Toward Rough Knowledge Bases with Quantitative Measures

Aida Vitória[1], Carlos Viegas Damásio[2], and Jan Małuszyński[3]

[1] Dept. of Science and Technology, Linköping University,
S 601 74 Norrköping, Sweden
aidvi@itn.liu.se

[2] Centro de Inteligência Artificial (CENTRIA),
Dept. Informática, Faculdade de Ciências e Tecnologia,
Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
cd@di.fct.unl.pt

[3] Dept. of Computer and Information Science,
Linköping University, S 581 83 Linköping, Sweden
janma@ida.liu.se

Abstract. We present a language for defining new rough relations from given decision tables and we show how to query relations defined in this way. The language provides a uniform formalism for expressing rough data together with background knowledge, and for capturing well-known techniques such as the variable precision rough set model. Its essential feature is the use of quantitative measures, such as support, strength and accuracy.

1 Introduction

Existing rough set techniques [1] do not provide natural support for incorporation of background knowledge. Addressing this problem, we have recently proposed a language [2, 3] that caters for implicit definition of rough sets from given ones. Our language seems to address similar problems as the system **CAKE** [4]. For comparison and discussion of some important differences see [3].

This paper extends our language of [2, 3] with quantitative measures, such as support, strength, accuracy, and coverage, which play an essential role in applications. The extension is very expressive since new rough sets can now be defined using conditions with quantitative measures. For example, we show that the variable precision rough set model [5] can be naturally expressed in this language. Another important add-on of the language is that clauses defining new rough sets are parameterized by weighted functions combining the supports of indiscernibility classes of given rough sets into the supports of the defined ones. We claim that this parameterization allows to define rough sets better tailored to specific applications. The rough sets defined in our language and their related quantitative measures can be queried, as discussed in Section 3. The practical application of Section 4 illustrates the usefulness of the language.

For space reasons, a formal declarative semantics and the operational semantics of the language are not discussed in this paper.

2 Rough sets

We extend the notion of rough set used in our previous work [2, 3] to take into account quantitative measures. Individuals of a universe of discourse U are in many cases classified on the basis of some associated information, such as values of certain chosen attributes. Formally, an attribute a is a total function $a : U \to V_a$. Every object is associated with a tuple of attributes and this tuple is the only way of referring to the object. Hence, different individuals with same attribute values are indiscernible.

Definition 1. *Given a set of attributes $A = \{a_1, \ldots, a_n\}$, a* rough set *(or* rough relation*) S is a pair of sets $(\overline{S}, \overline{\neg S})$ satisfying conditions (i) and (ii).*

(i) The elements of sets $\overline{S}$ and $\overline{\neg S}$ are expressions of the form $\langle t_1, \ldots, t_n\rangle : k$, where $\langle t_1, \ldots, t_n\rangle \in \prod_{a_i \in A} V_{a_i}$ and k is an integer larger than zero.

(ii) $(\langle t_1, \ldots, t_n\rangle : k \in \overline{S} \Rightarrow \forall k' \neq k(\langle t_1, \ldots, t_n\rangle : k' \notin \overline{S}))$ and $(\langle t_1, \ldots, t_n\rangle : k \in \overline{\neg S} \Rightarrow \forall k' \neq k(\langle t_1, \ldots, t_n\rangle : k' \notin \overline{\neg S}))$.

The rough complement *of a rough set $S = (\overline{S}, \overline{\neg S})$ is the rough set $\neg S = (\overline{\neg S}, \overline{S})$.*

For simplicity, we write t to designate a general tuple $\langle t_1, \ldots, t_n\rangle$ and $[t]$ represents the indiscernibility class described by tuple t. Intuitively, an element $t : k \in \overline{S}$ ($t : k \in \overline{\neg S}$) indicates that the indiscernibility class $[t]$ belongs to the upper approximation of rough set S ($\neg S$) and that this class contains $k > 0$ individuals that are positive examples of the concept described by S ($\neg S$). *Lower approximation* of rough set S, represented by $\underline{S}$, is defined as $\underline{S} = \{t : k_1 \in \overline{S} \mid \forall k_2 > 0\,(t : k_2 \notin \overline{\neg S})\}$. The *boundary region*, represented by $\underline{\overline{S}}$, is defined as $\underline{\overline{S}} = \{t : k_1 : k_2 \mid \exists k_1, k_2 > 0\ (t : k_1 \in \overline{S} \text{ and } t : k_2 \in \overline{\neg S})\}$.

Obviously, a decision table can be seen as an alternative representation of a rough set $D = (\overline{D}, \overline{\neg D})$. The fact that we consider only binary decision attributes is not a restriction in practice, as shown in [3] (see example 4.4).

Quantitative Measures Let a tuple t be the description of an indiscernibility class $[t]$ of a decision table $\mathcal{D} = (U, A, \mathtt{d})$. Assume also that $|d|$ ($|\neg d|$) is the number of individuals (or lines of the table) that have positive (negative) outcome for the decision attribute $\mathtt{d}$. The following quantitative measures are used.

- The *support* of $d(t)$, represented by $\mathtt{supp}(d(t))$, corresponds to the number of individuals in the indiscernibility class $[t]$ that are positive examples. Thus, if $t : k \in \overline{D}$ then $\mathtt{supp}(d(t)) = k$.
- The *strength* of $d(t)$, represented by $\mathtt{strength}(d(t))$, indicates how often individuals in the indiscernibility class $[t]$ have positive outcome for the decision attribute $\mathtt{d}$. Thus, if $t : k \in \overline{D}$ then $\mathtt{strength}(d(t)) = \frac{k}{|d|+|\neg d|}$.
- The *accuracy* of $d(t)$, represented by $\mathtt{acc}(d(t))$, corresponds to the conditional probability $Pr(d(i) = \mathtt{yes} \mid i \in [t])$. Thus, if $t : k_1 \in \overline{D}$ and $t : k_2 \in \overline{\neg D}$ then, $\mathtt{acc}(d(t)) = \frac{k_1}{k_1+k_2}$.

These definitions extend naturally to $\neg d(t)$. The implementation of the language will allow introduction of other quantitative measures in addition to those discussed above.

3 A Language for Defining Rough Relations

In this section, we extend the language for defining rough relations, presented in [2, 3], with quantitative measures. We restrict the definition of new rough relations to be non recursive.

A *quantitative measure expression* is a condition involving quantitative measures. For instance, $\mathtt{acc}(p(c_1, c_2)) > \mathtt{acc}(\neg q(c_1, c_2))$ is a quantitative measure expression stating that the accuracy of the indiscernibility class $[\langle c_1, c_2\rangle]$ of rough relation P is larger than the accuracy of the indiscernibility class $[\langle c_1, c_2\rangle]$ of rough relation $\neg Q$.

A *rough program* is a set of *rough facts* and *rough clauses*. Rough facts encode rough relations defined explicitly by a decision table, while rough clauses are used to define implicitly new rough relations obtained by combining different regions (e.g. lower approximation, upper approximation, and boundary) of other rough relations. Rough facts are expressions of either the form $\overline{r}(c_1, \ldots, c_n) : k_1.$ or $\overline{\neg r}(c_1, \ldots, c_n) : k_2.$, where each c_i is a constant. These facts state that $\langle c_1, \ldots, c_n\rangle : k_1 \in \overline{R}$ and $\langle c_1, \ldots, c_n\rangle : k_2 \in \overline{\neg R}$, respectively, where R is the rough relation denoted by r. A *rough literal* is an expression of either the form $\overline{r}(t_1, \ldots, t_n)$, or $\underline{r}(t_1, \ldots, t_n)$, or $\underline{\overline{r}}(t_1, \ldots, t_n)$, or $\overline{\neg r}(t_1, \ldots, t_n)$, or $\underline{\neg r}(t_1, \ldots, t_n)$, or $\underline{\overline{\neg r}}(t_1, \ldots, t_n)$. A rough clause has the form *head* $\mathtt{:-}[\alpha, F]$ *body.* where the *head* is a rough literal not referring to the boundary region and the *body* is a sequence (representing conjunction) of rough literals and quantitative measure expressions. Moreover, α should be a rational number between 0 and 1 and F should be an associative and commutative binary function.

An example of a rough clause is $\overline{p}(X_1, X_2)\ \mathtt{:-}[1, \mathtt{min}]\ \underline{q}(X_1, X_2), \overline{\neg r}(X_1, X_2).$, where symbols like p, q, and r denote some rough relations P, Q, and R, respectively. The expression $\underline{q}(X_1, X_2)$ can be seen as representing an indiscernibility class belonging to the lower approximation of rough relation Q. Note that X_1 and X_2 are variables that can be thought as representing any attribute value. Informally, this rough clause states that, given a tuple $\langle c_1, c_2\rangle$ describing an indiscernibility class, if $\langle c_1, c_2\rangle : k_2 \in \underline{Q}$ and $\langle c_1, c_2\rangle : k_3 \in \overline{\neg R}$ then $\langle c_1, c_2\rangle : k_1 \in \overline{P}$, where $\mathtt{supp}(p(c_1, c_2)) = k_1 \geq \lfloor 1 \times \mathtt{min}(k_2, k_3)\rfloor$. Hence, function $F = \mathtt{min}$ is used to combine $\mathtt{supp}(q(c_1, c_2))$ with the $\mathtt{supp}(\neg r(c_1, c_2))$. Note that the support k_1 should be computed by taking into account all clauses of a rough program.

In a rough clause, the user-defined parameters α and F make it possible to tune the definition to a particular application and data. For example, a rough clause like $\underline{p}(X, c)\mathtt{:-}[0.8, \_]\ \underline{q}(X, c).$ could be used if the user strongly doubts of the reliability of the information carried by 20% of the examples, belonging to any indiscernibility class only with positive examples of Q and for which the second attribute has value c. To represent that the choice of function F is irrelevant, we use the symbol '_'.

Also the problem of how to combine the support information, obtained from the expressions in the body strongly depends on the application. For instance, in a clause describing join of two decision tables, F should be the multiplication. In other applications, different functions (e.g. the minimum) might be more appropriate.

Variable Precision Rough Relations We discuss next how quantitative measure expressions in the body of rough clauses can be used to build more generalized rough approximations of a relation, in the same spirit of the *variable precision rough set model* (VPRSM) [5]. This section also illustrates an important application of our language, since the VPRSM is the rough set model most used in practical applications.

Let $Pr(q)$ be the prior probability of an individual to be a positive example of rough relation Q denoted by a predicate q. Thus, $Pr(q) = \frac{|q|}{|q|+|\neg q|}$. Moreover, assume that l and u are two *precision control parameters* such that $0 \leq l < Pr(q) < u \leq 1$. Generalization of lower (upper) approximation and boundary region of a rough relation can be obtained as follows. The lower approximation of Q ($\neg Q$)is obtained from those indiscernibility classes $[t]$ such that $\mathtt{acc}(q(t)) \geq u$ ($\mathtt{acc}(q(t)) \leq l$). Those indiscernibility classes $[t]$ such that $l < \mathtt{acc}(q(t)) < u$ are considered to belong to the boundary region $\overline{\underline{Q}}$. This technique can also be seen as a way to "thin" the boundary region and make the approximations obtained less sensitive to possible noise contained in the data.

Example 1. Let q denote a rough relation Q (possibly obtained directly from a decision table and encoded as a set of facts in our language), l and u be two precision control parameters. We define then a new rough relation Q_1 according to the ideas discussed above. Note that the use of $\overline{q}(X_1, X_2), \overline{\neg q}(X_1, X_2)$ in the body of the third and fourth rough clauses captures those indiscernibility classes $[t]$ in the boundary of Q. Moreover, it is worth to note the use of function `sum` to combine $\mathtt{supp}(q(t))$ with $\mathtt{supp}(\neg q(t))$, since $\mathtt{supp}(q(t)) + \mathtt{supp}(\neg q(t))$ gives the total number of individuals in the indiscernibility class $[t]$.

$\overline{q_1}(X_1, X_2)$ `:-`$[1, \_]$ $\underline{q}(X_1, X_2)$.
$\overline{\neg q_1}(X_1, X_2)$ `:-`$[1, \_]$ $\underline{\neg q}(X_1, X_2)$.
% `Any indiscernibility class t in the boundary s.t.` $acc(q(t)) \geq u$
% `is considered to be in` $[t] \in \underline{Q}$
$\underline{q_1}(X_1, X_2)$ `:-`$[1, \mathtt{sum}]$ $\overline{q}(X_1, X_2), \overline{\neg q}(X_1, X_2), \mathtt{acc}(q(X_1, X_2)) \geq u$.
% `Any indiscernibility class t in the boundary s.t.` $acc(q(t)) \leq l$
% `is considered to be in` $\underline{\neg Q}$
$\underline{\neg q_1}(X_1, X_2)$ `:-`$[1, \mathtt{sum}]$ $\overline{q}(X_1, X_2), \overline{\neg q}(X_1, X_2), \mathtt{acc}(q(X_1, X_2)) \leq l$.
% `Any indiscernibility class t in the boundary s.t.` $l < acc(q(t)) < u$
% `remains in the boundary`
$\overline{q_1}(X_1, X_2)$ `:-`$[1, \_]$ $\overline{q}(X_1, X_2), \mathtt{acc}(q(X_1, X_2)) > l, \mathtt{acc}(q(X_1, X_2)) < u$.
$\overline{\neg q_1}(X_1, X_2)$ `:-`$[1, \_]$ $\overline{\neg q}(X_1, X_2), \mathtt{acc}(q(X_1, X_2)) > l, \mathtt{acc}(q(X_1, X_2)) < u$.

Query Language is designed to identify subsets of the rough relations described by a given program and to classify new individuals. Due to the space limitations we can only provide an informal idea. A basic query has the form of rough literal. The answer identifies a subset of the respective rough relation. A query of the form $K = m(r(t))$ binds the variable K with the quantitative measure m of the indiscernibility class $[t]$ of rough relation R. A sequence of the above discussed queries is interpreted as conjunction. For instance, the rough query $(\overline{p}(X_1, b), K = \texttt{acc}(p(X_1, b))$ **,** $\mathcal{P})$ requests the description of all indiscernibility classes in the upper approximation of P such that the second attribute has value b. Moreover, it is also requests information about the accuracy of each of these indiscernibility classes.

We turn now our attention to classifier queries. These queries are of the form $(\texttt{classify}(d(t))$ **,** $\mathcal{P})$. For example, each tuple $t = \langle c_1, c_2, c_3\rangle$ describing an indiscernibility class of a rough relation D can be seen as a decision rule. Assume that rough relation D corresponds (implicitly) to the decision table $\mathcal{D} = (U, \{a_1, a_2, a_3\}, \texttt{d})$. If $t \in \overline{D}$ then it induces the decision rule $(a_1 = c_1 \wedge a_2 = c_2 \wedge a_3 = c_3) \rightarrow \texttt{d} = \texttt{yes}$. If $t \in \overline{\neg D}$ then it induces the decision rule $(a_1 = c_1 \wedge a_2 = c_2 \wedge a_3 = c_3) \rightarrow \texttt{d} = \texttt{no}$.

The query $(\texttt{classify}(d(c_1, X, c_3)), \mathcal{P})$ requests a prediction for the decision class to which a new individual i described by $(a_1 = c_1 \wedge a_3 = c_3)$ may belong. To answer this query a standard strategy is followed: each rule that match the description of i casts a number of votes corresponding to its support and the decision class is decided by majority of votes. The answer to the rough query is either the pair $(\texttt{d} = \texttt{yes}, C_F)$, or $(\texttt{d} = \texttt{no}, C_F)$, or $(\texttt{d} = \texttt{unknown}, 0)$, where C_F represents a certainty factor. The last case corresponds to the situation where no decision rule is fired.

4 An Application Example

Given a rough relation representing implicity a decision table $\mathcal{D}_A = (U, A, \texttt{d})$, it may be useful to evaluate how removal of some attributes from A affects the ability of making a decision. In other words, given a subset of attributes $B \subset A$, we may wish to identify those individuals i for which the knowledge about attributes B is absolutely necessary for making a decision, i.e. for determining $d(i)$. A similar problem has been studied in [6]. The set of those individuals i can be defined as $M = (\overline{\underline{D}}_{A\setminus B} \cap \underline{D}_A) \cup (\overline{\underline{D}}_{A\setminus B} \cap \underline{\neg D}_A)$. Although this definition looks different from the one used in [6], they are both equivalent. However, the formulation presented here is more suitable in the context of our framework. The set of individuals for which knowledge about attributes B is not strictly needed can then be defined as $\neg M = \underline{\neg D}_{A\setminus B} \cup \underline{D}_{A\setminus B} \cup \overline{\underline{D}}_A$.

Example 2. Consider a decision table $\mathcal{H}\text{p} = (U, \{\texttt{Age}, \texttt{A}_1, \texttt{A}_2\}, \texttt{Hp})$ encoded as a set of rough facts, where U is a set of patients examined for heart problems (decision attribute Hp). Assume that the conditional attributes $\texttt{A}_1$ and $\texttt{A}_2$ represent two medical tests and that $\texttt{A}_2$ is usually considered as being desirable to avoid. The following rough clauses monitor the impact in the boundary region of Hp of not considering test $\texttt{A}_2$. Rough clauses (3) and (4) encode the set M. Rough clauses (5), (6), and (7) capture the set $\neg M$.

(1) $\overline{\texttt{d}}\texttt{(Age, A1) :-[1,\_]}\ \overline{\texttt{hp}}\texttt{(Age, A1, A2).}$
(2) $\overline{\neg\texttt{d}}\texttt{(Age, A1) :-[1,\_]}\ \overline{\neg\texttt{hp}}\texttt{(Age, A1, A2).}$
(3) $\overline{\texttt{m}}\texttt{(Age, A1) :-[1,min]}\ \overline{\underline{\texttt{d}}}\texttt{(Age, A1)},\ \underline{\texttt{hp}}\texttt{(Age, A1, A2).}$

(4) $\overline{\texttt{m}}$(Age, A1) :-[1,min] $\underline{\texttt{d}}$(Age, A1), $\underline{\neg\texttt{hp}}$(Age, A1, A2).
(5) $\overline{\neg\texttt{m}}$(Age, A1) :-[1,_] $\underline{\neg\texttt{d}}$(Age, A1).
(6) $\overline{\neg\texttt{m}}$(Age, A1) :-[1,_] $\underline{\texttt{d}}$(Age, A1).
(7) $\overline{\neg\texttt{m}}$(Age, A1) :-[1,sum] $\overline{\texttt{hp}}$(Age, A1, A2), $\overline{\neg\texttt{hp}}$(Age, A1, A2).

We give now an example of a useful query. Assume that rough program $\mathcal{P}$ contains all facts encoding the decision table and the rough clauses (1) – (7).

- *"For which patients it may be useful to request the expensive test A_2? And what is the expected gain if only those patients undergo test A_2?"*
  Rough query: $(\overline{\texttt{m}}(Age, A1), K_1 = \texttt{strength}(\texttt{m}(Age, A1)),$
  $K_2 = \texttt{strength}(\neg\texttt{m}(Age, A1))$, $\mathcal{P})$.
  Notice that $\sum_{[t]\in\overline{M}}(\texttt{strength}(\texttt{m}(t)) + \texttt{strength}(\neg\texttt{m}(t)))$ gives an indication of the expected gain.

5 Future Work

The implementation based on compilation of rough programs into Prolog is under way. It will be made available through a Web page and will be accompanied by an extended report. We are studying the possibility of integrating it with the **Rosetta** system [7]. In this way, we could use the algorithms already available in **Rosetta** to compute reducts. On the other hand, **Rosetta** could profit from the fact that the VPRSM is easily definable in our language.

As future work, we plan to support several classification algorithms and other quantitative measures. One possibility to achieve this is by a predefined library of classification algorithms and quantitative measures. Another possibility is to extend the language with primitives for specification of classification algorithms and quantitative measures.

References

1. Pawlak, Z.: Rough sets. International Journal of Information and Computer Science **11** (1982) 341–356
2. Vitória, A., Damásio, C.V., Małuszyński, J.: Query answering for rough knowledge bases. In Wang, G., Liu, Q., Yao, Y., Skowron, A., eds.: Proc. of the 9th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, RSFDGrC'03. Volume 2639 of LNCS/LNAI., (Springer) 197–204
3. Vitória, A., Damásio, C.V., Małuszyński, J.: From rough sets to rough knowledge bases. Fundamenta Informaticae **57** (2003) 215–246
4. Doherty, P., Lukaszewicz, W., Szalas, A.: CAKE: A Computer Aided Knowledge Engineering Technique. In: Proc. of the 15th European Conference on Artificial Intelligence, (ECAI'02), Amsterdam, IOS Press (2002) 220–224
5. Ziarko, W.: Variable precision rough set model. Journal of Computer and Systems Science **46** (1993) 39–59
6. Komorowski, J., Øhrn, A.: Modelling prognostic power of cardiac tests using rough sets. Journal of Artificial Intelligence in Medicine **15** (1999) 167–191
7. Øhrn, A., Komorowski, J.: ROSETTA: A rough set toolkit for analysis of data. In: Proc. of Fifth International Workshop on Rough Sets and Soft Computing (RSSC'97). Volume 3. (1997) 403–407