee b = 1 \wedge c = 0$. That is, the set of objects $\{o_1, o_2, o_3\}$ has multiple representations in terms of logic formulas. On the other hand, it is impossible to find a formula whose meaning set is $\{o_2, o_3\}$.

Definition 7 A subset $X \subseteq U$ is called a definable set in an information table if there exists a formula ϕ in the logic language $\mathcal{L}$ such that $m(\phi) = X$; otherwise, X is undefinable. A subset $X \subseteq U$ is called a conditionally definable set with respect to a subset of attributes $A \subseteq At$ if there exists ϕ in the logic language $\mathcal{L}(A)$ such that $m(\phi) = X$; otherwise, X is conditionally undefinable.

A definable set may be viewed as a conditionally definable set with respect to the entire set of attributes, i.e., $A = At$. Let

$$\mathrm{DEF}_A(U) = \{m(\phi) \mid \phi \in \mathcal{L}(A)\} \tag{9}$$

denote the set of all definable sets with respect to attributes $A \subseteq At$. It is important to study the structure of $\mathrm{DEF}_A(U)$. Consider the empty set $\emptyset$. By the definition of an information table, we have $m(a = v \wedge \neg(a = v)) = \emptyset$. Hence $\emptyset \in \mathrm{DEF}_A(U)$. Consider the equivalence class $[x]_A$, $x \in U$. We have $m(\bigwedge_{a \in A} a = I_a(x)) = [x]_A$. Thus, $[x]_A \in \mathrm{DEF}_A(U)$. In fact, $[x]_A$ is a minimal nonempty definable set in $\mathrm{DEF}_A(U)$. $\mathrm{DEF}_A(U)$ is also closed under set complement, intersection and union. In summary, $(\mathrm{DEF}_A(U), ^c, \cap, \cup, \emptyset, U)$ is an atomic Boolean algebra with the minimum element $\emptyset$, the maximum element U, and the set of atoms corresponding to U/A.

One may say that a decision logic language and the indiscernibility relation provide two ways for characterizing subsets of a universe, through definable sets in the former and indiscernibility classes in the latter. The families of all definable sets and all indiscernibility classes form the basic ingredients of RST based data analysis. Let us note that definable sets can be equivalently expressed as unions of some subsets of U/A, that is:

$$\mathrm{DEF}_A(U) = \left\{\bigcup F \mid F \subseteq U/A\right\}. \tag{10}$$

It may be useful to operate exchangeably with formulations (9) and (10) while considering both foundations and applications of RST.

Example 5 Consider the set of attributes $A = \{a, b\}$ in Table 1. The atoms of the Boolean algebra $(\mathrm{DEF}_A(U), ^c, \cap, \cup, \emptyset, U)$ are as follows:

$$U/\{a, b\} = \{\{o_1, o_2\}, \{o_3\}, \{o_4, o_5\}, \{o_6\}, \{o_7, o_8, o_9\}\}.$$

Equivalence class $[o_1]_A = [o_2]_A = \{o_1, o_2\}$ is defined by a formula $a = 0 \wedge b = 0$, equivalence class $[o_3]_A = \{o_3\}$ is defined by a formula $a = 0 \wedge b = 1$, equivalence class $[o_4]_A = [o_5]_A = \{o_4, o_5\}$ is defined by a formula $a = 1 \wedge b = 0$, and so on. The

family of all definable sets takes the following form:

$$
\begin{aligned}
\mathrm{DEF}_A(U) = \{&\emptyset,\\
\text{// 1 atom}\quad &\{o_1, o_2\}, \{o_3\}, \{o_4, o_5\}, \{o_6\}, \{o_7, o_8, o_9\},\\
\text{// 2 atoms}\quad &\{o_1, o_2, o_3\}, \{o_1, o_2, o_4, o_5\}, \{o_1, o_2, o_6\}, \{o_1, o_2, o_7, o_8, o_9\},\\
&\{o_3, o_4, o_5\}, \{o_3, o_6\}, \{o_3, o_7, o_8, o_9\}, \{o_4, o_5, o_6\},\\
&\{o_4, o_5, o_7, o_8, o_9\}, \{o_6, o_7, o_8, o_9\},\\
\text{// 3 atoms}\quad &\{o_1, o_2, o_3, o_4, o_5\}, \{o_1, o_2, o_3, o_6\}, \{o_1, o_2, o_3, o_7, o_8, o_9\},\\
&\{o_1, o_2, o_4, o_5, o_6\}, \{o_1, o_2, o_4, o_5, o_7, o_8, o_9\},\\
&\{o_1, o_2, o_6, o_7, o_8, o_9\}, \{o_3, o_4, o_5, o_6\}, \{o_3, o_4, o_5, o_7, o_8, o_9\},\\
&\{o_3, o_6, o_7, o_8, o_9\}, \{o_4, o_5, o_6, o_7, o_8, o_9\},\\
\text{// 4 atoms}\quad &\{o_1, o_2, o_3, o_4, o_5, o_6\}, \{o_1, o_2, o_3, o_4, o_5, o_7, o_8, o_9\},\\
&\{o_1, o_2, o_3, o_6, o_7, o_8, o_9\}, \{o_1, o_2, o_4, o_5, o_6, o_7, o_8, o_9\},\\
&\{o_3, o_4, o_5, o_6, o_7, o_8, o_9\},\\
\text{// 5 atoms}\quad &U\}.
\end{aligned}
$$

Each definable set can be expressed as a union of some equivalence classes. For example, $\{o_1, o_2, o_3, o_4, o_5\} = \{o_1, o_2\} \cup \{o_3\} \cup \{o_4, o_5\}$ is described by a formula $(a = 0 \wedge b = 1) \vee (a = 0 \wedge b = 1) \vee (a = 1 \wedge b = 0)$, which is a disjunction of formulas defining three equivalence classes $\{o_1, o_2\}$, $\{o_3\}$ and $\{o_4, o_5\}$. In contrast, an undefinable set cannot be expressed this way.

3 Rough Set Approximations

Approximations of sets are the fundamental construct that distinguishes RST from other approaches. They are very important for applications developed within the standard RST based data analysis [31], as well as for other examples of employing RST, such as rough clustering [32], granular database engines [33] and complex pattern learning based on domain knowledge [15].

3.1 Approximations of a Single Set

Each definable set can be represented by a logic formula and hence we can make inference about definable sets. On the other hand, we cannot find a formula to represent an undefinable set. In order to make inference about undefinable sets, RST considers an approximation of an undefinable set by definable sets. More specifically, one can approximate the undefinable set from the below and the above by using two definable sets. By properties of the set of all definable sets, such approximations are unique.

With respect to a subset of attributes $A \subseteq At$, one can either define a logic language or an equivalence relation. As already mentioned, both of them produce the same definable sets in an information table. Thus, it is reasonable to apply atomic definable sets corresponding to the elements of U/A to approximate

other sets. This idea can be formalized using the notion of an approximation space $apr_A = (U, \text{IND}(A))$ (or $apr_A = (U, \text{DEF}_A(U))$) [1, 2].

Below, if a subset of attributes A is understood, we may drop it by simply writing $apr = (U, \text{IND})$ or $apr = (U, \text{DEF})$.

Definition 8 In an approximation space $apr = (U, \text{IND})$, a pair of lower and upper approximations of a subset $X \subseteq U$ is defined by

$$\begin{aligned} \underline{apr}(X) &= \text{the largest definable set in DEF}(U) \text{ that is contained by } X, \\ \overline{apr}(X) &= \text{the smallest definable set in DEF}(U) \text{ that contains } X. \end{aligned} \tag{11}$$

Example 6 Consider Table 1. The set of decision attribute $D = \{d\}$ produces a partition $U/D = \{X_1 = m(d = +) = \{o_1, o_2, o_4, o_5, o_6\}, X_2 = m(d = -) = \{o_3, o_7, o_8, o_9\}\}$. The set of attributes $\{a, b\}$ gives rise to an approximation space $apr_{\{a,c\}}$. The approximations of the two sets X_1 and X_2 are given by:

$$\begin{aligned} &\underline{apr}_{\{a,c\}}(X_1) = \{o_1, o_2, o_4, o_6\}, \qquad \underline{apr}_{\{a,c\}}(X_2) = \{o_3\}, \\ &\overline{apr}_{\{a,c\}}(X_1) = \{o_1, o_2, o_4, o_5, o_6, o_7, o_8, o_9\}, \\ &\overline{apr}_{\{a,c\}}(X_2) = \{o_3, o_5, o_7, o_8, o_9\}. \end{aligned}$$

Lower and upper approximations may be expressed also in other forms [34], which are convenient when seeking for analogies between RST and other approaches to data analysis and knowledge representation.

Definition 9 In an approximation space $apr = (U, \text{IND})$, approximations can be expressed in one of the following three equivalent ways:

– Element based definition

$$\begin{aligned} \underline{apr}(X) &= \{x \mid x \in U, [x]_{\text{IND}} \subseteq X\} \\ &= \{x \mid x \in U, \forall y \in U (x \,\text{IND}\, y \Longrightarrow y \in X)\}, \\ \overline{apr}(X) &= \{x \mid x \in U, [x]_{\text{IND}} \cap X \neq \emptyset\} \\ &= \{x \mid x \in U, \exists y \in U (x \,\text{IND}\, y, y \in X)\}; \end{aligned} \tag{12}$$

– Granule based definition

$$\begin{aligned} \underline{apr}(X) &= \bigcup \{[x]_{\text{IND}} \mid [x]_{\text{IND}} \in U/\text{IND}, [x]_{\text{IND}} \subseteq X\}, \\ \overline{apr}(X) &= \bigcup \{[x]_{\text{IND}} \mid [x]_{\text{IND}} \in U/\text{IND}, [x]_{\text{IND}} \cap A \neq \emptyset\}; \end{aligned} \tag{13}$$

– Subsystem based definition

$$\begin{aligned} \underline{apr}(X) &= \bigcup \{Y \mid Y \in \text{DEF}(U), Y \subseteq X\}, \\ \overline{apr}(X) &= \bigcap \{Y \mid X \in \text{DEF}(U), X \subseteq Y\}. \end{aligned} \tag{14}$$

It is convenient to operate with the above three formalizations exchangeably when applying and extending the original notions of RST. For instance, different definitions of approximations provide different means for handling information tables with missing values [35]. For the case of complete information tables, lower and upper approximations satisfy the following properties:

(L0)	$\underline{apr}(X) \in \mathrm{DEF}(U)$	(U0)	$\overline{apr}(X) \in \mathrm{DEF}(U)$
(L1)	$X \in \mathrm{DEF}(U) \Longrightarrow \underline{apr}(X) = X$	(U1)	$X \in \mathrm{DEF}(U) \Longrightarrow \overline{apr}(X) = X$
(L2)	$\underline{apr}(X) \subseteq X$	(U2)	$X \subseteq \overline{apr}(X)$
(L3)	$\underline{apr}(X) = (\overline{apr}(X^c))^c$	(U3)	$\overline{apr}(X) = (\underline{apr}(X^c))^c$
(L4)	$\underline{apr}(X \cap Y) = \underline{apr}(X) \cap \underline{apr}(Y)$	(U4)	$\overline{apr}(X \cup Y) = \overline{apr}(X) \cup \overline{apr}(Y)$
(L5)	$\underline{apr}(X \cup Y) \supseteq \underline{apr}(X) \cup \underline{apr}(Y)$	(U5)	$\overline{apr}(X \cap Y) \subseteq \overline{apr}(X) \cap \overline{apr}(Y)$
(L6)	$X \subseteq Y \Longrightarrow \underline{apr}(X) \subseteq \underline{apr}(Y)$	(U6)	$X \subseteq Y \Longrightarrow \overline{apr}(X) \subseteq \overline{apr}(Y)$
(L7)	$\underline{apr}(X) = \underline{apr}(\underline{apr}(X))$	(U7)	$\overline{apr}(\overline{apr}(X)) = \overline{apr}(X)$
(L8)	$\overline{apr}(X) = \underline{apr}(\overline{apr}(X))$	(U8)	$\overline{apr}(\underline{apr}(X)) = \underline{apr}(X)$

Properties (L0) and (U0) state that approximations of a set are definable sets. They imply properties (L1) and (U1), namely, the approximations of a definable set are the set itself. In general, according to Properties (L2) and (U2), a set falls within its lower and upper approximations, namely, $\underline{apr}(X) \subseteq X \subseteq \overline{apr}(X)$. Properties (L3) and (U3) state that lower and upper approximations are a pair of dual operators $\underline{apr}, \overline{apr} : 2^U \longrightarrow 2^U$ [36]. Hence, properties labeled by the same number may be interpreted as dual ones. The remaining properties formalize other important aspects of RST such as, for instance, monotonicity of approximation operators with respect to set inclusion or stability of outcomes of chains of successive approximations.

3.2 Rough Set Regions for a Single Set

Rough set approximations are often rephrased in terms of positive, negative and boundary regions, which gather objects (or rather some classes or granules of objects) that, respectively, certainly satisfy, certainly do not satisfy, and maybe (do not) satisfy the concepts represented by subsets of universe. Such regions are very useful in RST based decision making [37] and many other rough set applications, such as already mentioned granular database architecture [33], where, in a sense, only boundary related blocks of data need to be accessed to execute some common types of analytic SQL statements.

Definition 10 Based on lower and upper approximations, one can divide the universe U into the following positive, boundary and negative regions:

$$\begin{aligned} \mathrm{POS}(X) &= \underline{apr}(X), \\ \mathrm{BND}(X) &= \overline{apr}(X) - \underline{apr}(X), \\ \mathrm{NEG}(X) &= (\overline{apr}(X))^c. \end{aligned} \tag{15}$$

It is worth remembering that the above representations may not be equivalent any longer for some extensions of standard RST notions briefly mentioned in Sect. 2. However, in both of above cases, $\{\mathrm{POS}(X), \mathrm{BND}(X), \mathrm{NEG}(X)\}$ forms a partition of the universe U.

Example 7 Let us continue Example 6. For the set of decision attributes $D = \{d\}$ and its corresponding partition classes X_1 and X_2, the set of attributes $\{a, b\}$ induces the following regions:

$$\begin{aligned} &\mathrm{POS}_{\{a,c\}}(X_1) = \{o_1, o_2, o_4, o_6\}, && \mathrm{POS}_{\{a,c\}}(X_2) = \{o_3\}, \\ &\mathrm{BND}_{\{a,c\}}(X_1) = \{o_5, o_7, o_8, o_9\}, && \mathrm{BND}_{\{a,c\}}(X_2) = \{o_5, o_7, o_8, o_9\}, \\ &\mathrm{NEG}_{\{a,c\}}(X_1) = \{o_3\}, && \mathrm{NEG}_{\{a,c\}}(X_2) = \{o_1, o_2, o_4, o_6\}. \end{aligned}$$

One can notice a kind of duality of regions in the case of two sets that are complementary to each other. The situation gets more complicated when the number of classes increases. In such cases one may consider using, for example, generalized decision functions introduced into RST for the purpose of dealing simultaneously with larger collections of sets to be approximated [3].

The above-introduced regions can be employed to build the following representations of the approximated sets, which—although mathematically equivalent—support different intuitions of reasoning about data:

(i) $(\mathrm{POS}(X), \mathrm{BND}(X), \mathrm{NEG}(X))$
(ii) $(\mathrm{POS}(X), \mathrm{POS}(X) \cup \mathrm{BND}(X))$
(iii) $(\mathrm{POS}(X), \mathrm{BND}(X))$
(iv) $(\mathrm{POS}(X), \mathrm{NEG}(X))$

Representation (ii) is the pair of rough set approximations. Representations (iii) and (iv) emphasize the roles of boundary and negative regions, respectively. It is also worth mentioning about some useful extensions of regions and approximations based on combination of standard rough set methodology with, for instance, probability calculus [38, 39] and fuzzy sets [30, 40].

3.3 Approximations and Regions of a Partition

The approximation of a set can be easily extended to the approximation of a partition, also called a classification [2].

Definition 11 Let $\pi = \{X_1, \ldots, X_n\}$ be a partition of the universe U. Its approximations can be defined as the families of approximations of particular partition classes:

$$\begin{aligned} \underline{apr}(\pi) &= \{\underline{apr}(X_1), \ldots, \underline{apr}(X_n)\}, \\ \overline{apr}(\pi) &= \{\overline{apr}(X_1), \ldots, \overline{apr}(X_n)\}. \end{aligned} \tag{16}$$