'Rough Enough' – Software Demonstration

Anders Torvill Bjorvand
Troll Data Inc.
P.O. Box 335
N-1801 Askim, Norway
torvill@trolldata.no

Keywords: Rough sets, data mining, user interface, software architecture.

ABSTRACT

This article presents the data mining software system 'Rough Enough'. The system supports the process of data mining within the theoretical framework of rough sets. The presentation will emphasize the workflow and construction of the system. Detailed algorithms and representations will not be presented here.

INTRODUCTION

Due to restrictions on the length of this paper, we have to assume that the reader has a working knowledge of rough set theory and data mining. For an introduction, see for instance, [11] and [13].
Many software systems/libraries have been constructed to support the theory of rough sets. They all have a general common quality in addition to specific functionality related to the research performed by the developers. The most well-known systems are: Rough Set Expert System (RSES) [16], the Rough Set Library [6], RoughDas/RoughClass [14], DataLogic [18], Grobian [4], LERS [7] and Rosetta [19].
This presentation is based on version 2.01 of 'Rough Enough' which can be downloaded from the following Internet address: "http://home.sn.no/~torvill".

THE PROCESS OF DATA MINING

The process of data mining is illustrated in the following workflow figure:

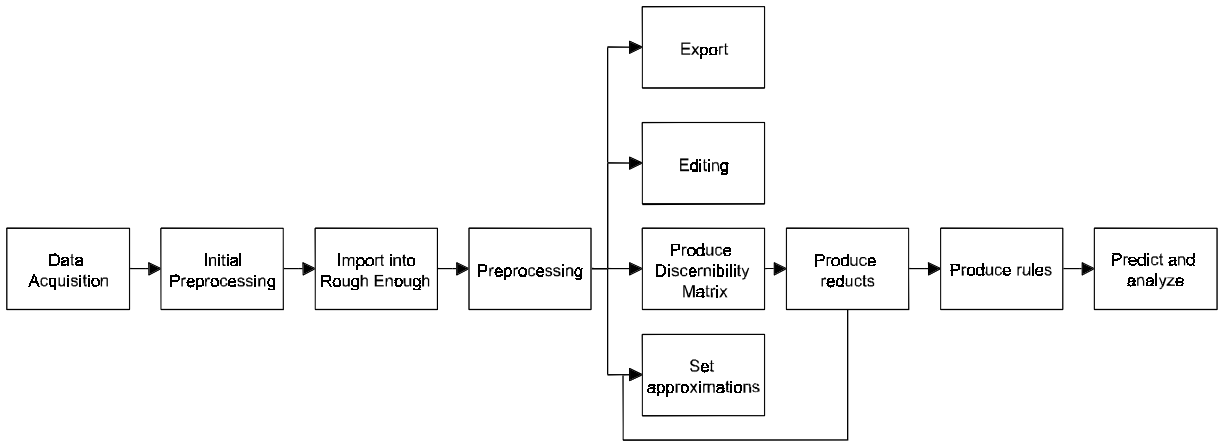
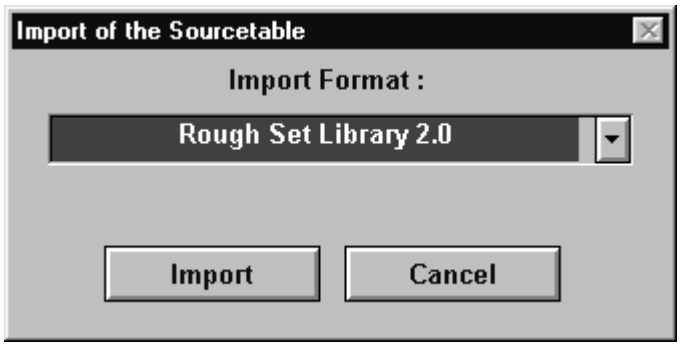


Figure 1 Workflow of data mining

In the presentations that follow, I will relate the functionality to this workflow. The two first points, data acquisition and initial preprocessing, are not supported in 'Rough Enough' and will therefore not be covered.

Import into 'Rough Enough'



The system supports import from most PC database and spreadsheet formats. The Rough Set Expert System (RSES) (see [16]) and the Rough Set Library 2.0 (see [6]) are also supported. SQL servers are easily accessible by just switching the database drivers. This makes the jump to client/server and the use of corporate data an easy one.

Figure 2 Import dialog box

Preprocessing

It is very important to use a relevant preprocessing of the data to achieve the data representation relevant for what you want to do. 'Rough Enough' has several methods of preprocessing. You may choose from the following methods:

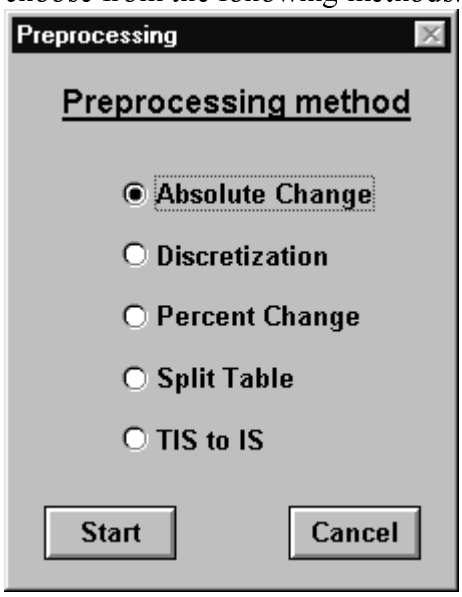


Figure 3 Preprocessing dialog box

- Absolute change: this method calculates the difference between the current object and the previous with respect to object numbering. See [1] for further discussions of this technique.
- Percent change: this method calculates the difference between the current object and the previous object in terms of a percent change.
- Split table: this method splits your table in two based on user input about how many objects you want to keep. The removed objects will be stored elsewhere and used to test rules applied to these unseen objects.
- TIS to IS: translates a temporal information system (time series) into an information system. The algorithm for this procedure and the introduction to temporal information systems can be found in [1] and [2].

In addition to these predefined techniques, the user has direct access to SQL and QBE. This is very convenient if we want to carry out a special task (which is often the case when it comes to data mining).

Editing data

Data can be readily edited in the 'Rough Enough' system. This is done in the main screen as shown in figure 4 where the following table is discretized with the dog breed as the decision attribute:

Object number	Breed	Tail	Hair	Size
1	Doberman Pincher	Short	Short	Big
2	Dalmatian	Long	Short	Big
3	Cocker Spaniel	Short	Long	Small
4	German Shepherd	Long	Medium	Big

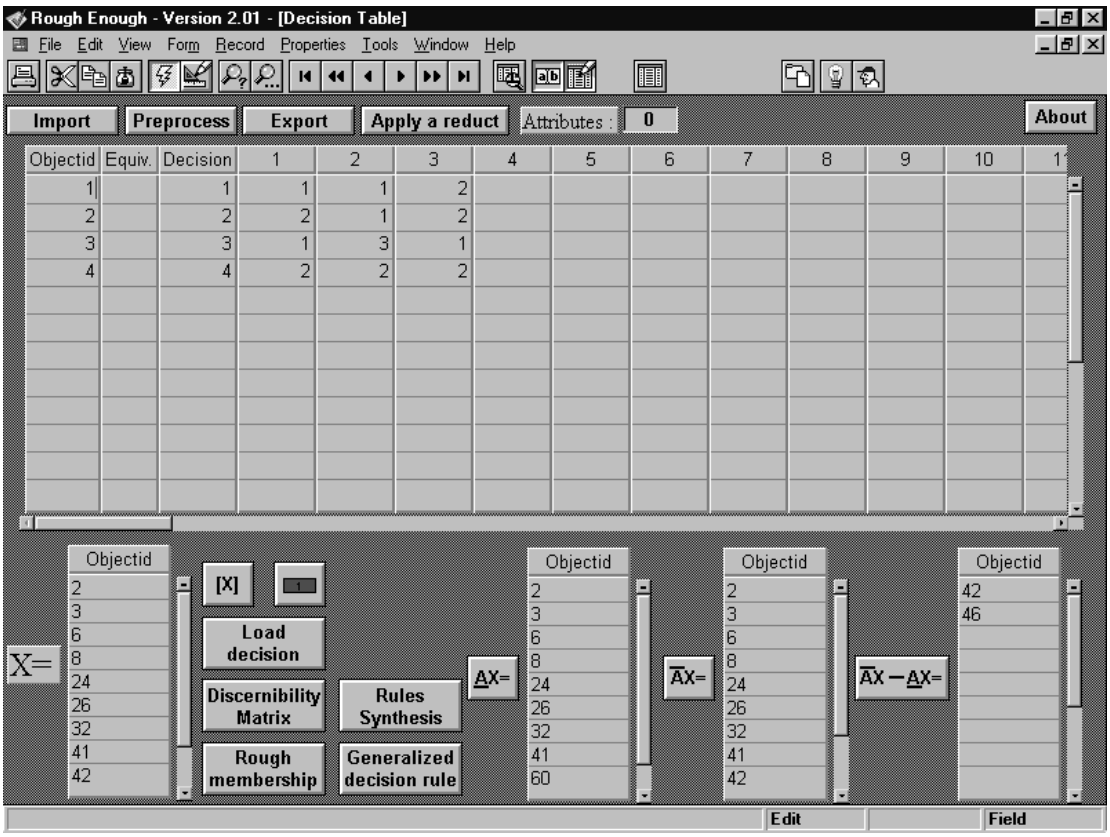


Figure 4 Main screen of 'Rough Enough'

Produce Discernibility Matrix

The discernibility matrix (see [12]) is used for the calculation of reducts and cores.

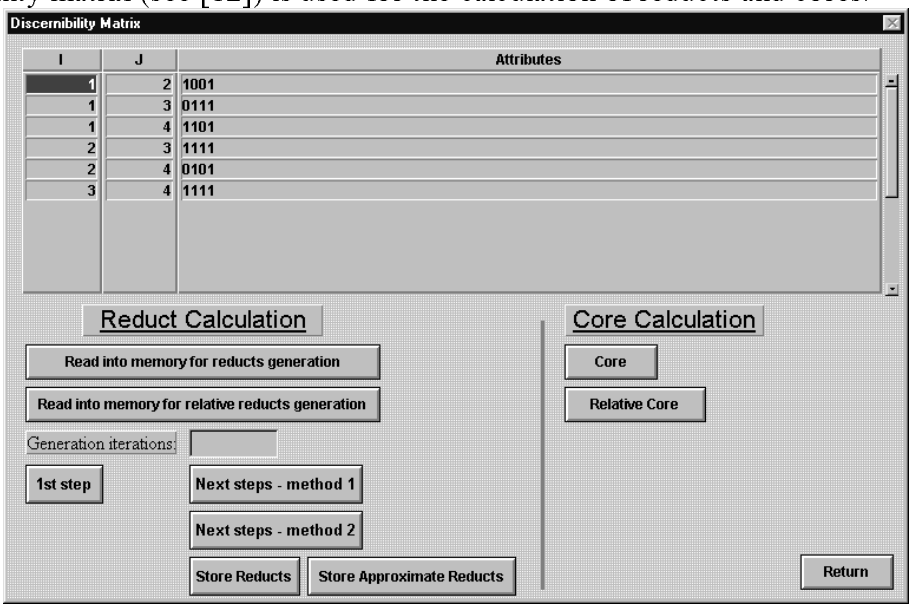


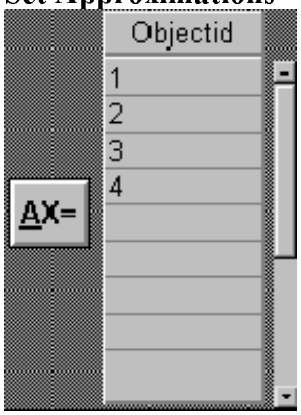
Figure 5 Discernibility matrix

The following discernibility matrix is presented in figure 5:

Objects/Objects	1	2	3	4
1	∅			
2	{Tail}	∅		
3	{Hair, Size}	{Tail, Hair, Size}	∅	
4	{Tail, Hair}	{Hair}	{Tail, Hair, Size}	∅

Since we need a form of the discernibility matrix more suitable for binary manipulation, for every index of the discernibility matrix (every pair of object combination), we use a binary array with as many elements as the number of attributes. A binary one in this array indicates that this particular attribute discerns between these two objects. A zero, of course, indicates the opposite. In this example, the order of the attributes in the binary array is: Tail, Hair, Size:
'Rough Enough' also includes information concerning the decision attribute (if any) in the last position of the array.

Set Approximations



The system has several facilities to work with set approximations: equivalence classes, decision classes, lower approximation, upper approximation, boundary region, rough membership value and generalized decision rule. All these computations again rely on the selections of attributes that you have chosen. These operations are rather trivial, so I will not elaborate on them any further. See [1] for details.

If you press the button shown on figure 6, the lower approximation will be computed and displayed in the small table.

Figure 6 Lower approximation

Produce Reducts

Reducts are found through the use of genetic algorithms. This process is described in more details in [1] and [17]. The reducts that are discovered can easily be used later on to automatically discard certain attributes in the process of creating rules.
As can be seen in figure 7, the genetic algorithm is controlled through every generation. Two different fitness functions (method 1 and 2) can be applied interchangeably.

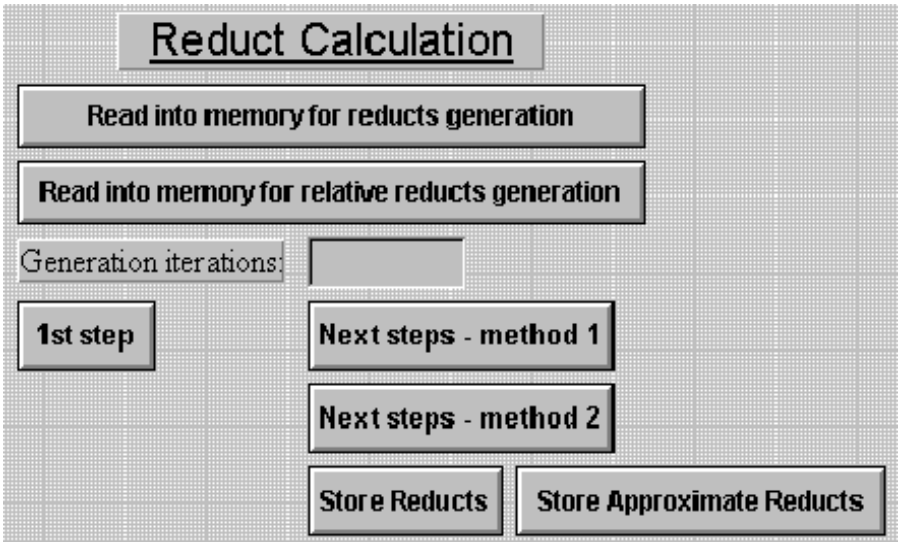


Figure 7 Reduct calculation interface

Produce Rules, Predict and Analyze

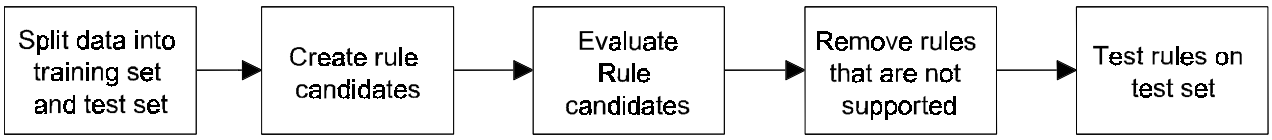


Figure 8 The process of extracting rules

Unfortunately, we can not get detailed about rule creation. The process however, is illustrated in figure 8. Several of the steps can, of course, be iterated. Several voting strategies, described in [1] and [3], are implemented.

CONCLUSION

The 'Rough Enough' system supports most parts of the data mining process. The emphasis of the system is on the use of genetic algorithms for reduct finding and the support for temporal information systems (time series).

FUTURE RESEARCH

A module is under development that produces a decision support system in the highly portable Java™ language (see [5]). This decision support system relies on rules created and modified by the user within the 'Rough Enough' system. The JavaOS™ (see [9]) is a small and compact operating system directed specifically at small, embedded computer systems. The JavaOS™ directly supports the Java Virtual Machine specification (see [8]). This enables us to design embedded computer systems that can support the entire data mining process including data acquisition. The system can also perform actions directly relevant to the decision - eg controlling/modifying an industrial process. The synthesized system will conform to the JavaBean™ (see [15]) distributed component object model.
Modifications and enhancements should be made to the system to directly support the proposed structure of a Real Time Temporal Information System (RTTIS) (see [1] or [2]).

REFERENCES

[1] Bjorvand, Anders Torvill (1996). *Time Series and Rough Sets*. Master's Thesis, the Norwegian Institute of Technology, Department of Computer Systems, Trondheim, Norway.

[2] Bjorvand, Anders Torvill (1997). *Mining Time Series Using Rough Sets - A Case Study*. Proc. 1st European Symposium on Principles of Knowledge Discovery and Data Mining, Trondheim, Norway.

[3] Bjorvand, Anders Torvill (1997). *'Rough Enough' - A System Supporting the Rough Sets Approach*. Sixth Scandinavian Conference on Artificial Intelligence 1997. Helsinki, Finland. To appear.

[4] Gediga, Günther and Düntsch, Ivo (1997). *The Rough Set Engine Grobian*. The 15th IMACS World Congress 1997 on Scientific Computation, Modelling and Applied Mathematics.

[5] Gosling, James and Joy, Bill and Steele, Guy (1996). *The Java Language Specification*. Addison-Wesley.

[6] Gwaryz M. and Sienkiewicz J. (1993). *Rough Set Library, Version 2.0, User's Manual*. Warsaw University of Technology.

[7] Grzymala-Busse, J. W. and Sikora, D. J. (1988). *LERS1 - A System for Learning from Examples Based on Rough Sets*. Report TR-88-5, Department of Computer Science, University of Kansas.

[8] Lindholm, Tim and Yellin, Frank (1997). *The Java™ Virtual Machine Specification*. Addison-Wesley.

[9] Madany, Peter (1997). *JavaOS™: A Standalone Java™ Environment*. Available at the following Internet address: http://www.javasoft.com/products/javaos/javaos.white.html.

[10] Pawlak, Zdzislaw (1982). "Rough Sets". Printed in *International Journal of Computer and Information Sciences*, **11**, pp. 341-356.

[11] Pawlak, Zdzislaw (1991). *Rough Sets - Theoretical Aspects of Reasoning About Data*. Kluwer Academic Publishers, Dordrecht.

[12] Skowron, Andrzej and Rauszer, Cecylia (1992). "The Discernibility Matrices and Functions in Information Systems". Printed in *Intelligent Decision Support - Handbook of Applications and Advances of the Rough Sets Theory*. Edited by Roman Slowinski. Pp. 331-362. Kluwer Academic Publishers, Dordrecht.

[13] Slowinski, Roman (1992) editor. *Intelligent Decision Support - Handbook of Applications and Advances of the Rough Sets Theory*. Kluwer Academic Publishers, Dordrecht.

[14] Slowinski, Roman (1992) and Stefanowski, Jerzy (1992). "'RoughDas' and 'RoughClass' Software Implementations of the Rough Sets Approach". Printed in *Intelligent Decision Support - Handbook of Applications and Advances of the Rough Sets Theory*. Edited by Roman Slowinski. Kluwer Academic Publishers, Dordrecht.

[15] Sun Microsystems (1996). *The JavaBeans™ 1.0 API specification*. Available at the following Internet address: http://splash.javasoft.com/beans/spec.html.

[16] Synak, Piotr (1995). *Rough Set Expert System User's Guide - Version 1.0*. Developed by the group supervised by Professor Andrzej Skowron at the Institute of Mathematics, the University of Warsaw.

[17] Wróblewski, Jakub (1995). *Finding Minimal Reducts Using Genetic Algorithms (extended version)*. Warsaw University of Technology - Institute of Computer Science - Reports - 16/95.

[18] Ziarko, W. and Golan, R. and Edwards, D. (1993). *An Application of Datalogic/R Knowledge Discovery Tool to Identify Strong Predictive Rules in Stock Market Data*. AAAI-93 Workshop on Knowledge Discovery in Databases.

[19] Øhrn, Alexander and Komorowski, Jan (1997). *ROSETTA - A Rough Set Toolkit for Analysis of Data*. Proc. Third International Joint Conference on Information Sciences, Durham, NC, USA, Vol. 3, pp. 403-407.