TEMPORAL DYNAMICS IN INFORMATION TABLES

D. CIUCCI

Abstract. An information table can change over time in several different ways: objects enter/exit the system, new attributes are considered, etc. As a consequence rough set instruments also change. At first, we recall a classification of dynamic increase of information with respect to three different factors: objects, attributes, values. Then, the corresponding changes in rough sets are discussed. Results about approximations, positive region and generalized decision are given and algorithms to update reducts and rules provided.

1. Introduction

The interest in dynamics analysis in rough sets naturally stems in the fact that knowledge evolves in time and, thus, rough-set techniques are influenced by dynamics. Indeed, since the very beginning, attempts to deal with this issue have been carried out [Orl82].

Generally speaking, evolution in time can have different nuances: change in the information already available, addition of new information or also elimination of some information. In our classification [Ciu10a] we give particular attention to the increase of information (and by duality decrease), and we suppose that the acquired information cannot change at a following time. On the contrary, this problem is tackled in [ZLCJ09, CLQR10, CLZ10a] where the update of approximations is studied in the case of coarsening and refining of values.

When considering Information Tables, we can have an increase of information with respect to objects, attributes and values. Increase of information with respect to objects has been studied in [SZ95, Woj01], where an algorithm to incrementally update the reducts has been proposed. Further, in [SPS05] objects observed at different times are collected in a unique *temporal information system*, which is then studied, for instance with regards to changes of functional dependencies between attributes. Other ways to update reducts and rules without recompute them, are given in [ZW04] using decision trees to represent the rules and in [FTCH09] using an heuristic based on an index to evaluate the strength of a reduct. However, both these works suppose that the Information Table is consistent. Finally, in [LLRZ09, LLRZ11] the problem to update *interesting knowledge* (coverage and accuracy above a given threshold) is studied and proper heuristics given.

The increase of information with respect to values is addressed in [DH07]. The authors show how approximations and positive regions change when an unknown value becomes known and an algorithm to update reducts is also proposed. Grzymala-Busse [GBGB08] analyses in quantitative terms what happens when known values are turned to unknown. The result is that imprecise tables (i.e., with more unknown values) generate better rules.

APPEARED IN FUNDAMENTA INFORMATICAE, 115, 57-74, 2012

Attribute dynamics has been partially tackled in [CC04] and then later in [Ciu11] where it is shown that to an increase of information corresponds a deeper knowledge on the problem and algorithms to update reducts and rules are given. The problem of new attributes has also been discussed in the case of Information Tables with unknown and do-not-care values in [LRW$^+$07] and in rough fuzzy sets in [Che11]. Of course, attribute dynamics is strictly linked to the notion of reduct hence, ideas used in this widely and deeply studied field can also be related to attribute dynamics.

Finally, one can suppose that at a single time step, different evolutions of the systems occurs. Here, we suppose that if this happens, we are always able to split the event is sub-events which cover only one kind of dynamics. A different approach is given in [CLZ10b] where the problem of simultaneous increase of knowledge with respect to attributes and objects is studied and a way to update approximations is given.

In the current literature on dynamics in rough sets, some problems have not yet been solved, for instance how positive regions change in presence of new objects or how reducts based on generalized decision can be updated when some unknown value become known. Here we make an overview of the existing results and give the missing ones. Results on how approximations, positive regions and generalized decisions change in presence of an increase of knowledge are given. Then, the problem to update reducts and rules is tackled and proper algorithms provided.

2. Rough Sets Basis

Information Tables (or Information Systems) [Paw81, PS07b] are at the basis of rough sets. They have been defined to represent knowledge about objects in terms of observables (attributes).

Definition 2.1. An *Information Table* is a structure $\mathcal{K}(X) = \langle X,\ A, val,\ F\rangle$ where:

- the universe X is a non empty set of *objects*;
- A is a non empty set of *condition attributes*;
- val is the set of all *possible values* that can be observed for all attributes;
- F (called the *information map*) is a mapping $F : X \times A \to val \cup \{*\}$ which associates to any pair object–attribute, the value $F(x, a) \in val$ assumed by a for the object x. If $F(x, a) = *$ it means that this particular value is unknown.

Let us note that we do not deal with different semantics of incomplete information tables, but simply take into account that for some reason a value can be missing, i.e., $F(x, a) = *$.
A *decision table* is an Information Table where the attributes are divided in two groups $A \cup D$, with A condition attributes and D decision attributes which represent a set of decisions to be taken given the conditions represented by A. Usually, $|D| = 1$, that is, only one decision is considered.

Given an information (or decision) table, the indiscernibility relation with respect to a set of attributes $B \subseteq A$ is defined as

$$x\mathcal{I}_B y \quad \text{iff} \quad \forall a \in B,\ F(x, a) = F(y, a)$$

This relation is an equivalence one, which partitions X in equivalence classes $[x]_B$, our *granules* of information. Due to a lack of knowledge we are not able to distinguish objects inside the granules, thus, it can happen that not all subsets of X can be precisely characterized in terms of the available attributes B. However, any set

$H \subseteq X$ can be approximated by a lower and an upper approximation, respectively defined as:

$$L_B(H) = \{x : [x]_B \subseteq H\} \tag{1a}$$

$$U_B(H) = \{x : [x]_B \cap H \neq \emptyset\} \tag{1b}$$

The pair $(L_B(H), U_B(H))$ is called a *rough set* (let us remark that sometimes with rough set it is meant a set H which cannot be described by means of the equivalence classes, in contrast with an *exact set* K such that $L_B(K) = K = U_B(K)$).

Other forms of imprecision arise in decision tables, when considering the decision attributes D. Indeed, it may happen that two objects with same conditions have different decision. In this case the decision table is said *non-deterministic*, and it is useful to introduce the *generalized decision*:

$$\delta_B(x) = \{i : F(y, d) = i \text{ and } x\mathcal{I}_B y\}$$

which for a given set of conditions collects all the possible decisions. Thus, in a non-deterministic situation, only a subset of objects can be precisely classified: the *positive region* of the decision table, defined as

$$POS_B(\mathcal{K}(X), d) = \cup L_B([x]_{\{d\}})$$

The most important tools available in rough set theory are reducts and rules. A decision table can be simplified by searching for a reduct: the smallest set of attributes which preserves the classification. More precisely, given a decision table, a set of attributes B_1 is a *reduct* of B_2, with $B_1 \subseteq B_2$, if

(R1) B_1 and B_2 generate the same generalized decision: for all objects $x \in X$, $\delta_{B_1}(x) = \delta_{B_2}(x)$;

(R2) A minimality condition holds, that is B_1 is the smallest set which satisfies condition (R1): if there exist a set C such that $\delta_C = \delta_{B_1} = \delta_{B_2}$, then $B_1 \subseteq C \subseteq B_2$.

Clearly, there can be more than one reduct for a given set B_2.

Another way to define a reduct is to consider the smallest set of attributes which preserves the positive region. Formally, a set of attributes B_1 is a *reduct* of B_2 if

(R1$_{pos}$) B_1 and B_2 generate the same positive region: $POS_{B_1}(\mathcal{K}(X), d) = POS_{B_2}(\mathcal{K}(X), d)$;

(R2$_{pos}$) A minimality condition holds, that is B_1 is the smallest set which satisfies condition (R1$_{pos}$): if there exist a set C such that $POS_C(\mathcal{K}(X), d) = POS_{B_1}(\mathcal{K}(X), d) = POS_{B_2}(\mathcal{K}(X), d)$, then $B_1 \subseteq C \subseteq B_2$.

If generalized decisions are preserved then also positive region is, but not the other way round.

Finally, classification rules can be deduced by a reduct or directly computed by a proper algorithm, for instance LEM (and its descendants). Here, we are not going into details on how we can obtain rules, for an overview and references the reader is referred to [PS07a]. We just denote a rule as $r : a_1 = v_1, \ldots, a_n = v_n \rightarrow d_1$ or $d_2 \ldots$ or d_m, with the meaning that when conditions $a_i = v_i$ are satisfied then an object can belong to one of the decisions d_j. Of course, in the deterministic case we have $m = 1$, that is only one decision is possible. In the sequel, given a rule r, the set of its conditions will be denoted as $cond(r) = \{(a_i = v_i) : i = 1 \ldots n\}$ and the set of its decisions as $dec(r) = \{d_1, \ldots, d_m\}$.

Information Tables are not the only starting point in rough set theory. Indeed, from an Information Table we can define a binary relation on objects, traditionally

an equivalence or tolerance one and more generally, any binary relation. This relation is then used to cluster objects in granules and define approximations. A different approach consists in taking for granted to have a relation and begin the investigation from a so called Approximation Space.

Definition 2.2. An *Approximation Space* is a pair $\mathcal{A} = (X, R)$ with X a set of objects and R a binary relation on X.

Moreover, we can go further and get rid of the relation, supposing to have available somehow a granulation of our objects under investgation. We thus arrive at a covering of the universe of discourse.

Definition 2.3. Let X be a non empty set, a covering $\mathbb{C}(X)$ of X is a collection of sets $C_i \subseteq \mathcal{P}(X)$ such that $\bigcup C_i = X$.

3. A Classification of Dynamics

In a broad sense, we can distinguish between two type of dynamics: synchronic and diachronic (or asynchronous) [Pag04]. In synchronic dynamics time is fixed and it is typical of a multi-source or multi-agent situation: two stockbrokers with different predictions, different web-sites about weather forecasts, etc. Here, we are dealing with asynchronous dynamics where knowledge is supposed to change in time and in particular with an increase of information, that is, the knowledge already acquired cannot change, only new pieces of information can be added. These changes can regard different factors: new objects enter into the system under investigation, new facts are taken into account or unknown facts become known. Of course, several of these changes can appear simultaneously, for instance going from time t to time $t+1$, it may happen that n new objects enter the system and m new facts are considered. However, in this case, we suppose to be able to split the two events in two separate steps: from time t to $t+\frac{1}{2}$, n new objects enter the system and from time $t+\frac{1}{2}$ to time $t+1$, m new facts happen.

We are now going to give a classification of these dynamics in Information Tables. Moreover, we also touch the problem to define and treat dynamics in Approximation Spaces and coverings with the aim of reporting some results obtained in the Information Table case to the other two environments.

3.1. Temporal Dynamics in Information Tables.

If we consider an Information Table evolving in time, it may change in terms of objects, attributes, values or information map. Generally speaking, a change can be of any kind. For instance, a value of an attribute for a given object can change from a time step to another. However, here we want to investigate those situations characterized by a monotone increase of knowledge from time t to time $t+1$. So, it is possible to add information but not to alter the existing one. Three situations where the knowledge increases in time are now formalized in the following way.

Definition 3.1. [Ciu10a] Let $\mathcal{K}^{(t_1)}(X_1) = \langle X_1, A_1, val_1, F_1\rangle$ and $\mathcal{K}^{(t_2)}(X_2) = \langle X_2, A_2, val_2, F_2\rangle$, with $t_1, t_2 \in \mathbb{N}$, $t_1 \leq t_2$ be two Information Tables. We will say that there is a *monotonic increase of information* from time t_1 to time t_2

- wrt *values* iff $\mathcal{K}^{(t_1)}$ and $\mathcal{K}^{(t_2)}$ are defined on the same set of objects, attributes and values and $F_1(x,a) \neq *$ implies $F_2(x,a) = F_1(x,a)$.

- wrt *attributes* iff $X_1 = X_2$, i.e., $\mathcal{K}^{(t_1)}$ and $\mathcal{K}^{(t_2)}$ are defined on the same set of objects and $A_1 \subseteq A_2$, $val_1 \subseteq val_2$ and $\forall a \in A_1$, $\forall x \in X_1, F_2(x,a) = F_1(x,a)$.

- wrt *objects* iff $\mathcal{K}^{(t_1)}$ and $\mathcal{K}^{(t_2)}$ have the same set of attributes and values, $X_1 \subseteq X_2$ and $\forall x \in X_1$, $F_2(x,a) = F_1(x,a)$.

In all the three cases we can also define a *decrease of knowledge* when the reverse ordering holds.

Example 3.1. Let us consider a simple example of a medical decision table in which patients are characterized according to some attributes and a decision on their disease has to be taken. In tables 1 and 2 we can see a monotone increase of information wrt attributes from time t_0 to time t_1. Indeed, the difference between time t_0 and time t_1 is the new attribute "Temperature" which is added while the others do not change.

TABLE 1. Medical decision table, time t_0.

Patient	Pressure	Headache	Muscle Pain	Disease
p_1	2	yes	yes	A
p_2	3	no	yes	B
p_3	1	yes	no	NO
p_4	2	yes	yes	NO

TABLE 2. Medical decision table, time t_1.

Patient	Temperature	Pressure	Headache	Muscle Pain	Disease
p_1	*	2	yes	yes	A
p_2	high	3	no	yes	B
p_3	normal	1	yes	no	NO
p_4	high	2	yes	yes	NO

On the other hand, from time t_1 to time t_2 we have a monotone increase of knowledge with respect to values, since the only modification is the value F(p_1,Temperature) which from missing becomes defined.

TABLE 3. Medical decision table, time t_2.

Patient	Temperature	Pressure	Headache	Muscle Pain	Disease
p_1	very high	2	yes	yes	A
p_2	high	3	no	yes	B
p_3	normal	1	yes	no	NO
p_4	high	2	yes	yes	NO

Finally, at time t_3 a new object p_5 enters into the system, and from table 3 to table 4 we have a monotone increase of information wrt objects.

TABLE 4. Medical decision table, time t_3.

Patient	Temperature	Pressure	Headache	Muscle Pain	Disease
p_1	very high	2	yes	yes	A
p_2	high	3	no	yes	B
p_3	normal	1	yes	no	NO
p_4	high	2	yes	yes	NO
p_5	high	3	yes	yes	NO

3.2. **Temporal Dynamics in Approximation Spaces and Coverings.** In order to reflect the increase of information from Information Tables to Approximation Spaces and coverings, we need to define what the increase of information in these two further settings is.

If we directly start our investigation from an Approximation Spaces there are two sources of information which can vary: the set of objects X and the relation R. Thus, we can derive two notions of increase of knowledge in Approximation Spaces.

Definition 3.2. [Ciu10a] Given two Approximation Spaces $\mathcal{A}^{(t_1)} = (X_1, R_1), \mathcal{A}^{(t_2)} = (X_2, R_2)$ with $t_1, t_2 \in \mathbb{N}$, $t_1 \leq t_2$, we have an *increase of knowledge in Approximation Spaces*

- wrt *relations* if $R_1 \subseteq R_2$ and the objects are the same, i.e., $X_1 = X_2$;
- wrt *objects* if $X_1 \subseteq X_2$ and the relations are defined in the same way on all common objects, i.e., let $Y = \bigcap X_i$, then for all $x, y \in Y$, $R_1(x, y)$ iff $R_2(x, y)$.

That is, either we add new objects and new relations involving them without affecting the existing ones or we add new relations among the existing objects.

Example 3.2. Let us consider an Approximation Space at time t_0 represented by $X_0 = \{a, b, c, d\}, R_0 = \{(i, i), (b, c), (a, d)\}$, where i stands for any object, i.e., the relation is reflexive. Then, at time t_1 we have an increase of knowledge with respect to objects if the approximation space is updated as $X_1 = \{a, b, c, d, e\}, R_1 = \{(i, i), (b, c), (a, d), (d, e)\}$. That is, we added object e and the relations involving it. At time t_2 we have a monotone increase of knowledge with respect to relations if, for instance, the approximation space is $X_2 = \{a, b, c, d, e\}, R_2 = \{(i, i), (b, c), (a, d), (d, e), (b, d), (c, e)\}$. That is, objects are the same but new relations between b and d and between c and e are added.

As a further step of abstraction we can consider to have directly available the granules of our universe. In this case, in order to deal with increase of knowledge an ordering among coverings is required. This is not so trivial, since different equivalent definitions of partition orderings differ when generalized to coverings (see [BC09] for an overview). So, we assume to have a notion of ordering (or quasi-ordering) available, let us call it $\preceq$, and define monotonicity with respect to this order as follows.

Definition 3.3. [Ciu10b] Given two coverings $\mathbb{C}^{(t_1)}(X_1), \mathbb{C}^{(t_2)}(X_2)$ with $t_1, t_2 \in \mathbb{R}$, $t_1 \leq t_2$, we have an *increase of knowledge in coverings*

- wrt *objects* of type 1 if $X_1 \subseteq X_2$, $\mathbb{C}^{(t_1)}(X_1) \subseteq \mathbb{C}^{(t_2)}(X_2)$ and all $\gamma \in \mathbb{C}^{(t_2)}(X_2) \backslash \mathbb{C}^{(t_1)}(X_1)$ contain at least one element $x \in X_2 \backslash X_1$;

- wrt *objects* of type 2 if $X_1 \subseteq X_2$ and the new objects are added to an existing class or form a new class. That is, for all the granules $\gamma_2 \in \mathbb{C}^{(t_2)}(X_2)$ either $\gamma_2 = \gamma_1$ with $\gamma_1 \in \mathbb{C}^{(t_1)}(X_1)$ or $\gamma_2 = \gamma_1 \cup \{x_1, \ldots, x_n | x_i \in X_2 \backslash X_1\}$ (and in this case $\gamma_1 \notin \mathbb{C}^{(t_2)}(X_2)$) or $\gamma_2 = \{x_1, \ldots, x_m | x_i \in X_2 \backslash X_1\}$;
- wrt *granules* if $X_1 = X_2$ and $\mathbb{C}^{(t_1)}(X) \leq \mathbb{C}^{(t_2)}(X)$.

In the case of monotonicity of type 1 all the granules at time t_1 remains at time t_2 and new ones are added which contains at least a new element. This evolution is more "faithful" to Approximation Spaces dynamics in case the coverings are obtained from an Approximation Space [Yao98, Ciu10a].
In the monotonicity of type 2, the new elements either constitute a new class or are added to an existing one. As shown in [Ciu10b] this evolution gives a sufficient condition to obtain the conservation of covering reduction.

Example 3.3. Consider the universe $X = \{a, b, c, d\}$ and the covering $\{\{a, d\}, \{b, c\}\}$. At a following time a new object e can enter the system. Then, we have an increase of knowledge in coverings wrt objects of type 1 if the covering becomes, for instance, $\{\{a, d\}, \{b, c\}, \{d, e\}, \{e\}\}$. On the other hand, we have an increase of knowledge in coverings wrt objects of type 2 if the new covering is $\{\{a, d\}, \{b, c, e\}, \{e\}\}$. Then, if this last system is updated such that the new covering is $\{\{a, d\}, \{b, c, e\}, \{c, e\}, \{e\}\}$ we have an increase of knowledge with respect to granules if the following quasi ordering is considered:

$$\mathbb{C}_1(X) \leq \mathbb{C}_2(X) \quad \text{iff} \quad \forall C_i \in \mathbb{C}_1(X)\ \exists D_j \in \mathbb{C}_2(X) \quad \text{such that} \quad C_i \subseteq D_j.$$

4. Dynamics and Rough Sets Instruments

According to the above presented classification, we now discuss how rough set instruments change in presence of an increase of information. At first, we give results on approximations, positive regions and generalized decisions. Then, in the case of reducts and rules, after on overview of existing literature, algorithms to update them are proposed. Finally, the influence of these results on Approximation Spaces and coverings is discussed.

4.1. **Approximations.**

Proposition 4.1. [Ciu11] *Let $\mathcal{K}^{(t_0)}(X)$ and $\mathcal{K}^{(t_1)}(X)$ be two Information Tables characterized by a monotone increase of knowledge with respect to* **attributes** *from time t_0 to time t_1. If we denote the attributes at time t_0 as A_0 and at time t_1 as A_1, then*

$$L_{A_0}(H) \subseteq L_{A_1}(H) \subseteq H \subseteq U_{A_1}(H) \subseteq U_{A_0}(H)\,.$$

Proposition 4.2. [DH07] *Let $\mathcal{K}^{(t_0)}(X)$ and $\mathcal{K}^{(t_1)}(X)$ be two Information Tables characterized by a monotone increase of knowledge with respect to* **values** *from time t_0 to time t_1 and A a set of attributes. Then*

$$L_A^{t_0}(H) \subseteq L_A^{t_1}(H) \subseteq H \subseteq U_A^{t_1}(H) \subseteq U_A^{t_0}(H)\,.$$

Proposition 4.3. *Let $\mathcal{K}^{(t_0)}(X)$ and $\mathcal{K}^{(t_1)}(X)$ be two Information Tables characterized by a monotone increase of knowledge with respect to* **objects** *from time t_0 to time $t_1$and A a set of attributes. Then*

$$L_A^{t_1}(H) \subseteq L_A^{t_0}(H) \subseteq H \subseteq U_A^{t_0}(H) \subseteq U_A^{t_1}(H)\,.$$

Proof. Having more objects available, granules can become bigger. Thus, as an easily consequence we get that lower approximations can become smaller and upper approximations bigger. □

Let us note that propositions 4.2 and 4.3 apply to more general rough-set models, for instance tolerance rough sets. On the contrary, proposition 4.1 is more problematic. Indeed, in general and contrary to what we said in [CC04], it does not hold in the case of similarity rough sets. As an example just consider the relation "x and y are similar if they have at least one half of the attributes in common". Then, if we take the Information Table 3, we see that with respect to the set of all condition attributes $L^{t_1}(\{p_1, p_2\}) = \emptyset$ whereas not considering the attribute Temperature $L^{t_0}(\{p_1, p_2\}) = \{p_2\}$ and so $L^{t_0}(\{p_1, p_2\}) \not\subseteq L^{t_1}(\{p_1, p_2\})$. Further conditions has to be added to the similarity relation to have also the monotonicity on the approximations (see for instance [SS96]) or the approximations should be defined in a different way. For instance, in [LRW+07], approximations with "unknown" and "do not care" values are obtained using the characteristic relation approach. In this last case a proposition analogous to 4.1 and a way to update approximations in presence of incoming attributes are given. Similarly, for rough fuzzy sets, it is possible to define approximations in a way to guarantee monotonicity of approximations while adding attributes [Che11].

4.2. **Positive Regions and Generalized Decision.** The propositions of the previous section have a direct consequence on the number of objects that can be correctly classified, that is on the size of the positive region.

Proposition 4.4. [Ciu11] *Let $\mathcal{K}^{(t_0)}(X)$ and $\mathcal{K}^{(t_1)}(X)$ be two Information Tables characterized by a monotone increase of knowledge with respect to* **attributes** *from time t_0 to time t_1. If we denote the attributes at time t_0 as A_0 and at time t_1 as A_1, then*

$$POS_{A_0}(\mathcal{K}^{(t_0)}(X), d) \subseteq POS_{A_1}(\mathcal{K}^{(t_1)}(X), d)$$

Proposition 4.5. [DH07] *Let $\mathcal{K}^{(t_0)}(X)$ and $\mathcal{K}^{(t_1)}(X)$ be two Information Tables characterized by a monotone increase of knowledge with respect to* **values** *from time t_0 to time t_1. Then,*

$$POS_B(\mathcal{K}^{(t_0)}(X), d) \subseteq POS_B(\mathcal{K}^{(t_1)}(X), d)$$

Proposition 4.6. *Let $\mathcal{K}^{(t_0)}(X)$ and $\mathcal{K}^{(t_1)}(X)$ be two Information Tables characterized by a monotone increase of knowledge with respect to* **objects** *from time t_0 to time t_1. Then,*

$$POS_B(\mathcal{K}^{(t_1)}(X), d) \subseteq POS_B(\mathcal{K}^{(t_0)}(X), d)$$

Proof. An easy corollary of proposition 4.3. □

As expected, generalized decisions have the opposite behaviour: for a given object, its generalized decision becomes smaller in attributes and values increase of knowledge, and bigger in the case of more objects.

Proposition 4.7. [Ciu11] *Let $\mathcal{K}^{(t_0)}(X)$ and $\mathcal{K}^{(t_1)}(X)$ be two Information Tables characterized by a monotone increase of knowledge with respect to* **attributes** *from time t_0 to time t_1. Then, for all $x \in X$:*

$$\delta_{A_{t_1}}(x) \subseteq \delta_{A_{t_0}}(x).$$

Proposition 4.8. *Let $\mathcal{K}^{(t_0)}(X)$ and $\mathcal{K}^{(t_1)}(X)$ be two Information Tables characterized by a monotone increase of knowledge with respect to* **values** *from time t_0 to time t_1. Then, for all $x \in X$:*

$$\delta_{A_{t_1}}(x) \subseteq \delta_{A_{t_0}}(x).$$

Proof. With more available values there is a finer granulation, thus an object x at time t_1 can be indiscernible with fewer objects with respect to time t_0. □

Proposition 4.9. *Let $\mathcal{K}^{(t_0)}(X)$ and $\mathcal{K}^{(t_1)}(X)$ be two Information Tables characterized by a monotone increase of knowledge with respect to* **objects** *from time t_0 to time t_1. Then, for all $x \in X_0$:*

$$\delta_{A_{t_0}}(x) \subseteq \delta_{A_{t_1}}(x).$$

Proof. At time t_1, an object x can be indiscernible from more objects than at time t_0 (the ones that entered the system at time t_1), thus the generalized decision can become wider. □

4.3. **Reducts Update.** Now, let us consider the reducts. The problem to update them and test the performances of the associated rules has been tackled in the following works

(1) Increase of knowledge wrt attibutes in [Ciu11], reducts based on the generalized decision;
(2) Increase of knowledge wrt values in [DH07, GBGB08], reducts based on the positive region;
(3) Increase of knowledge wrt objects in [SZ95, Woj01], reducts based on the positive region;

As can be seen, only one kind of reduct has been considered in each type of increase of knowledge. Here, we review these methods and give the ones concerning the other kind of reduct.

Table 4.3 summarizes where each algorithm is introduced and the complexity of each of them. The information about the rules update is also report and it will be discussed in the following section. We note that in [Ciu11] the cost of the algorithms has not been explicitly computed but it can be simply recovered from the algorithm.

Increase in attributes. Supposing that $A_{t_0} \subseteq A_{t_1}$, it can happen that Red_{t_0}, the reduct of A_{t_0}, is a reduct of A_{t_1}. In this case the reduct at time t_1 does not change. On the contrary, if Red_{t_0} is not a reduct of A_{t_1}, we have that $Red_{A_{t_0}} \subseteq Red_{A_{t_1}}$ and the rules obtained at time t_1 are more precise, in the sense that they contain more conditions. If we desire to compute the reducts at time t_1 we can start from an existing one and update it, instead of re-calculating them all. The way in which we do this depends on the kind of reducts we are using: based on the generalized

Increase	What	Where	Complexity
Attribute	Reduct - positive region	Alg. 1	$O(\|X\|^2\|A\|)$
	Reduct - gen. decision	[Ciu11]	$O(\|X\|\|A\|)$
	Rules	[Ciu11]	$O(\|X\| + \|Val_a\|)$
Values	Reduct - positive region	[DH07]	$O(\|A\|^2\|X\| \log \|X\|)$
	Reduct - gen. decision	Alg. 2	$O(\|X\|^2\|A\|^2)$
	Rules	Section 4.4 - Increase in values	$O(\|R\| \cdot \max\{\|R\|, \|Val_a\|\})$
Objects	Reduct - positive region	Alg. 3	$O(\|X\|^2 p 2^p)$
	Reduct - gen. decision	Alg. 4	$O(\|X\|^2 p 2^p)$
	Rules	Alg. 5	$O(\|R\|)$

TABLE 5. In the last column, X is the number of objects, A is the number of attributes, $|Val_a|$ is the number of values that attribute a can assume, $|R|$ the number of rules and $p = |A| - |Red|$, with Red the reduct to be updated.

decision or on the positive region. The former has been discussed in [Ciu11] whereas a way to update a positive-region reduct is presented in algorithm 1.

Algorithm 1 Update reducts (positive region) in case of a new attribute

Require: A set of objects $X = \{x_1, x_2, \dots x_n\}$, a reduct Red_{t_0}, a new attribute a
Ensure: An updated reduct

1: **if** $POS_A(\mathcal{K}(X), d) \neq POS_{(A \cup \{a\})}(\mathcal{K}(X), d)$ **then**
2: add a to the reduct: $Red_{t_1} = Red_{t_0} \cup \{a\}$
3: **end if**

That is, we add a to the reduct Red_{t_0} if it enables to discern objects belonging to different decision classes which were equivalent with respect to attributes in Red_{t_0}. The cost of the algorithm is the cost of computing the Positive Region, which depends both on the number of objects and attributes and can be estimated (without any particular heuristics) as $O(|X|^2|A|)$.

Clearly, if we add more than one attribute we can proceed by applying the above algorithm for each attribute. In this case, the order of considering the attributes will influence the result (heuristics can be found in literature [PS07a]) and we may also loose some reduct respect to computing them from scratch. Another approach could be to ask if there exists a reduct at time t_1 which contains the reduct at time t_0. This can be solved in polynomial time similarly to the covering problem discussed in [MSS08].

Example 4.1. Let us consider the monotone increase with respect to attributes existing from table 1 to table 3. At time t_0 a possible reduct is made of only one attribute: Red_{t_0} ={Pressure}. At time t_1 we add the attribute Temperature and we recompute the positive region. We have that $POS_A(\mathcal{K}(X), d) = \{x_2, x_3\} \neq X = POS_{(A \cup \{Temp\})}(\mathcal{K}(X), d)$. Hence, we add the attribute Temperature and obtain Red_{t_1} ={Pressure, Temperature}.

Increase in values. In case of a monotone increase of knowledge with respect to values, Deng and Huang [DH07] defined an algorithm to generate an updated reduct

based on the positive region without recomputing it from scratch. The idea is that, at a given time step, those attributes with missing values in the negative region, useful for classification or not, are put in the reduct. Indeed, when those values will become known they could be useful to give a better classification by eliminating some objects from the negative region. Then, the reduct at time $t+1$ is computed starting from the reduct at time t and not from the whole set of attributes.

A similar procedure can also be used when computing reducts based on the generalized decision, instead of on the positive region. That is, a reduct must contain all the attributes $a \subset A$ such that there exist an object x, $F(x,a) = *$ and $|\delta_A(x)| > 1$, i.e. there is no certain decision on x. This is due to the fact that at a certain point in the future the value $F(x,a)$ could become known and contribute to decide on x.

Example 4.2. Let us consider the Information Table 2 at time t_0 with one missing value which evolves at time t_1 according to table 3.

The relation between objects is the usual one when dealing with missing values [Kry98]:

$$\begin{aligned} \forall x, y \in X: \quad & x \mathcal{R}_D y \quad \text{iff} \quad \forall a_i \in D \subseteq A, \\ & \text{either} \quad F(x,a_i) = F(y,a_i) \quad \text{or} \quad F(x,a_i) = * \quad \text{or} \quad F(y,a_i) = * \end{aligned}$$

Starting to examine the attribute in the order from left to right, we see that temperature cannot be eliminated since the missing attribute is relative to the object p_1 about which we have not a certain decision. Then, Pressure can be deleted, whereas Muscle Pain and Headache cannot. So the reduct at this stage is Red_0={Temperature, Headache, Muscle Pain}.

At time t_1 we start from Red_0 and see that Muscle Pain can now be deleted with the result that Red_1= {Temperature, Headache}.

We note that without the criterion to maintain the attributes with objects with missing values, Temperature would have been deleted.

The above procedure to update a reduct can be schematized as in algorithm 2.

Algorithm 2 Update reducts (generalized decision) in case of an increase of values

Require: A decision table at time t_1, a reduct Red_{t_0}
Ensure: An updated reduct

1: $Red_{t_1} = Red_{t_0}$
2: **for** $a \in Red_{t_0}$ **do**
3: **if** $\delta_{Red_{t_1} \setminus a} = \delta_{Red_{t_1}}$ **then**
4: **if** $\forall x \in X, |\delta_{Red_{t_1}}| = 1$ or $F(x,a) \neq *$ **then**
5: $Red_{t_1} = Red_{t_1} \setminus a$
6: **end if**
7: **end if**
8: **end for**

The cost of the algorithm depends of the size of the reduct. Assuming the worst case, $R = |Red_{t_0}| = |A|$, we have that R should be multiplied by the cost of computing the generalized decisions of line 3 (and 4) which is $|X|(|X||A|)$. Summarizing, we have a complexity of $O(|X|^2|A|^2)$.

Increase in objects. In the case of objects Shan and Ziarko [SZ95] propose an algorithm to update all the reducts when one (and only one) new object becomes known. The idea is to maintain a decision matrix for each decision class, where rows contain objects belonging to the lower approximation of the class and columns not belonging. The decision rules are computed for each class. When a new object enters into the system, it is added as a row to the decision matrix corresponding to its decision class, say d, and as a column to all the others. In the first case, new rules are added to the set of rules deciding d, if the new object is consistent with already existing ones. In the second case, all rules corresponding to the other classes are updated and if the new object is inconsistent with some other, the old rules are deleted.

This result has been modified and improved with respect to complexity issues by Wojna [Woj01]. The basic idea is to maintain only the rules/reduct satisfying a given constraint, instead of computing all rules and reducts. In particular, some experiments are conducted using as a constraint the coverage greater than a given threshold. As a result "the application of stronger constraints brought a significant reduction of used memory and time and very small deterioration of accuracy or even improvement".

Further, in [LLRZ09], authors study how accuracy and coverage of rules vary when new objects enter/exit the system. They also propose an algorithm to update so called *interesting knowledge*, i.e., rules with coverage and accuracy above a given threshold, when simultaneously some objects enter or exit the system. The performance of this approach are then improved in [LLRZ11].

However, if we wonder how to update a given reduct or rule when a new object enter the system, none of the above works answer the question. Indeed, [SZ95] (and also [Woj01]) only deals with consistent rules and reducts and rules are not updated but computed from reducts. On the contrary [LLRZ09] consider also inconsistent rules but only "interesting knowledge" is updated. Now, we present a simple solution on how to update reducts (both based on generalized decision and positive regions) in presence of a new object while rules are treated in the next section.

Algorithm 3 Update reducts (positive region) in presence of a new object

Require: A set of attributes A, a reduct $Red_{t_0} \subseteq A$, a new object x
Ensure: A collection of reducts RED which updates Red_{t_0}

1: $D = \{y : F(y,d) = F(x,d)\}$
2: **if** $L_{Red_{t_0}}(D) = L_A(D)$ **then**
3: $\quad RED = \{Red_{t_0}\}$
4: **else**
5: $\quad$ Search for the smallest subsets $Red_{t_0} \subset B_i \subseteq A$ such that $L_{B_i}(D) = L_A(D)$
6: $\quad RED = \cup\{B_i\}$
7: **end if**

As shown in algorithms 4 and 3, given Red_{t_0} a reduct at time t_0 and a new object, we first check if Red_0 is still a reduct of the new system. If not, we search for the smallest subsets B_i of attributes containing Red_{t_0} such that B_i is a reduct with respect to the positive region condition or the generalized decision one.

Algorithm 4 Update reducts (generalized decision) in presence of a new object

Require: A set of attributes A, a reduct $Red_{t_0} \subseteq A$, a new object x
Ensure: A collection of reducts RED which updates Red_{t_0}

1: **if** $\delta_{Red_{t_0}}(x) = \delta_A(x)$ **then**
2: $RED = \{Red_{t_0}\}$
3: **else**
4: Search for the smallest subsets $Red_{t_0} \subset B_i \subseteq A$ such that $\delta_B(x) = \delta_A(x)$
5: $RED = \cup\{B_i\}$
6: **end if**

Example 4.3. Let us consider table 3 and its reduct Red_{t_0}={Temperature, Pressure}. At a following time, object p_5 is added as in table 4 and Red_{t_0} is no more a reduct of the Information Table. Indeed, at time t_1 patients p_2 and p_5 do not belong to the positive region and the generalized decision of the same objects is different if computed on the set Red_{t_0} of attributes or on all the attributes. Thus, to Red_{t_0} we have to add some new attribute to get a reduct, and the only solution is to add Headache: RED={{Temperature,Pressure,Headache}}. We note that RED is not the set of all reducts (for instance, another could be {Temperature, Headache, Muscle Pain}), but only of the ones obtained from Red_{t_0}.

The complexity of both algorithms in the worst case is given by the *else* branch which requires to search a new reduct. Let $p = |A| - |Red_{t_0}|$, then we have that the cost of line 4 in algorithm 4 and line 5 in algorithm 3 is $|X|^2 \sum_{i=1}^{p} \binom{p}{i}(|Red_{t_0}| + i)$ where $|Red_{t_0}| + i$ is the cardinality of $|B_i|$. The solution of this sum is $|X|^2((2^p - 1) + (p2^{p-1}))$ and so we have a complexity in the worst case of $O(|X|^2 p2^p)$.

4.4. **Rules Update.** Similar methods to the ones described for updating reducts can be applied directly to rules updating.
Increase in attributes. The case of increase of attributes has been considered in [Ciu11]. The deterministic rules are not changed, since the new attribute will not affect the classification. Imprecise rules can be improved if the new attribute is able to discern similar objects with different decisions. That is, we add a new rule if the set of objects satisfying the conditions of the actual rule is not contained in one of the equivalence classes with respect to the new attribute. Let us note that the new rules are better than the hold ones in the sense that they have less elements in the right (decision) part.

From a theoretical standpoint, we expect that the rules computed with more attributes are more accurate. We can however wonder if, as in the case of values increase of knowledge, the less accurate rules have better performances. In order to verify this, we made some test in [Ciu11]. We took into account the data sets Iris, Pima Diabetes and Breast Cancer from the UCI repository[6]. As a result, we can observe that on average accuracy decreases as attributes diminish. That is, to a monotone decrease of knowledge wrt attributes corresponds a decrease of accuracy. More interesting, we can also get some (indirect) indication on the dependence of the decision from the conditions and, further, in some cases, the accuracy of some subset of attributes is better than the whole set of attributes. This could be useful in a pre-processing phase, to understand which are the most important attributes.

[6]http://archive.ics.uci.edu/ml/

Of course, proper studies are needed to develop methods able to single out in an efficient way those particular attributes.
Increase in values. In case of increase of values we can distinguish four cases. Let us suppose to have a rule $r : a_1 = v_1, \dots a_n = v_n \to d_1$ or $d_2 \dots$ or d_m and that the value $F(x, a)$ becomes known. Then, the scenarios are:

(1) $a \notin \{a_1, \dots, a_n\}$ and the rule matches x conditions. Check if the rules r' with conditions $\{a, a_1, \dots, a_n\}$ are better than r, that is if the decision set $dec(r')$ is smaller than r. If this is the case then substitute r with the rules r' otherwise do nothing.
(2) $a \notin \{a_1, \dots, a_n\}$ and the rule does not match x conditions. In this case there is nothing to do.
(3) $a \in \{a_1, \dots, a_n\}$ and the rule matches x conditions. If also $F(x, d) \in dec(r)$ then the rule is unchanged, otherwise the decision of x, i.e. $F(x, d)$, is added to $dec(r)$.
(4) $a \in \{a_1, \dots, a_n\}$ and the rule does not match x conditions. The rule is unchanged but we have to check if there exists a rule which covers x and contains a, otherwise a new rule obtained by x can be added.

Example 4.4. Let us consider again the increase wrt values going from table 2 to table 3. With respect to the rule

$$r : \text{ Pressure} = 3 \quad \text{and} \quad \text{Muscle-Pain} = \text{yes} \to B$$

we are in case 2, so there is nothing to do. On the contrary, we have to apply case 1 to the rule

$$r : \text{ Pressure} = 2 \quad \text{and} \quad \text{Muscle-Pain} = \text{yes} \to A \,\text{or}\, NO$$

So, we add the attribute Temperature to the rule and obtain two better (i.e., more certain) rules:

$$r_1 : \text{ Temperature=very high} \quad \text{and Pressure} = 2 \quad \text{and} \quad \text{Muscle-Pain} = \text{yes} \to A$$
$$r_2 : \text{ Temperature=high} \quad \text{and Pressure} = 2 \quad \text{and} \quad \text{Muscle-Pain} = \text{yes} \to NO$$

and substitute r with r_1 and r_2.

The cost of this procedure depends on the number of rules to update, which is at most equal to the number of all rules $|R|$, and the worst case between 1 and 4. Case 1 requires $c_1 \cdot |Val_a|$ operations and case 4 at most $c_2|R|$ (c_1, c_2 are constants), which makes the total complexity $O(|R| \cdot \max\{|R|, |Val_a|\})$.
Increase in objects. In the case of objects increase of information, let us consider that a new object x enters the system. Then, if a rule also correctly covers x there is no need to change it. If a non-deterministic rule do not correctly covers x, that is rule and object have same conditions but different decision, then x-decision can be added to the rule's decision. On the contrary, if a deterministic rule do not correctly applies to x, we have two choices: delete the rule, if only deterministic rules are desired, or add the x-decision to the rule's decisions, making it an uncertain rule. Finally, if there are no rules matching x-conditions, a new rule can be added. This procedure is summarized in algorithm 5.

Example 4.5. Let us consider table 3 and the rule:

$$r : \text{ Temperature} = \text{high} \quad \text{and} \quad \text{Pressure} = 3 \to B$$

Algorithm 5 Update rules - increase of objects

Require: a set of rules R_{t_0}, a new object x
Ensure: a new set of rules R_{t_1}
1: boolean cons {cons=1 means that the new rules cannot add inconsistency}
2: $R_{t_1} = \emptyset$
3: **if** $\nexists r \in R_{t_0}$ matching x conditions **then**
4: $r = cond(x) \to F(x, d)$
5: $R_{t_1} = R_{t_1} \cup \{r\}$
6: **else**
7: **for all** $r \in R_{t_0}$ matching x conditions **do**
8: **if** $F(x, d) \in dec(r)$ OR $cons = 0$ **then**
9: $r' = cond(r) \to (dec(r) \cup \{F(x, d)\})$
10: $R_{t_1} = R_{t_1} \cup \{r'\}$
11: **end if**
12: **end for**
13: **end if**

When patient p_5 enters the system (table 4), the rule does not hold anymore. Thus, we can change the rule as

$$r_1 : \text{ Temperature} = \text{high} \quad \text{and} \quad \text{Pressure} = 3 \to B \,\text{or}\, NO$$

or delete it.

The most similar approach to algorithm 5 to update rules in case of new objects is the one in [ZW04]. In this work the authors introduce a way to organize rules in a decision tree and update them in presence of new objects. When a new object x appears, the rule set is tested and three cases occur:

(1) If there is no rule covering x, a new rule is added;
(2) If there exists a consistent rule covering x, there is nothing to do;
(3) If there exist a rule covering x but which is not consistent with x, then the inconsistent rules are updated adding attributes to the reduct used to generate them.

So, the main difference with our approach is the hypothesis in [ZW04] that there exists a set of attributes which makes the rule set consistent (which in reality is not always the case). Further, in order to keep the system consistent, they try to add new conditions to the rule, while we do keep the set of conditions fixed, in order to not increase the complexity of the algorithm. A similar approach is also proposed in [FTCH09] where rules are updated to accommodate incoming objects and the whole system is supposed to be consistent. The peculiarity of this approach is that it considers a subset of all possible rules, chosen according to a *reduct strength index* and to the merging of reducts with same number of conditions and outcome.

4.5. **Dependencies Among the Three Approaches.** Of course, Information Tables, Approximation Spaces and coverings are not independent tools and as outlined in section 2, it is possible to define the increase of knowledge also in these two environments. So, we can ask, given an increase of knowledge in information tables, which is the relapse on the dynamics of approximation spaces and coverings. Unfortunately, very few can be said. Indeed, we just have the following results [Ciu10a, Ciu10b]:

- an increase of information wrt objects in an Information Table induces an increase of information in an Approximation Space;
- an increase of knowledge wrt objects of type 1 in Approximation Spaces induces an increase of knowledge wrt objects in coverings.

Other dependencies generally do not hold. Indeed, there can be a monotone increase of knowledge wrt to attributes or values in an Information Table which does not reflect in a monotone relation in Approximation Spaces. Similarly, since the increase of knowledge wrt to granules depends on a given ordering, it is not so immediate to give a general result on the dependencies among Approximation Spaces and coverings. However, looking at the covering in example 3.3, we can see that it can be obtained by the Approximation Spaces in example 3.2 using the successor neighborhood. Thus, in this case, an increase of knowledge wrt relations in Approximation Spaces induces an increase of knowledge wrt granules in coverings.

Thus, considering the results of Section 4 we have that only proposition 4.3 can be applied to Approximation Spaces and coverings. Indeed other kinds of monotonicity do not propagate from Information Tables to the other paradigms or simply do not apply to them, for instance, attribute reduction has not sense in absence of attributes.

5. Conclusion

The influence of dynamic evolution on rough sets has been studied. From a theoretical standpoint (and as one could expect) we have better approximations and classifications with attribute and value increase of information and worst in case of objects increase. From an application standpoint, we gave a way to update reducts and rules. In this direction, some tests should be done. First of all to understand the consequence, both in efficiency and quality, of updating instead of re-computing. Further, as outlined in [GBGB08, Ciu11] to a decrease of information values and attributes can correspond better rules. It should be investigated this issue in a deeper way, in order to understand, if possible, when this happens and in which proportion.

Acknowledgement

The author wants to express his gratitude to Gianpiero Cattaneo for having introduced him to rough set theory and addressed his attention to dynamic problems [CC04].

Many thanks also to the reviewers for their suggestions and in particular for pointing out interesting references.

References

[BC09] D. Bianucci and G. Cattaneo, *Information entropy and granulation co-entropy of partitions and coverings: A summary*, Transactions on Rough Sets **10** (2009), 15–66.

[CC04] G. Cattaneo and D. Ciucci, *Investigation about Time Monotonicity of Similarity and Preclusive Rough Approximations in Incomplete Information Systems*, LNAI, vol. 3066, Springer, 2004, pp. 38–48.

[Che11] Y. Cheng, *The incremental method for fast computing the rough fuzzy approximations*, Data& Knowledge Engineering **70** (2011), no. 1, 84–100.

[Ciu10a] D. Ciucci, *Classification of rough sets dynamics*, Proceedings RSCTC10, LNAI, vol. 6086, 2010, pp. 257–266.

[Ciu10b] ———, *Temporal dynamics in rough sets based on covering*, Proceedings RSKT10, LNAI, vol. 6401, 2010, pp. 126–133.

[Ciu11] ———, *Attribute dynamics in rough sets*, Proceedings ISMIS11 (Marzena Kryszkiewicz, Henryk Rybinski, Andrzej Skowron, and Zbigniew W. Ras, eds.), LNCS, vol. 6804, 2011, pp. 43–51.

[CLQR10] H. Chen, T. Li, S. Qiao, and D. Ruan, *A rough set based dynamic maintenance approach for approximations in coarsening and refining attribute values*, Int. J. Intelligent Systems **25** (2010), no. 10, 1005–1026.

[CLZ10a] H. Chen, T. Li, and J. Zhang, *A method for incremental updating approximations based on variable precision set-valued ordered information systems*, in Hu et al. [HLR$^+$10], pp. 96–101.

[CLZ10b] ———, *A method for incremental updating approximations when objects and attributes vary with time*, in Hu et al. [HLR$^+$10], pp. 90–95.

[DH07] D. Deng and H. Huang, *Dynamic reduction based on rough sets in incomplete decision systems*, Rough Sets and Knowledge Technology (JingTao Yao, Pawan Lingras, Wei-Zhi Wu, Marcin Szczuka, Nick Cercone, and Dominik Slezak, eds.), LNAI, vol. 4481, 2007, pp. 76–83.

[FTCH09] Y. Fan, T. Tseng, C. Chern, and C. Huang, *Rule induction based on an incremental rough set*, Expert Systems with Applications **36** (2009), no. 9, 11439–11450.

[GBGB08] J. Grzymala-Busse and W. Grzymala-Busse, *Inducing better rule sets by adding missing attribute values*, Proceedings RSCTC08 (C.C. Chan, J. Grzymala-Busse, and W. Ziarko, eds.), LNAI, vol. 5306, 2008, pp. 160–169.

[HLR$^+$10] X. Hu, T. Y. Lin, V. V. Raghavan, J. W. Grzymala-Busse, Qing Liu, and Andrei Z. Broder (eds.), *2010 IEEE International Conference on Granular Computing, GrC 2010, San Jose, California, USA, 14-16 august 2010*, IEEE Computer Society, 2010.

[Kry98] M. Kryszkiewicz, *Rough set approach to incomplete information systems*, Information Sciences **112** (1998), 39–49.

[LLRZ09] D. Liu, T. Li, Da Ruan, and W. Zou, *An incremental approach for inducing knowledge from dynamic information systems*, Fundamenta Informaticae **94** (2009), 245–260.

[LLRZ11] Dun Liu, Tian-Rui Li, Da Ruan, and Junbo Zhang, *Incremental learning optimization on knowledge discovery in dynamic business intelligent systems*, J. Global Optimization **51** (2011), no. 2, 325–344.

[LRW$^+$07] T. Li, D. Ruan, G. Wets, J. Song, and Y. Xu, *A rough sets based characteristic relation approach for dynamic attribute generalization in data mining*, Knowledge-Based Systems **20** (2007), no. 5, 485–494.

[MSS08] M. J. Moshkov, A. Skowron, and Z. Suraj, *Extracting relevant information about reduct sets from data tables*, Transactions on Rough Sets **9** (2008), 200–211.

[Orl82] Ewa Orlowska, *Dynamic information systems*, Fundamenta Informaticae **5** (1982), 101–118.

[Pag04] Piero Pagliani, *Pretopologies and dynamic spaces*, Fundamenta Informaticae **59** (2004), no. 2-3, 221–239.

[Paw81] Z. Pawlak, *Information systems - theoretical foundations*, Information Systems **6** (1981), 205–218.

[PS07a] Z. Pawlak and A. Skowron, *Rough sets and boolean reasoning*, Information Sciences **177** (2007), 41–73.

[PS07b] ———, *Rudiments of rough sets*, Information Sciences **177** (2007), 3–27.

[SPS05] R. W. Swiniarski, K. Pancerz, and Z. Suraj, *Prediction of model changes of concurrent systems described by temporal information systems*, Proceedings of The 2005 International Conference on Data Mining, DMIN 2005, Las Vegas, Nevada, USA, June 20-23, 2005 (Hamid R. Arabnia and Anthony Scime, eds.), CSREA Press, 2005, pp. 51–57.

[SS96] A. Skowron and J. Stepaniuk, *Tolerance approximation spaces*, Fundamenta Informaticae **27** (1996), 245–253.

[SZ95] N. Shan and W. Ziarko, *Data-based acquisition and incremental modification of classification*, Computational Intelligence **11** (1995), 357—370.

[Woj01] A. Wojna, *Constraint based incremental learning of classification rules*, Rough Sets and Current Trends in Computing 2000 (Wojciech Ziarko and Y. Y. Yao, eds.), LNCS, vol. 2005, 2001, pp. 428–435.

[Yao98] Y.Y. Yao, *Relational interpretations of neighborhood operators and rough set approximation operators*, Information Sciences **111** (1998), 239–259.

[ZLCJ09] W. Zou, T. Li, H. Chen, and X. Ji, *Approaches for incrementally updating approximations based on set-valued information systems while attribute values' coarsening and refining*, GrC 2009 Proceedings, 2009, pp. 824–829.

[ZW04] Z. Zheng and G. Wang, *RRIA: A rough set and rule tree based incremental knowledge acquisition algorithm*, Fundamenta Informaticae **59** (2004), no. 2-3, 299–313.

Dipartimento di Informatica, Sistemistica e Comunicazione, Università di Milano–Bicocca, Via Bicocca degli Arcimboldi 8, I–20126 Milano (Italy)

E-mail address: ciucci@disco.unimib.it