Division of Informatics, University of Edinburgh

Centre for Intelligent Systems and their Applications

A Modular Approach to Generating Fuzzy Rules with Reduced Attributes for the Monitoring of Complex Systems

by

Qiang Shen, Alexios Chouchoulas

Informatics Research Report EDI-INF-RR-0122

Division of Informatics **May 2000**
http://www.informatics.ed.ac.uk/

A Modular Approach to Generating Fuzzy Rules with Reduced Attributes for the Monitoring of Complex Systems

Qiang Shen, Alexios Chouchoulas

Informatics Research Report EDI-INF-RR-0122

DIVISION *of* INFORMATICS
Centre for Intelligent Systems and their Applications

May 2000

Presented in: Engineering Applications of Artificial Intelligence, 13(3):263-278

Abstract :

The building of intelligent monitoring and diagnostic systems for complex industrial domains tends to be hindered by the knowledge-acquisition bottleneck. Creating good knowledge bases for such tasks is notoriously difficult, especially where human experts are not readily available. High dimensionality of the domain attributes presents a further obstacle for a number of rule-induction algorithms which would, otherwise, have the potential for automating knowledge acquisition. This paper attempts to tackle both problems, by proposing a highly modular framework for data-driven fuzzy ruleset induction incorporating a dimensionality-reduction step based on rough set theory. This removes redundant and information-poor attributes from the data, thereby significantly increasing the speed of the induction algorithm, which is employed to generalise historic data into fuzzy association rules. The aid of dimensionality reduction extends past the training stage of the system into its runtime. By removing information-poor attributes, the implemented system is kept simple by requiring fewer connections to physical instrumentation, while the system's response times are increased. The paper introduces the techniques jointly forming the proposed framework, and demonstrates the applicability of the approach by building a monitoring system for an urban water treatment plant. The results of this application are presented and discussed, and comparisons to alternative approaches are given.

Keywords : systems monitoring, fuzzy rule induction, rough dimensionality, reduction, knowledge acquisition, knowledge-based systems

Copyright © 2002 by The University of Edinburgh. All Rights Reserved

The authors and the University of Edinburgh retain the right to reproduce and publish this paper for non-commercial purposes.

Permission is granted for this report to be reproduced by others for non-commercial purposes as long as this copyright notice is reprinted in full in any reproduction. Applications to make other use of the material should be addressed in the first instance to Copyright Permissions, Division of Informatics, The University of Edinburgh, 80 South Bridge, Edinburgh EH1 1HN, Scotland.

A Modular Approach to Generating Fuzzy Rules with Reduced Attributes for the Monitoring of Complex Systems

Qiang Shen [a,1] Alexios Chouchoulas [a,2]

[a] *School of Artificial Intelligence, Division of Informatics, University of Edinburgh, 80 South Bridge, Edinburgh EH1 1HN, UK*

Abstract

The building of intelligent monitoring and diagnostic systems for complex industrial domains tends to be hindered by the knowledge-acquisition bottleneck. Creating good knowledge bases for such tasks is notoriously difficult, especially where human experts are not readily available. High dimensionality of the domain attributes presents a further obstacle for a number of rule-induction algorithms which would, otherwise, have the potential for automating knowledge acquisition. This paper attempts to tackle both problems, by proposing a highly modular framework for data-driven fuzzy ruleset induction incorporating a dimensionality-reduction step based on rough set theory. This removes redundant and information-poor attributes from the data, thereby significantly increasing the speed of the induction algorithm, which is employed to generalise historic data into fuzzy association rules. The aid of dimensionality reduction extends past the training stage of the system into its runtime. By removing information-poor attributes, the implemented system is kept simple by requiring fewer connections to physical instrumentation, while the system's response times are increased. The paper introduces the techniques jointly forming the proposed framework, and demonstrates the applicability of the approach by building a monitoring system for an urban water treatment plant. The results of this application are presented and discussed, and comparisons to alternative approaches are given.

Key words: systems monitoring; fuzzy rule induction; rough dimensionality reduction; knowledge acquisition; knowledge-based systems

[1] qiangs@dai.ed.ac.uk
[2] alexios@dai.ed.ac.uk

1 Introduction

The ever-increasing demand for dependable, trustworthy intelligent diagnostic and monitoring systems, as well as knowledge-based systems in general, has focused the attention of researchers on the knowledge-acquisition bottleneck. The task of gathering information and extracting general knowledge from it is well-known to be the most difficult part of creating a knowledge-based system.

Conditions usually deteriorate when the task becomes more complex, for instance in monitoring of a large industrial plant. The data collected for such cases is likely to present a large number of attributes, not all of which will be essential in determining the operational status. Also, unknown, unavailable or inaccurate values cannot be ruled out, as instruments inevitably develop faults of their own, and the humans reading them are not infallible. The absence of experts to interpret the data often proves to be another problem. A technique to allow the formulation of knowledge from historical data into machine-usable forms is clearly highly desirable.

This paper proposes a methodology to ease the task of knowledge gathering in an automatic, efficient and domain-independent way. The aim of the present work is the creation of a low-dimensionality ruleset from historical data provided in a domain. The paper presents a highly modular approach to data-driven fuzzy ruleset generation. In particular, Lozowski's rule induction algorithm (RIA), as presented in (Lozowski et al., 1996), is chosen to act as the starting point for this, although the flexibility of the system discussed here allows the incorporation of almost any rule-induction algorithm. The current RIA, provided with datasets of continuous values, induces classification rules to partition the data into underlying categories. It is chosen for features such as the graceful handling of missing and inaccurate information or vague data, domain independence, or incremental operation (Chouchoulas and Shen, 1998). However, as with many RIAs, this algorithm exhibits high computational complexity due to its generate-and-test nature. The effects of this become evident where data of high dimensionality needs to be processed.

Thus, in order to remove redundancy from the input data, as well as to speed up the RIA, a preprocessing step is required. This step, implemented here using rough set theory (Pawlak, 1991), reduces the dimensionality of potentially very large datasets without losing information that is needed for rule induction, insofar as this is possible in the domain at hand. It also helps to simplify the design and implementation of the actual monitoring or diagnostic tool itself, by determining what measurements should be made available to the system. In addition, the reduced input dimensionality increases the processing speed of the monitoring system, leading to better response times in case of emergency. The reduction of input dimensionality offers a further advantage,

in that fewer connections between the diagnostic hardware and the monitoring instrumentation are required, which reduces the number of possible fault points.

The ultimate goal of this work, producing a knowledge-rich ruleset for a problem domain is very similar to existing, established machine-learning mechanisms such as Quinlan's C4.5 algorithm (Quinlan, 1993). However, there are key differences between the present framework and C4.5-like methods. C4.5, because of the hierarchical structure of its output decision rules, tends to involve more attributes than the rough-set-based technique. Although the dimensionality of individual rules learned by C4.5 is not necessarily higher than that of the rules induced by the proposed methodology, empirical results show that a decision tree created by C4.5 may require many more input attributes overall. This means that an implemented monitoring system trained with C4.5 might need access to more instrumentation, making it more sensitive to instrumentation faults, and reducing its efficiency in real-time fault detection. For many engineering systems, minimising the number of observables (from which measurements are made) is itself an important requirement, since measurements can be very difficult and costly to obtain. Furthermore, the RIA used here allows an 'undecidable' answer, through which it declares its ignorance as to a classification. The algorithm tends to give this answer rather than a wrong one, alerting its users to unusual borderline conditions. C4.5 always yields a classification, which would thus need frequent checking for correctness by the user. Another distinction of the present framework is its modularity, which offers a significant flexibility in utilising different rule-induction algorithms to generate different types of (fuzzy) rulesets. For example, instead of using Lozowski's RIA, the algorithm reported by Holve (1998) may be used. This would allow the generation of rules that are decomposed into a hierarchy of sub-rulesets, so that inference may be performed more efficiently. This may help in further alleviation of the high-dimensionality bottleneck.

The rest of the paper is arranged as follows. It starts by discussing the theoretical background of the currently used RIA and dimensionality-reduction algorithms in Section 2. To illustrate the ideas behind these algorithms, worked examples are included. Section 3 then describes in detail a problem case, which both justifies the need for the present work and sets up the scene for the experimental investigations reported. Section 4 describes the proposed framework, integrating dimensionality reduction and rule induction, and discusses the important design and implementation issues involved. This is then followed by the presentation and detailed discussion of experimental results in Section 5, supported by a comparison to the application of C4.5 to the same problem. Section 6 concludes the paper, and proposes further work.

Table 1
Example dataset for the XOR function.

#	A	B	Y	#	A	B	Y
1	-0.4	-0.3	no	9	0.1	-0.3	yes
2	-0.4	0.2	yes	10	0.1	0.3	no
3	-0.3	-0.4	no	11	0.2	-0.1	yes
4	-0.3	0.3	yes	12	0.2	0.4	no
5	-0.2	-0.2	no	13	0.3	-0.4	yes
6	-0.2	0.1	yes	14	0.3	0.2	no
7	-0.1	-0.1	no	15	0.4	-0.2	yes
8	-0.1	0.4	yes	16	0.4	0.1	no

2 Background

2.1 Lozowski's rule-induction algorithm

Lozowski et al. (1996) proposed a technique for data-driven rule induction which aims at extracting linguistically expressed rules from real-valued examples. Although this RIA was proposed to be used in conjunction with neural-network-based classifiers, it appears to be independent of the actual type of classifier used.

Provided with training data, this method may be used to induce approximate relationships between the characteristics of the conditional attributes and their decision attributes or classes. The premise attributes of the induced rules are represented by fuzzy variables (Zadeh, 1965; Cox, 1994), facilitating the modelling of the inherent uncertainty of the knowledge domain.

A simple example helps to explain the operation of this algorithm. The chosen example, drawn from the slides accompanying (Lozowski et al., 1996), is a dataset extracted by feeding various pairs of values to a neural network trained on the exclusive OR function (XOR). Given two inputs A and B, the neural network essentially calculates $Y = -AB$. A dataset of sixteen input values and the neural network's output can be found in Table 1. The output of the function approximator is converted to a nominal, linguistic value by letting positive values stand for 'yes' and negative ones stand for 'no'.

An additional input item needed by the algorithm is the fuzzification of the conditional attributes. In this case, both A and B are fuzzified using the same three sets: N for negative, Z for around zero and P for positive. This is shown in Fig. 1.

The algorithm goes through the following stages, as shown in Fig. 2: finding decision regions, generating attribute vectors, evaluating T-norm sets, creating S-norms and, finally, creating rules. These steps are briefly outlined below.

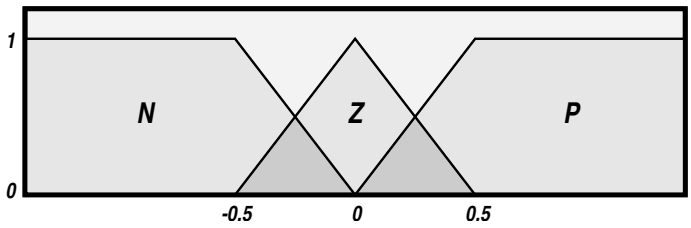

Fig. 1. Fuzzification of both XOR inputs.

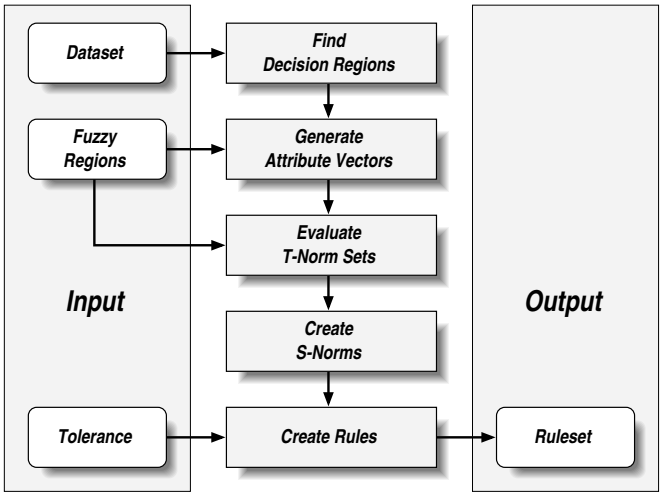

Fig. 2. A block diagram of the RIA, showing the algorithm's steps, along with its input and output.

2.1.1 Locate the decision regions.

The first step is to locate the *decision regions*. A decision region is a set comprising the row numbers of dataset objects for which a certain decision attribute has a certain value:

$$D_{y_i=c} = \left\{ x : \;\; y_{xi} = c \;\; \wedge \;\; x \;\; \in \;\; \{1, \ldots, k\} \right\} \tag{1}$$

where y_i is a decision attribute, c is the chosen value of that attribute, y_{xi} is the value of the decision attribute y_i as given by the object with number x; k is the number of objects in the dataset. (Note that this paper uses row numbers; Lozowski et al. use whole dataset objects. The terms *object* and *example* are used interchangeably.)

This step creates one decision region for each decision attribute and each value for that attribute that is present in the dataset, with the exception of the 'unknown' value. In the context of the XOR example, the following two decision regions are constructed:

$$D_{Y=\mathrm{no}} = \{1, 3, 5, 7, 10, 12, 14, 16\}$$

		N		**Z**		**P**	
P		0.0	0.0	0.0	0.2	*0.0*	*0.2*
		0.0	0.0	0.0	0.2	*0.0*	*0.4*
		0.0	0.0	0.0	0.6	*0.0*	*0.4*
		0.0	0.0	0.0	0.8	*0.0*	*0.2*
Z		0.2	0.0	*0.2*	*0.4*	0.0	0.6
		0.4	0.0	*0.2*	*0.2*	0.0	0.6
		0.4	0.0	*0.6*	*0.4*	0.0	0.4
		0.2	0.0	*0.6*	*0.2*	0.0	0.2
N		*0.6*	*0.0*	0.4	0.0	0.0	0.0
		0.6	*0.0*	0.2	0.0	0.0	0.0
		0.4	*0.0*	0.4	0.0	0.0	0.0
		0.2	*0.0*	0.2	0.0	0.0	0.0

B Y=no

		N		**Z**		**P**	
P		*0.0*	*0.2*	*0.0*	*0.2*	0.0	0.0
		0.0	*0.2*	*0.0*	*0.4*	0.0	0.0
		0.0	*0.6*	*0.0*	*0.2*	0.0	0.0
		0.0	*0.4*	*0.0*	*0.6*	0.0	0.0
Z		*0.0*	*0.6*	0.2	0.4	*0.2*	*0.0*
		0.0	*0.2*	0.4	0.6	*0.4*	*0.0*
		0.0	*0.4*	0.6	0.2	*0.2*	*0.0*
		0.0	*0.2*	0.2	0.2	*0.8*	*0.0*
N		0.0	0.0	*0.6*	*0.0*	*0.4*	*0.0*
		0.0	0.0	*0.4*	*0.0*	*0.6*	*0.0*
		0.0	0.0	*0.4*	*0.0*	*0.2*	*0.0*
		0.0	0.0	*0.2*	*0.0*	*0.2*	*0.0*

B Y=yes

A

Fig. 3. Sets of T-norms for all XOR combinations of conditional attributes and both decisions shown as a pair of tables. Each table cell is a set of T-norms. T-norm sets in italic represent values of A and B consistent with the definition of XOR.

$$D_{Y=\text{yes}} = \{2, 4, 6, 8, 9, 11, 13, 15\}.$$

2.1.2 Generate all the possible fuzzy set vectors.

This step generates a set of all the possible combinations of fuzzy sets defined in the underlying ranges of the n conditional attributes. Each of the n attributes is a fuzzy region divided up into a number of fuzzy sets. In the example, there are $n = 2$ conditional attributes, each fuzzified with a region which contains three fuzzy sets. This step would hence produce 3×3 combinations: $\{\langle N, N\rangle, \langle N, Z\rangle, \langle N, P\rangle, \ldots, \langle P, P\rangle\}$.

2.1.3 Evaluate the sets of T-norms.

A T-norm of an example $\underline{x}$ with conditional attributes $x_1, x_2, \ldots, x_n$ with respect to a fuzzy set vector $\underline{\pi} = \langle \mu_1, \mu_2, \ldots, \mu_n \rangle$ (as calculated in the previous step) is defined as:

$$T^{\underline{\pi}}(\underline{x}) = \min\Big(\mu_1(x_1),\, \mu_2(x_2),\, \ldots,\, \mu_n(x_n)\Big). \tag{2}$$

This step involves the evaluation of sets of T-norms. A set of T-norms is defined in terms of a decision $y_i = c$ and a fuzzy set vector $\underline{\pi}$ as follows:

	N	Z	P
P	0.0	0.8	*0.4*
Z	0.4	*0.8*	0.6
N	*0.6*	0.4	0.0

A **B** **Y=no**

	N	Z	P
P	*0.6*	*0.6*	0.0
Z	*0.6*	0.6	*0.8*
N	0.0	*0.6*	*0.6*

A **B** **Y=yes**

Fig. 4. S-norms for the XOR example. Italics show the combinations that are consistent with the definition of XOR.

$$\mathbb{T}^{\underline{\pi}}_{y_i=c} = \Big\{ w : \quad w = T^{\underline{\pi}}(\underline{x_j}) \; \forall j \in D_{y_i=c} \Big\}. \tag{3}$$

In a typical fuzzy region, no more than two fuzzy sets overlap for each underlying real element, while the region is split into two or more fuzzy sets. This causes a lot of memberships to evaluate to zero, in turn forcing $T^{\underline{\pi}}(\underline{x})$ also to evaluate to zero, due to the use of $\min(\cdot)$. Thus, most of the T-norms will be equal to zero. Due to the small size of the example, it is possible to show all individual T-norms of dataset examples (please see Fig. 3).

2.1.4 Create S-norms.

An S-norm $S^{\underline{\pi}}_{y_i=c}$ of a decision $y_i = c$ with respect to a fuzzy set vector $\underline{\pi}$ is defined as the maximum element of a set of T-norms $\mathbb{T}^{\underline{\pi}}_{y_i=c}$:

$$S^{\underline{\pi}}_{y_i=c} = \max\{x : \quad x \in \mathbb{T}^{\underline{\pi}}_{y_i=c}\} \tag{4}$$

This results in a multidimensional array of candidate rules describing the mapping between conditional and decision attributes. There is one array for each pair of decision attributes and decision attribute values (i.e., one for each decision region, as calculated above). The resultant S-norms for the XOR example are shown in Fig. 4.

2.1.5 Create rules.

It is now possible to generate actual fuzzy rules from the arrays of S-norms. Any S-norm can be promoted to a rule. However, there is an S-norm for each possible decision $y_i = c$ and every possible combination of fuzzified conditional attributes. This implies that the candidate ruleset may include contradicting rules. A means of deciding which S-norm best describes a given fuzzy set vector is needed.

A constant ε dubbed the *uncertainty margin* (or *tolerance*) is used for this. A fuzzy rule concludes that class y_i has value c, given attribute values matching

Table 2
XOR ruleset induced for different values of ε.

$0.0 \leq \varepsilon \leq 0.2$		
A	B	Y
N	N	no
N	Z	yes
N	P	yes
Z	N	yes
Z	Z	no
P	Z	no
P	N	yes
Z	P	yes
P	P	no

$0.2 \leq \varepsilon \leq 0.4$		
A	B	Y
N	N	no
N	P	yes
P	N	yes
P	P	no

$0.4 \leq \varepsilon \leq 0.6$		
A	B	Y
N	N	no
P	N	yes
N	P	yes

the fuzzy premises in $\underline{\pi}$, if and only if the corresponding S-norm is equal to at least ε more than the S-norm (or S-norms) supporting classifications $y_i = c'$, where $c' \in \mathbb{Y}_i$ (the set of possible values for decision class y_i) and $c' \neq c$.

$$y_i = \begin{cases} c & \text{if} \quad \left(\forall c' \in \mathbb{Y}_i - \{c\}\right) \Rightarrow S^{\underline{\pi}}_{y_i=c} - S^{\underline{\pi}}_{y_i=c'} > \varepsilon, \\ - & \text{otherwise} \end{cases} \tag{5}$$

In other words, in order to promote an S-norm to a rule, all other contradictory S-norms must have values at least ε less than the S-norm in question. In Fig. 4, contradictory S-norms are those in corresponding table cells of the two tables for decisions $Y =$ yes and $Y =$ no.

If no S-norm can be decided upon (that is, all S-norm values are within the ε neighbourhood), the whole rule is undecided for the fuzzy set vector in question.

The uncertainty margin introduces a trade-off to the rule-generation step. The higher ε is, the fewer rules are generated. The accuracy of the generated rules in performing a certain application task is, of course, expected to be affected by changing ε. A careful selection of an appropriate ε value is therefore needed for a given problem at hand. Please refer to Table 2 for three rulesets generated for the XOR example, given different values of ε.

2.2 Rough-set-aided attribute reduction

Rough set theory (Pawlak, 1991) is a formal mathematical tool that can be applied to reducing the dimensionality of datasets. The rough set attribute reduction (RSAR) method removes redundant input attributes from datasets of discrete values, all the while making sure that no information is lost. The

Table 3
An example dataset.

$x \in \mathbb{U}$	a	b	c	d	$\Rightarrow$	e
0	1	0	2	2		0
1	0	1	1	1		2
2	2	0	0	1		1
3	1	1	0	2		2
4	1	0	2	0		1
5	2	2	0	1		1
6	2	1	1	1		2
7	0	1	1	0		1

approach is fast and efficient, making use of standard operations from conventional set theory.

To demonstrate the RSAR algorithm, an example will be followed through. Suppose that a dataset $\mathbb{D}$ is viewed as a table, where attributes are columns and objects are rows, as in Table 3 (adapted from Pawlak, 1991)). Let $\mathbb{U}$ denote the set of all objects in the dataset, $\mathbb{A}$ the set of all attributes, $\mathbb{C}$ the set of conditional (or input) attributes, and $\mathbb{D}$ the set of decision attributes. Thus, in this example, $\mathbb{U} = \{0, 1, 2, 3, 4, 5, 6, 7\}$, $\mathbb{A} = \{a, b, c, d, e\}$, $\mathbb{C} = \{a, b, c, d\}$ and $\mathbb{D} = \{e\}$.

The value of attribute $q \in \mathbb{A}$ in object $x \in \mathbb{U}$ is written as $f(x, q)$, which defines an equivalence relationship over $\mathbb{U}$. With respect to a given q, the function partitions the universe into a set of pairwise disjoint subsets of $\mathbb{U}$:

$$R_q = \{x : x \in \mathbb{U} \ \ \wedge \ \ f(x, q) = f(x_0, q) \ \ \forall \ \ x_0 \in \mathbb{U}\}. \tag{6}$$

For instance, drawing from the example discussed above:

$$R_a = \Big\{\{1, 7\}, \{0, 3, 4\}, \{2, 5, 6\}\Big\}$$
$$R_b = \Big\{\{0, 2, 4\}, \{1, 3, 6, 7\}, \{5\}\Big\}.$$

Assume a subset of the set of attributes, $P \subset \mathbb{A}$. Two objects numbered x and y in $\mathbb{U}$ are *indiscernible* with respect to P if and only if $f(x, q) = f(y, q) \ \ \forall \ \ q \in P$. The indiscernibility relation for all $P \in \mathbb{A}$ is written as $\text{IND}(P)$. $\text{U}/\text{IND}(P)$ is used to denote the partition of $\mathbb{U}$ given $\text{IND}(P)$ and is calculated as:

$$\text{U}/\text{IND}(P) = \bigotimes \Big\{q \in P : \text{U}/\text{IND}(q)\Big\}, \text{ where}$$
$$A \otimes B = \{X \cap Y : \forall \ \ X \in A, \ \forall \ \ Y \in B, \ X \cap Y \neq \emptyset\}. \tag{7}$$

For instance, if $P = \{b, c\}$, objects 0 and 4 are indiscernible; 1, 6 and 7 likewise. The rest of the objects are not. This applies to the example dataset as follows:

$$\begin{aligned}
\mathrm{U/IND}(P) &= \mathrm{U/IND}(b) \otimes \mathrm{U/IND}(c) = \qquad (8)\\
&= \Big\{\{0,2,4\},\{1,3,6,7\},\{5\}\Big\} \otimes \Big\{\{2,3,5\},\{1,6,7\},\{0,4\}\Big\} = \\
&= \Big\{\{0,2,4\} \cap \{2,3,5\}, \{0,2,4\} \cap \{1,6,7\}, \{0,2,4\} \cap \{0,4\}, \\
&\quad \cdots \\
&\quad \{5\} \cap \{0,4\}\Big\} = \\
&= \Big\{\{2\},\{0,4\},\{3\},\{1,6,7\},\{5\}\Big\}. \qquad (9)
\end{aligned}$$

If $P = \{a, b, c\}$, then, similarly:

$$\mathrm{U/IND}(P) = \mathrm{U/IND}(a) \otimes \mathrm{U/IND}(b) \otimes \mathrm{U/IND}(c). \qquad (10)$$

A rough set approximates traditional sets by using a pair of sets, named the *lower* and *upper approximations* of the set in question. The lower and upper approximations of a set $P \subseteq \mathbb{U}$ (given an equivalence relation $\mathrm{IND}(P)$ are defined as:

$$\underline{P}Y = \bigcup\{X : X \in \mathrm{U/IND}(P), X \subseteq Y\} \qquad (11)$$

$$\overline{P}Y = \bigcup\{X : X \in \mathrm{U/IND}(P), X \cap Y \neq \emptyset\}. \qquad (12)$$

Assuming that P and Q are equivalence relations in $\mathbb{U}$, the *positive region* $\mathrm{POS}_P(Q)$ is defined as:

$$\mathrm{POS}_P(Q) = \bigcup_{X \in Q} \underline{P}X. \qquad (13)$$

A positive region contains all the objects in $\mathbb{U}$ that can be classified into attribute set Q using the information in attribute set P. For example, assuming $P = \{b, c\}$ and $Q = \{e\}$:

$$\mathrm{POS}_{\mathrm{IND}(P)}\Big(\mathrm{IND}(Q)\Big) = \bigcup\Big\{\{\},\{2,5\},\{3\}\Big\} = \{2,3,5\}. \qquad (14)$$

What this means is that, with respect to conditional attributes b and c, objects 2, 3 and 5 can definitely be classified in terms of decision attribute e. The

Table 4
Objects 2, 3 and 5 are discernible and can definitely be classified into e using the selected conditional attributes b and c. The rest of the objects cannot be classified unambiguously.

$x \in \mathbb{U}$	b	c	$\Rightarrow$	e
0	0	2		0
1	1	1		2
2	0	0		1
3	1	0		2
4	0	2		1
5	2	0		1
6	1	1		2
7	1	1		1

remaining objects could, possibly, be classified, but it is not certain. This is shown more intuitively in Table 4.

The *degree of dependency* of a set Q of decision attributes on a set of conditional attributes P is defined as (Pawlak, 1991, p. 47):

$$\gamma_P(Q) = \frac{\parallel \mathrm{POS}_P(Q) \parallel}{\parallel \mathbb{U} \parallel}, \tag{15}$$

where $\parallel A \parallel$ denotes the cardinality of set A. The complement of γ gives a measure of the contradictions in the selected subset of the dataset. If $\gamma = 0$, there is no dependence; for $0 < \gamma < 1$, there is a partial dependence. If $\gamma = 1$, there is complete dependence.

It is now possible to define the *significance* of an attribute. This is done by calculating the change of dependency when removing the attribute from the set of considered conditional attributes. Given P, Q and an attribute $x \in P$:

$$\sigma_P(Q, x) = \gamma_P(Q) - \gamma_{P-\{x\}}(Q). \tag{16}$$

The higher the change in dependency, the more significant x is. This allows the calculation of the significance of any conditional attribute, for instance a:

$$\sigma_P(Q, a) \quad = \quad \gamma_{\{a,b,c\}}(\{e\}) - \gamma_{\{b,c\}}(\{e\}) \quad = \quad \tfrac{1}{8}.$$

This shows that attribute a is not indispensable, having a significance of 0.125, while attributes b and c can be dispensed with, as they do not provide any information that is significant for the classification of the data objects into the class values in e.

Attribute reduction involves the removal of attributes that have no significance to the classification at hand. An *attribute reduct set* (or simply *reduct*) is then defined as a subset R of the set of conditional attributes $\mathbb{C}$ such that $\gamma_{\mathbb{C}}(\mathbb{D}) = \gamma_R(\mathbb{D})$. For the set of decision attributes $\mathbb{D}$, it is obvious that a dataset may have more than one attribute reduct set. The set $\mathcal{R}$ of all attribute reduct sets R is defined as:

$$\mathcal{R} = \Big\{X : X \subseteq \mathbb{C}, \;\; \gamma_{\mathbb{C}}(\mathbb{D}) = \gamma_X(\mathbb{D})\Big\}. \tag{17}$$

The RSAR will not compromise with a set of conditional attributes that has a large part of the information embedded in the initial conditional attribute set, $\mathbb{C}$ — it *always* attempts to reduce the attribute set while losing *no* information that is significant to the classification at hand. RSAR searches for the attribute reduct sets of least cardinality. That is, it seeks one or more elements in the set of *minimal reducts* $\mathcal{R}_{min} \subseteq \mathcal{R}$:

$$\mathcal{R}_{min} = \Big\{X : X \in \mathcal{R}, \;\; \forall\, Y \in \mathcal{R}, \;\; \parallel X \parallel \;\leq\; \parallel Y \parallel \Big\}. \tag{18}$$

In terms of computational complexity and memory requirements, the calculation of all possible subsets of a given set is an NP-hard task. To solve this problem, the reduct subset search space is treated as a tree traversal. Each node of the tree represents the addition of one conditional attribute to an initially empty reduct. Instead of generating the whole tree and picking the best path on it, the path is chosen progressively, using the following heuristic: the next attribute chosen to be added to the reduct is the attribute that adds the most to the reduct's dependency. The search ends when the dependency reaches one, or when no more attributes are left. This is dubbed the QuickReduct, and is similar to the algorithm introduced in (Jelonek et al., 1995), where it was proposed in conjunction with neural-network-based classifiers. Please refer to Fig. 5 for a pseudocode version of the algorithm.

Returning to the example, it is now possible to show the workings of QuickReduct. The reduct R starts off as the empty set (step 1). For each of the attributes, γ is calculated with respect to the decision attribute e:

$$\gamma_{\{\}\cup\{a\}}(e) = \frac{0}{8} \qquad \gamma_{\{\}\cup\{b\}}(e) = \frac{1}{8} \qquad \gamma_{\{\}\cup\{c\}}(e) = \frac{0}{8} \qquad \gamma_{\{\}\cup\{d\}}(e) = \frac{2}{8}.$$

This shows that attributes a and c are not of much use on their own. Attribute d contributes the most information, allowing the classification of two of the eight examples. QuickReduct hence adds d to the reduct (steps 6 and 7) and

attempts to evaluate the addition of a second attribute from those remaining (a, b, c):

$$\gamma_{\{d\}\cup\{a\}}(e) = \frac{3}{8} \qquad \gamma_{\{d\}\cup\{b\}}(e) = \frac{8}{8} \qquad \gamma_{\{d\}\cup\{c\}}(e) = \frac{8}{8}.$$

The addition of either attribute b or c to the reduct (currently $\{d\}$) allows perfect classification of the data. The first such attribute, b (because of step 5) is added to the reduct (in steps 6 and 7).

RSAR offers a number of advantages. It pre-processes datasets without altering the attribute values themselves, thus maintaining the semantics. It is not a lossy algorithm; it will remove a conditional attribute from the dataset only if this action removes absolutely no information (with respect to the classification at hand). Unlike statistical correlation-reducing approaches like the Principal Components Analysis (PCA), discussed among other places in (Haykin, 1994), the dimensionality reduction does not require human input, or the setting of variance thresholds. This same feature is also a disadvantage, since other techniques offer more aggressive dimensionality reduction, accepting that in some cases the loss of a little information may in fact prove to be advantageous (e.g., in noisy environments). Another disadvantage of the RSAR algorithm is its lacking efficiency. This is successfully rectified by the QuickReduct algorithm which converts an exhaustive evaluation of all attribute combinations into a best-first tree search. In terms of input domain, RSAR is mainly intended for discrete domains, although it can be adapted to cope with continuous ones, as described in Section 4.

Algorithm 1 QuickReduct$(\mathbb{C}, \mathbb{D})$
Input: $\mathbb{C}$*, the set of all conditional attributes;* $\mathbb{D}$*, the set of decision attributes.*
Output: R*, the attribute reduct,* $R \subseteq \mathbb{C}$*.*

```
(1)  R ← {}
(2)  do
(3)     T ← R
(4)     foreach  x ∈ (C − R)
(5)        if  γ_{R∪{x}}(D) > γ_T(D)
(6)           T ← R ∪ {x}
(7)     R ← T
(8)  until  γ_R(D) = γ_C(D)
(9)  return  R
```

Fig. 5. The QuickReduct algorithm.

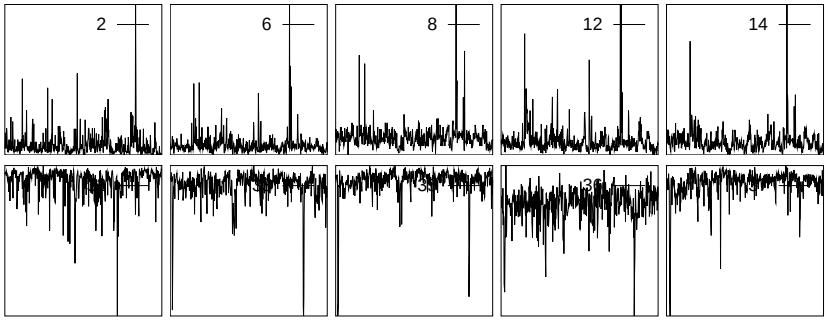

Fig. 6. Two groups of attributes, plotted over time. There is obviously a high degree of redundancy in the dataset.

3 A problem case

The chosen test application was the Water Treatment database (M. Poch et al., 1993), as archived in the UCI Machine Learning Database Repository (University of California, Irnive, 1995). The dataset comprises a set of historical data obtained over a period of 521 days, with one series of measurements per day. Thirty-eight different input attribute values are measured per day, with one set of such measurements forming one datum. There are also missing attribute values in the dataset (due, for instance, to instrument failure or some other event precluding the measurement). All the measurements are real values.

The domain was chosen because of its realism. A large plant is likely to have a similar number of conditional attributes, not all of which will be essential in determining the operational status. Interrelations between attributes are unavoidable as the plant is a single system with interconnections, and unknown values cannot possibly be ruled out.

The goal is to implement a fuzzy reasoning system that, given this dataset of past measurements and without the benefit of an expert in the field at hand, will monitor the plant's status.

The thirty-eight conditional attributes account for the following five aspects of the water treatment plant's operation:

(1) Input to plant (9 attributes)
(2) Input to primary settler (6 attributes)
(3) Input to secondary settler (7 attributes)
(4) Output from plant (7 attributes)
(5) Plant performance (9 attributes).

The status of the plant is represented by classifying each day into one of thirteen different categories, some representing normal operation of varying

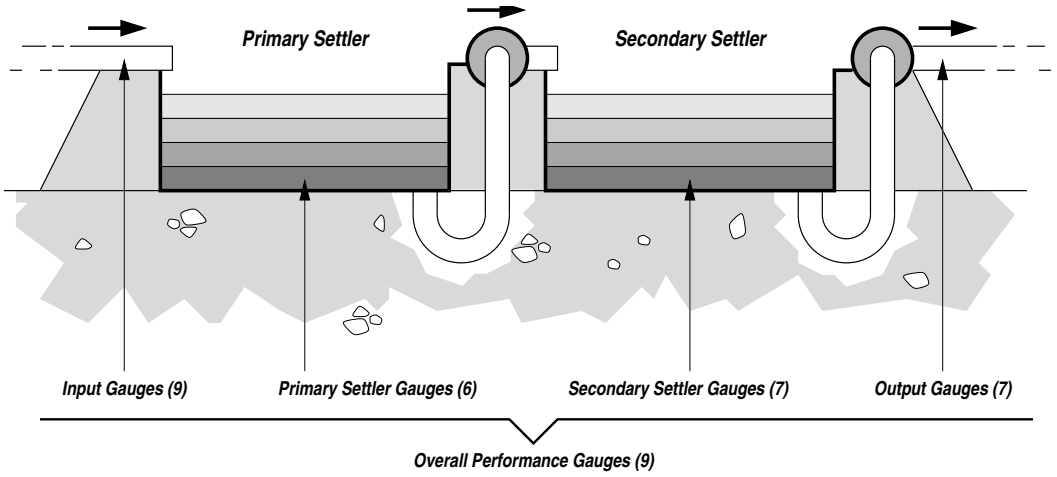

Fig. 7. Schematic diagram of the water treatment plant, indicating the number of measurements sampled at various points.

Table 5
The thirteen possible states of the water treatment plant.

Value	Number	Description
1	275	Normal situation
2	1	Secondary settler problems
3	1	Secondary settler problems
4	4	Secondary settler problems
5	114	Good performance
6	3	Solids overload
7	1	Secondary settler problems
8	1	Storm
9	65	Normal situation, low influent
10	1	Storm
11	53	Normal situation
12	1	Storm
13	1	Solids overload

types, others pointing out faults in various parts of the plant. The categories are shown in Table 5. Certain values are redundant, denoting the same condition in different operational situations. Because of the efficiency of the actual plant from which the measurements were taken, all the faults appear only for short periods (usually single days) and are dealt with immediately. This does not allow for a lot of examples of faults, which is a clear drawback if a monitoring system is to be produced.

In order to reduce redundancies and increase the number of examples of faults, it was necessary to collate most of the fault cases into three major categories, so that each category would be well represented in the dataset. This is shown in Table 6. Additionally, another set of experiments was performed on a further simplified, two-category dataset (*OK* and *Fault*). This was done by collating categories *Normal* and *Good performance* into a new one, *OK*. The result is a

Table 6
Collating the thirteen perceived states of the plant into three states with acceptable representations in the dataset.

New value	Number	Description (old values)
1	393	Normal situation (1, 9, 11)
2	114	Good performance (5)
3	14	Fault (2–4, 6–8, 10, 12, 13)

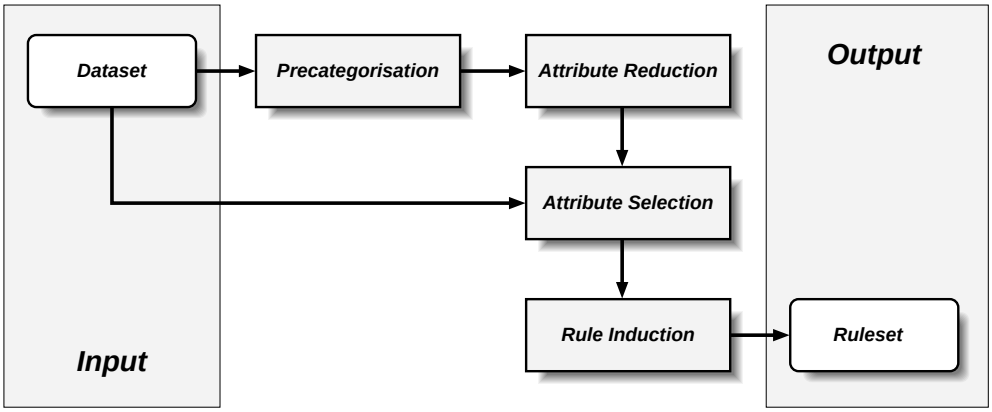

Fig. 8. Block diagram of the dimensionality-reducing rule-induction framework.

purely monitoring system which further demonstrates the applicability of the present work.

4 Combining rough sets with the fuzzy rule induction algorithm

In essence, the combination proposed here deals with large datasets by applying an attribute-reduction algorithm on the dataset to discover a set of attributes that convey all the information with as little redundancy as possible. The desired attributes are then extracted from the dataset and fed to a rule-induction algorithm to generate a suitable ruleset.

The block diagram of the proposed framework is given in Fig. 8. The framework was designed with generalisation as a primary goal. A few small-scale in-house projects have applied it to a number of application domains, with some success.

A particular implementation of this framework is the system that integrates the following modules.

Precategorisation. This reads a dataset and outputs a version in which continuous values have been replaced by discrete ones (please refer to Section 4.1 for details).
Attribute Reduction: the RSAR, as described in section 2.2. Thanks to the

inherent modularity of the approach, any suitable preprocessor may be substituted (precategorisation can be bypassed if the preprocessor is able to deal with continuous values).

Attribute Selection. This is a trivial sub-program that, given a set of attributes (by the attribute reduction module) and a dataset, outputs only the specified attributes from the dataset.

Rule Induction: the RIA. The specific algorithm used in this paper is described in Section 2.1, but any similar algorithm may be used.

Important issues regarding this implementation are discussed below.

4.1 Precategorisation

A means of quantisation is required as a preprocessing stage before running the RSAR, mapping real values of an attribute into ordered discrete symbolic values. The discretisation needed here must split the domain into no fewer discrete values than needed, as this will result in too much loss of information, and hence in rulesets with reduced accuracy. Splicing the domain into more values than necessary will, on the other hand, tax the induction process, and is likely to produce less effective attribute reductions.

The RIA will generate fuzzy rules like: *if* Size *is* Large *and* Width *is* Wide *then* Type *is* One. It does not matter to the RIA what membership a dataset example's *Size* attribute has in the fuzzy set *Large*, as long as its membership is higher by at least ε than the membership in any of the other sets covering the attribute.

Hence, the RIA essentially deals with ranges of values based on which fuzzy set provides the highest membership for a data example. Precategorisation thus takes advantage of the fuzzification information used by the induction algorithm, along with the dataset itself. It partitions the continuous domain into discrete symbolic values (encoded as integers), one for each range in the domain:

$$\mathbb{P} = \Big\{[x_0, x_1),\ [x_1, x_2),\ \ldots,\ [x_{n-1}, x_n)\Big\} \quad \text{so that}$$
$$\forall x,\ x_i \le x < x_{i+1},\ i = 0, 1, \ldots, n-1 \Rightarrow\ \forall k\ \exists j,\ \ \mu_j(x) \ge \mu_k(x), \qquad (19)$$

where $\mathbb{P}$ is the set of all discrete values in ascending order of x, n denotes the number of such ranges and is equal to the cardinality of $\mathbb{P}$, and j and k indicate membership functions defined within the range $[x_i, x_{i+1})$. The process is illustrated in a simpler manner in Fig. 9. The required information on fuzzification may be provided by a human expert, or by a clustering method

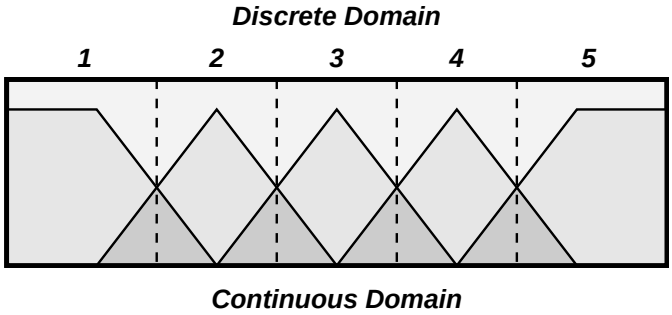

Fig. 9. Precategorisation of a continuous region into five discrete areas, by exploiting fuzzification information.

(Krishnapuram and Keller, 1996).

4.2 Rough set attribute reduction with unknown values

Another problem that needs to be dealt with is the requirement that unknown or missing values be treated properly and consistently by all modules in this system. Unfortunately, the RSAR does not fulfil this requirement, although the design of the RIA does.

The easier method of completely removing dataset objects with unknown values before attribute reduction is unsuitable here: it may also remove other, valuable information from the dataset, thus reducing the accuracy of the attribute reduction. The implemented solution is as follows: if, when building the initial universe partition sets (as described in Section 2.2), an unknown/missing attribute value is encountered, this value is ignored and is not involved in building the partition sets for that particular attribute in the domain. This is of no consequence to the execution of the rest of the algorithm, and no information is lost; single conditional attribute values are ignored, not entire dataset objects.

4.3 RIA tree pruning

The RIA in use is NP-hard, and may become intractable when inducing rules for datasets with many conditional attributes (Chouchoulas and Shen, 1998). The most important problem, in terms of both memory and runtime is dealing with the large numbers of combinations of fuzzy values. This is not so important when only a few attributes are involved, but with an application such as the water treatment plant discussed in Section 3, the algorithm is clearly intractable in terms of both time and space, when used alone.

Although the use of RSAR helps reduce the dimensionality of the dataset, it

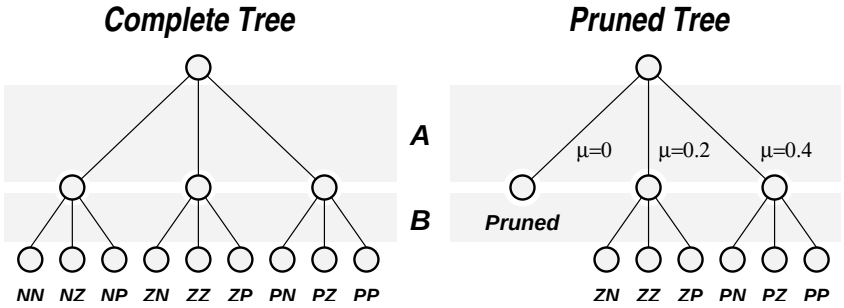

Fig. 10. RIA tree pruning. Arcs represent evaluations of membership functions. Each complete path to a leaf is one combination of fuzzy set vectors.

does not completely remove the potential intractability of the RIA. To further diminish this problem, it is convenient to treat the creation of fuzzy-set vectors as the creation of a tree. In this context, a leaf node is one combination of membership functions, and each arc represents one evaluation of a membership function.

The minimum membership is retained when creating T-norm sets. Any membership function that evaluates to zero means that all leaf nodes in the subtree will eventually evaluate to zero, too, because of the use of $\min(\cdot)$. A subtree is therefore useless and can be ignored (pruned) if its root node evaluates to zero. This is shown in Fig. 10, where the leftmost branch of the right tree represents a membership evaluation yielding zero. The rest of the sub-tree is ignored. In this case, out of nine combinations in the complete tree, only six are used.

The gains with large numbers of conditional attributes are much greater. For the water treatment problem, using thirty-eight conditional attributes and assuming that (for simplicity) each conditional attribute is broken up into five fuzzy sets, the RIA needs to generate 5^{38} combinations. If the pruning algorithm is used instead, and making the reasonable assumption that any value will belong to at most two fuzzy sets, there is a *worst case* of 2^{38} combinations to be evaluated. The time needed is fifteen orders of magnitude less than that needed for the full tree traversal. The savings are significant, but the number of combinations is still far too large — this is the reason for this algorithm's most prominent disadvantage, its intractability. It is also the one of the motivations behind employing RSAR to remove some of the problems of rule induction over large datasets.

Additionally, saving space as well as time is facilitated by further improving the algorithm in various minor ways, essentially merging the generation of combinations, the evaluation of T-norms and the creation of S-norms into one step so that no temporary storage is required for the combinations.

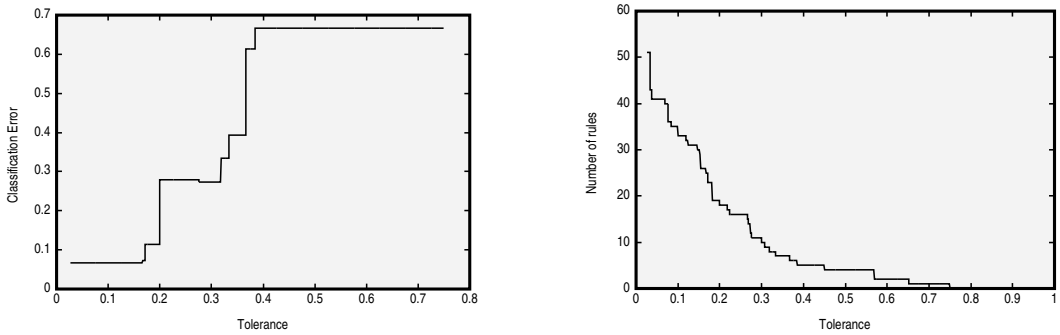

Fig. 11. Canonical algorithm results.

5 Experimental results

5.1 Canonical results

As described in Section 2.1, there is a trade-off between the number of generated rules and the classification error as decided by ε, the uncertainty margin or tolerance. To have an efficient runtime performance, the induced ruleset should be generated so as to be as small as possible (without sacrificing much classification accuracy). This makes it important to have a right choice for ε. However, the selection of ε is, in general, an application-specific task. A good choice for the uncertainty margin that provides a balance between a resultant ruleset's size and accuracy can be found by experiment. Results provided throughout this section demonstrate this.

The results of a rule induction run, to be given below, are shown using pairs of graphs: one shows the classification error plotted against ε (labelled tolerance); the other is a graph of the number of induced rules against ε. As an example, a canonical run of the RIA for the well-known Iris plant dataset (Fisher, 1988) is presented in Fig. 11.

5.2 Dimensionality reduction and classification performance

The goal of this experiment is to show how RSAR can be of service in reducing the dimensionality of a dataset and, therefore, the time required to run the RIA on the dataset, while allowing learned rules to produce better results than those derived from randomly selected attribute subsets of the same dimensionality. A comparison to the application of Quinlan's C4.5 algorithm (Quinlan, 1993) is included.

Running the RSAR algorithm on the three-class water treatment plant dataset reduced the dataset to eleven conditional attributes. The two-class version

provided a more significant reduction, with RSAR selecting merely two conditional attributes from the total of thirty-eight, namely the conductivity of the water in the primary settler and *any* other attribute. Input flow was arbitrarily chosen as the second conditional attribute because there is no intuitively obvious relation between it, the primary settler conductivity attribute, and the operational status of the plant.

The rule-induction algorithm was then executed using the selected attributes to generate a ruleset. Typical results are illustrated in Figs 12 and 13 for the three-class and two-class datasets respectively. The left plot gives the resulting classification error, while the right plot shows the size of the ruleset generated as a function of the uncertainty margin (labelled 'tolerance'). An additional number of rule induction runs were performed, using randomly selected groups of eleven and two conditional attributes each (for the three-class and two-class datasets respectively). The minimum, average and maximum values for the classification error and ruleset size of these extra runs are shown on the graphs for comparison.

It should be noted that in all these graphs, 'undecidable' answers by the RIA are considered wrong answers, thus giving slightly more conservative results. In this case, the top classification accuracy is around 73% with 'undecidable' counted as a wrong answer, or 77% with 'undecidable' counted as an acceptable admission of ignorance. For the two-class dataset, this was 96.5% or 97.1% respectively. These results were obtained using 10-fold cross-validation of the learned rulesets.

Clearly, the rulesets induced using the set of attributes chosen by the RSAR method are of better quality than the average randomly selected ones, which implies that the latter lose some important information in the course of attribute reduction and rule induction. Detailed investigation revealed that, on top of the information loss incurred by choosing a subset of the original attributes, inexpert fuzzification is also responsible for the error during the rule-induction phase. The fuzzification of certain conditional attributes is less successful than others, leading to the further removal of useful information. The use of a human expert in designing the fuzzification of the application would prove valuable in reducing the classification error of the resultant ruleset. Accuracy is also reduced because of the contents of the dataset. There is very little information present on plant faults, and thus does not allow the system to generalise enough to successfully detect all three (or two) states.

Even given the reduction in ruleset quality, though, the use of the set of attributes discovered by the RSAR algorithm could be several orders of magnitude faster than generating numerous rulesets in the hope of finding the optimum one. Running the RIA on all 38 conditional attributes would be computationally prohibitive; the reduced, eleven-attribute set requires approximately

six seconds per example (this was timed on a common desktop workstation at the time of writing). The RIA is intended for off-line rule induction, where such speeds of computation are acceptable. Additionally, it is important to note that this particular RIA can be trained incrementally. The algorithm offers linear complexity with respect to the number of examples in the dataset — its NP-hard complexity is only with respect to the number of attributes. Nevertheless, the two-attribute version of the dataset is fast enough for rules to be induced on the fly, facilitating immediate inspection by a user.

Although the choice of the RIA used in this application was meant to emphasise the savings of dimensionality reduction, even much more efficient rule induction algorithms would benefit from this. As stated previously, however, the benefits do not limit themselves to the training stage; they extend to the runtime use of the system. By reducing the dimensionality of the data, the dimensionality of the rule set is also decreased. This results in fewer measured variables, which is very important for dynamic systems where observables are often restricted. This in turn leads to fewer connections to instrumentation and faster system responses in emergencies.

C4.5 is a widely accepted and powerful algorithm, providing a good machine learning benchmark. The decision trees it generates can be interpreted very quickly by the monitoring system. However, C4.5 selects twenty-one conditional attributes from the three-class dataset, as compared to the eleven chosen by the RSAR algorithm. This requires ten more gauges to be made available to the monitoring system. The runtime efficiency of the rules induced by such an approach may therefore be reduced, whilst the difficulty in performing effective monitoring and diagnosis would be increased due to the requirement for additional measurements. For the two-class case, C4.5 also requires more conditional attributes to construct the decision tree.

In terms of classification performance, C4.5 obtains classification accuracies of around 81% for the three-class dataset, and 96.8% for the two-class one. For easy comparison, the classification performances of the two approaches are summarised in Table 7. Also included in the table are the numbers of conditional attributes selected by RSAR and C4.5.

It is clear that, for the two-class case, the integrated system reported here performed very well. This shows that real-world problems do contain a lot of redundancy which, once stripped out, enables the technique discussed in this paper to come up with highly accurate rulesets comprising rules with a very limited number of conditional attributes. That said, the current implementation of the proposed framework does not allow the classification accuracy for the three-class case to reach that obtainable by C4.5, though it does use a much smaller set of premise attributes. Several issues are worth noting in relation to this relatively lower classification performance. First, due to the

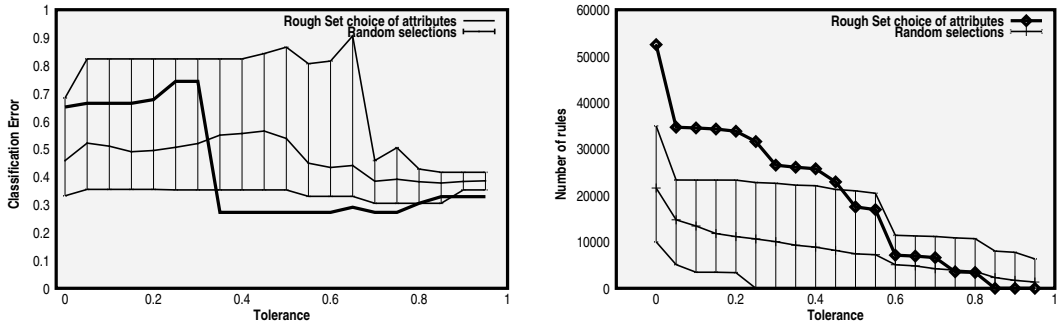

Fig. 12. Effects of using the Rough Set dimensionality-reduction algorithm.

Table 7
Comparison of classification accuracies and the number of conditional attributes required by the algorithms.

Method	Accuracy		Number of Input Attributes	
	2-class	3-class	2-class	3-class
RSAR+RIA				
('undecidable' is wrong)	96.5%	73%	2	11
('undecidable' is acceptable)	97.3%	77%	2	11
C4.5	96.8%	81%	3	21

realism of the dataset, its contents do not provide sufficient training cases to differentiate better between the normal and good performance categories. A substantial majority of misclassifications occurred only between these two classes. For strict monitoring purposes, however, this does not matter anyway. Of course, this also accounts for C4.5's slightly suboptimal performance in the three-class case. Second, the present work requires good fuzzification. In running the experiments reported here, no attempt was made to optimise the fuzzification, either via an automated membership function tuning tool (Gómez Marin-Blazquez et al., 1999) or via a domain expert. The fuzzification was instead performed by a basic statistical count of the dataset. As demonstrated later, better fuzzification leads to better classification. Therefore, with a better partition of the underlying numerical value range, the integrated system should be able to produce a more accurate ruleset. Finally, the ruleset generated by the RIA was not processed by any post-processing tools so as to allow its behaviour and capabilities to be revealed fully. If the induced ruleset is enhanced through post-processing, the performance can be expected to improve. This conjecture is supported by existing work in processing various types of learned ruleset, as reported in the literature (Mees, 1999; Kim et al., 1999; Gómez-Skarmeta and Jiménez, 1997). The experimental evaluation of this aspect remains as an important part of future work.

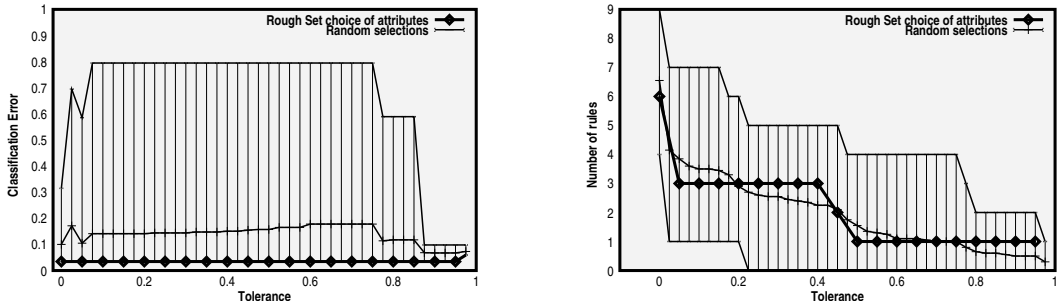

Fig. 13. Impact of dimensionality reduction using a two-class dataset.

5.3 Impact of dimensionality reduction

This experiment set out to investigate whether the integrated system could generate rulesets from reduced sets of conditional attributes such that:

- classification accuracy does not drop significantly;
- ruleset size remains within reasonable limits;
- rule induction time is kept to a minimum.

In order to use a sustained illustrative example throughout the paper, the same water treatment problem is employed here. However, to save computer time involved in carrying out a large number of runs of the system, this set of experiments was conducted on a subset of the original three-class version of the water treatment dataset. In particular, fourteen examples from each of the three classes were randomly selected from the dataset. The reduced size and completeness of the dataset can naturally be expected to account for the higher classification errors shown in Fig. 14. Again, it is worth pointing out that the classification error is calculated by conservatively counting the undecidable classes as incorrectly classified.

Several groups of studies were carried out, using different selections of conditional attributes. The first eleven attributes in each set were the ones selected by RSAR, and the remaining ones were chosen at random. All the results obtained are very positive towards the use of the RSAR approach. As expected, the classification error did not decrease for sets of more than eleven attributes, but the number of induced rules increased significantly, as did the time required for rule induction.

The time required for a full rule-induction run depends on the number of conditional attributes. Hence, the eleven-attribute test took the least time, as shown in Fig. 15. The gain in runtime (and space requirements) justifies the use of RSAR to reduce dataset dimensionality in this classification task.

These results do not particularly flatter this technique. It was considered essential, however, that the system be tested on real-world data, not a simple,

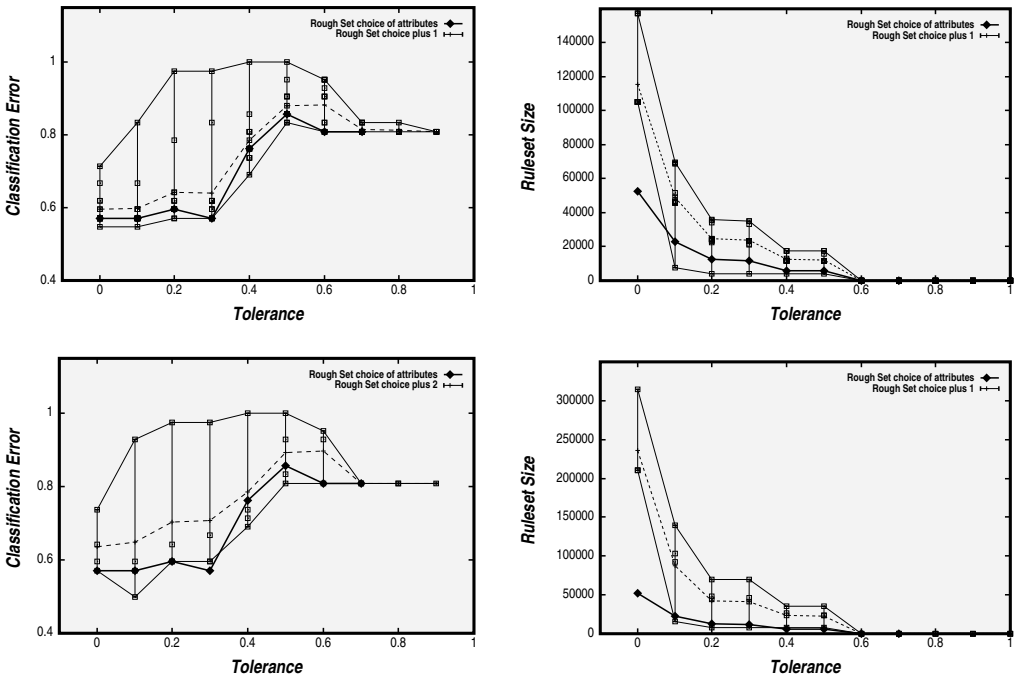

Fig. 14. Ruleset quality after dimensionality reduction, with twelve- (top) and thirteen- (bottom) attribute rulesets, both including the results of using eleven RSAR-selected attributes for comparison.

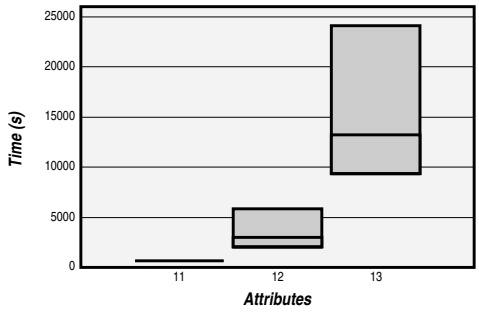

Fig. 15. Time required to induce and evaluate rulesets using eleven-, twelve- and thirteen-attribute datasets. For twelve and thirteen attributes, the entire test group is represented by showing the minimum, average and maximum times.

artificial domain. This application does not exhibit linearly separable clusters of data, which also proves problematic for established and successful machine learning algorithms like C4.5.

5.4 Impact of correct fuzzification

In the preceding discussions, deviations from the classification accuracy achievable by the standard machine learning benchmark for the three-class case were partially blamed on inexpert fuzzification of the domain. This is because the rulesets induced by the integrated rule-induction system are, as with the orig-

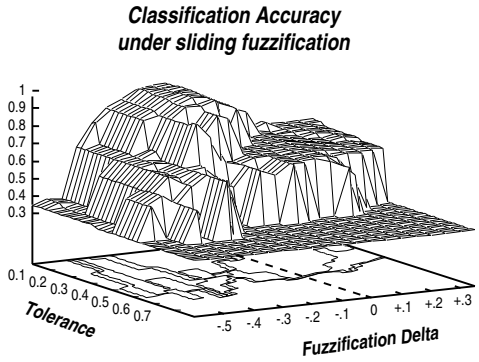

Fig. 16. Implications of shifting membership functions to the left or right along the x axis. Note: classification *accuracy* is shown here rather than classification error. The dashed line marks the original, unshifted fuzzification information from (Lozowski et al., 1996).

inal RIA, based on the evaluation of fuzzy-set membership functions. This makes the choice and configuration of these functions crucial for correct results. The fuzzification used for the monitoring task at hand was performed using basic statistical analysis and intuition rather than human expertise in the specific domain or an established clustering tool.

To demonstrate the importance of correct fuzzification, the system was executed on a simple domain, the Iris dataset. Numerous rule-induction runs were performed and the rulesets were evaluated. For each run, the entire group of fuzzy-set membership functions was shifted to the left or right along the x axis. The results were plotted as a three-dimensional graph, as given in Fig. 16. This is essentially a family of accuracy/tolerance graphs. Classification accuracy is shown, rather than classification error, in an effort to show the effects of correct fuzzification as a hill. The third axis is the shift factor of the fuzzy membership functions. The dashed line marks the original fuzzification information from the work of Lozowski et al. (1996). The highest point in the graph is very close to that fuzzification. Shifting the membership functions to the left or right produces less-accurate rulesets.

There are various approaches to optimising fuzzy rule sets automatically. A few aim at optimising the definition of the membership functions rather than the product ruleset (Gómez-Skarmeta and Jiménez, 1997). Such systems could be applied here to improve the fuzzification information. In addition, the present experiment only investigated the effects of simple modifications of membership functions, with a fixed number of fuzzy sets used per attribute. Work carried out and reported elsewhere, but in close relation to the problem at hand, also shows that changing this number has significant impact upon the performance of the ruleset (Gómez Marin-Blazquez et al., 1999).

6 Conclusion

Most attempts to build successful intelligent monitoring and diagnostic systems tend to stumble on the knowledge-acquisition bottleneck. This paper has presented an approach that integrates rough-set-assisted attribute reduction with a potentially powerful rule-induction algorithm in a modular manner. The RSAR helps reduce the dimensionality of the domain with which the RIA has to cope. The RSAR algorithm has proved to be very useful in stripping out insignificant information, while retaining more important conditional attributes. The most desirable feature of this technique is the fact that, unlike transformation-based approaches, it maintains the underlying semantics of the dataset. In this way the original RIA, which is sensitive to the dimensionality of the dataset, becomes usable on datasets consisting of a moderately large number of attributes. The resulting learned ruleset becomes manageable, and may even outperform rules learned using more conditional attributes.

The framework of the integrated system is also very flexible. It has been designed so that alternative attribute-reduction and/or rule-induction methods may be adapted to substitute the existing modules. Not requiring the presence of human experts is another desirable feature, especially where such expertise is unavailable or even non-existent. Even with less than optimal fuzzification of the domain at hand, the system is able to perform acceptably well for real-world problems. Automated optimisation of the ruleset or the fuzzification may improve the situation further.

The approach is, of course, not perfect. Much improvement work remains to be done. For example, as indicated above, fuzzification plays a very important role in obtaining high-quality rulesets, because of the design of the original RIA. An optimising pre-processor for domain fuzzification would be very helpful. Techniques for fuzzy-set optimisation, typically by the use of a genetic algorithm, to search for the most suitable fuzzy-set definitions, have been proposed (Gómez Marin-Blazquez et al., 1999). Such a technique can be added to the system in an effort to increase the quality of fuzzification when no expert is readily available. Also, an investigation into ruleset post-processors is required; a suitable post-processing module would increase ruleset accuracy while decreasing its cardinality. Several initial attempts at this have recently been reported (Mees, 1999; Kim et al., 1999; Gómez-Skarmeta and Jiménez, 1997), providing a very useful starting point for such further work. Finally, to test and perhaps to reinforce the generality of the proposed approach, work is ongoing to apply the framework to the building of knowledge bases for non-physical (but still real) problem domains. It is also very interesting to investigate the actual effects of utilising an alternative fuzzy rule induction algorithm to the method currently employed, especially one that would allow hierarchical rulesets to be created (Holve, 1998). This may further boost

the runtime efficiency of the learned rulebases which has already been gained by attribute reduction, by performing inference with a guided focus over the hierarchical structure.

References

Chouchoulas, A., and Q. Shen (1998). Rough set-aided rule induction for plant monitoring. *Proceedings of the 1998 International Joint Conference on Information Science*, **2**,316–319.

Cox, E. (1994). *The Fuzzy Systems Handbook: a Practitioner's Guide to Building, Using and Maintaining Fuzzy Systems*. Academic Press, Inc.

Fisher, R. A. (1988). *Iris plants database*. Donated to the UCI Machine Learning Database Repository University of California, Irnive (1995) by Michael Marshall.

Gómez-Skarmeta, A. F., and F. Jiménez (1997). Generating and tuning fuzzy rules using hybrid systems. *Proceedings of the Sixth IEEE International Conference on Fuzzy Systems*, IEEE Press. 247–251.

Gómez Marin-Blazquez, J., Q. Shen, and A. Tuson (1999). Tuning fuzzy membership function with neighbourhood search techniques: a comparative study. *Proceedings of the Third IEEE International Conference on Intelligent Engineering Systems*, IEEE Press.

Haykin, S. (1994). *Neural Networks*. Macmillan College Publishing Company, New York.

Holve, R. (1998). Investigation of automatic rule generation for hierarchical fuzzy systems. *Proceedings of the Seventh IEEE International Conference on Fuzzy Systems*, IEEE Press. 973–978.

Jelonek J., K. Krawiec, and R. Slowinski (1995). Rough set reduction of attributes and their domains for neural networks. *Computational Intelligence*, **11**(2) 339–347.

Kim, M. W., J. G. Lee, and C. Min (1999). Efficient fuzzy rule generation based on fuzzy decision tree for data mining. *Proceedings of the Eighth IEEE International Conference on Fuzzy Systems*, IEEE Press. 1223–1228.

Krishnapuram, R., and J. Keller (1996). The possibilistic c-means algorithm: insights and recommendations. *IEEE Transactions on Fuzzy Systems*, **4**(3) 385–393.

Lozowski, A., T. J. Cholewo, and J. M. Zurada (1996). Crisp rule extraction from perceptron network classifiers. *Proceedings of International Conference on Neural Networks*, volume of Plenary, Panel and Special Sessions, Washington, D.C. 94–99.

Mees, W. (1999). Detection of defects in a fuzzy knowledge base. *Proceedings of the Eighth IEEE International Conference on Fuzzy Systems*, IEEE Press. 204–209.

Pawlak, Z., (1991). *Rough Sets: Theoretical Aspects of Reasoning About Data*.

Kluwer Academic Publishers, Dordrecht.

Poch, M., and J. Bejar and U. Cortes (1993). *Faults in an urban waste water treatment plant (dataset).* Donated to the UCI Machine Learning Database Repository.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning.* The Morgan Kaufmann Series in Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA.

University of California, Irvine (1995). *The Machine Learning Database Repository.* http://www.ics.uci.edu/~mlearn/MLSummary.html

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, **8** 338–353.