



An Introduction to Rough Set Theory and Its Applications

A tutorial

Zbigniew Suraj

Chair of Computer Science Foundations,
University of Information Technology and Management,
H. Sucharskiego Str. 2, 35-225 Rzeszów, Poland

zsuraj@wsiz.rzeszow.pl

ICENCO'2004, December 27-30, 2004, Cairo, Egypt

Contents

Introduction	2
1. Basic Concepts of Rough Sets	9
2. Rough Sets in Data Analysis	14
3. Rough Sets and Concurrency	30

Introduction

Rough set theory is a new mathematical approach to imperfect knowledge.

The problem of imperfect knowledge has been tackled for a long time by philosophers, logicians and mathematicians. Recently it became also a crucial issue for computer scientists, particularly in the area of Artificial Intelligence. There are many approaches to the problem of how to understand and manipulate imperfect knowledge. The most successful approach is based on the fuzzy set notion proposed by L. Zadeh in the article *Fuzzy sets*, Information and Control, 8, 338-353, 1965.

Rough set theory proposed by Z. Pawlak in [1] presents still another attempt to this problem. The theory has attracted attention of many researchers and practitioners all over the world, who contributed essentially to its development and applications.

Rough set theory has a close connections with many other theories. Some of them, we will discuss here. Despite of its connections with other theories, the rough set theory may be considered as own independent discipline.

Rough sets have been proposed for a very wide variety of applications. In particular, the rough set approach seems to be important for Artificial Intelligence and cognitive sciences, especially in machine learning, knowledge discovery, data mining, expert systems, approximate reasoning and pattern recognition.

The main advantages of the rough set approach are as follows:

- It does not need any preliminary or additional information about data – like probability in statistics, grade of membership in the fuzzy set theory.
- It provides efficient methods, algorithms and tools for finding hidden patterns in data.
- It allows to reduce original data, i.e. to find minimal sets of data with the same knowledge as in the original data.
- It allows to evaluate the significance of data.
- It allows to generate in automatic way the sets of decision rules from data.
- It is easy to understand.
- It offers straightforward interpretation of obtained results.
- It is suited for concurrent (parallel/distributed) processing.
- It is easy internet access to the rich literature about the rough set theory, its extensions as well as interesting applications, e.g. <http://www.rsd.s.wsz.rzeszow.pl>

The first, pioneering paper on rough sets, written by Zdzisław Pawlak¹, was published by International Journal of Computer and Information Sciences in 1982.

From 1982 to 1990, the Warsaw Research Group at the Polish Academy of Sciences and Warsaw University of Technology was most active in conducting rough set related works, and produced many reports and theses on rough sets. In September 1992, there was the first international workshop on rough sets in Poznań, Poland, together with conference proceedings has been organized [18]. The selected papers from this conference were published in Intelligent Decision Support--Handbook of Applications and Advances of the Rough Sets Theory by Kluwer Academic Publishers in 1992. Most of the rough set related papers written in English before 1990 are listed in the annotated bibliography of the first book [2] on rough sets. More recent papers up until 1998 and those works done in Poland, North America, Asia and other European countries are annotated in the appendix 1 of another book [12]. In [12] there is also short information about the software systems relating to rough sets. The first tutorial article [45] provides a complementary, easy-to-read introduction to rough sets.

Since the early-1990's, many people has been very active in organizing workshops and publishing conference proceedings on rough sets [18]-[30]. The distribution of the proceedings of these workshops is limited to mostly the workshop participants. The first International Conference on Rough Sets and Current Trends in Computing was held in 1998 at Warsaw, Poland [31]. Since then, this series of conferences has been held every two years at different locations in Europe and North America: 1998, Warsaw, Poland; 2000, Banff, Canada; 2002, Malvern, USA; 2004, Uppsala, Sweden. The selected papers from these conferences and other articles have been published by Springer-Verlag as the series of Lecture Notes in Artificial Intelligence [31]-[34]. In April 2004, the first volume of a new LNCS journal subline – *Transactions on Rough Sets* published by Springer-Verlag was appeared [52]. The journal includes high-quality research articles, dissertations, monographs and extended and revised versions of selected papers from conferences. A good source of information about the most recent rough set literature mostly with short abstracts provides the Rough Set Database System [46-47]. This database contains over 1700 entries published from 1982 to 2004. Since 1996 the International Rough Set Society publishes own bulletin including important and current information for the rough set community [48-51].

In October 2003, another series of international conferences was initiated. This series is a continuation of international workshops devoted to the subject of rough sets, held in Canada, China, Japan, Poland, Sweden and the USA. It achieved a status of bi-annual international conference, starting from the year of 2003, in Chongqing, China. The conference encompasses rough sets, fuzzy sets, granular computing as well as knowledge discovery and data mining. The second conference of this series is to be held in Regina, Canada, in September 2005; the third is planned in Japan in 2007. The proceedings of the first conference [35] are available from the Springer-Verlag in Berlin, Germany. Most rough set research groups have their own software packages and tools to assist the analysis, and/or simulation of various applications. The short information on software systems based on rough sets are annotated in the Appendix 2 of the book [12].

The rest of the tutorial paper is organized as follows. Section 1 provides a general formulation of basic concepts of rough sets. In Section 2, the basic concepts of rough set

¹ Zdzisław Pawlak (1926-) – Polish mathematician and computer scientist, a professor of the Polish Academy of Sciences and Warsaw University of Technology.

theory in terms of data are presented. Section 3 discusses some relationships between rough sets and concurrency.

This paper is a draft version of a tutorial delivered at the 1st International Computer Engineering Conference on New Technologies for the Information Society organized by the Cairo University in December 2004.

References

An article

1. Z. Pawlak: Rough sets, *International Journal of Computer and Information Sciences*, 11, 341-356, 1982.

Books

2. Pawlak, Z.: *Rough Sets Theoretical Aspects of Reasoning about Data*. Kluwer Academic Publishers, Dordrecht 1991.
3. Cios, K.J., Pedrycz, W., Swiniarski, R.W.: *Data Mining. Methods for Knowledge Discovery*. Kluwer Academic Publishers, Dordrecht 1998.
4. Demri, S.P., Orlowska, E.,S.: *Incomplete Information: Structure, Inference, Complexity*. Springer-Verlag, Berlin 2002.
5. Polkowski, L.: *Rough Sets. Mathematical Foundations*. Springer-Verlag, Berlin 2002.

Edited books

6. R. Slowinski (Ed.): *Intelligent Decision Support--Hanbook of Applications and Advances of the Rough Sets Theory*. Kluwer Academic Publishers, Dordrecht 1992.
7. W. Ziarko (Ed.): *Rough Sets, Fuzzy Sets and Knowledge Discovery (RSKD'93). Workshops in Computing*, Springer-Verlag & British Computer Society, London, Berlin 1994.
8. T.Y. Lin, A.M. Wildberger (Eds.): *Soft Computing: Rough Sets, Fuzzy Logic, Neural Networks, Uncertainty Management, Knowledge Discovery*. Simulation Councils, Inc., San Diego, CA, 1995.
9. T.Y. Lin, N. Cercone (Eds.): *Rough Sets and Data Mining. Analysis of Imprecise Data*. Kluwer Academic Publishers, Dordrecht 1997.
10. E. Orlowska (Ed.): *Incomplete information: Rough set analysis*. Physica-Verlag, Heidelberg, 1997.
11. L. Polkowski, A. Skowron (Eds.): *Rough Sets in Knowledge Discovery 1. Methodology and Applications*. Physica-Verlag, Heidelberg 1998.
12. L. Polkowski, A. Skowron (Eds.): *Rough Sets in Knowledge Discovery 2. Applications, Case Studies and Software Systems*. Physica-Verlag, Heidelberg 1998.
13. W. Pedrycz, J.F. Peters (Eds.): *Computational Intelligence in Software Engineering*. World Scientific Publishing, Singapore 1998.
14. S.K. Pal, A. Skowron (Eds.): *Rough Fuzzy Hybridization: A New Trend in Decision-Making*. Springer-Verlag, Singapore 1999.
15. L. Polkowski, S. Tsumoto, T.Y. Lin (Eds.): *Rough Set Methods and Applications. New Developments in Knowledge Discovery in Information Systems*. Physica-Verlag, Heidelberg, 2000.
16. M. Inuiguchi, S. Hirano, S. Tsumoto (Eds.): *Rough Set Theory and Granular Computing, Studies in Fuzziness and Soft Computing, Vol. 125*, Springer-Verlag, Berlin 2003.
17. S.K. Pal, L. Polkowski, A. Skowron (Eds.): *Rough-Neural Computing. Techniques for Computing with Words*. Springer-Verlag, Berlin 2004.

Workshop proceedings

18. R. Słowiński, J. Stefanowski (Eds.): Proceedings on the First International Workshop on Rough Sets: State of the Art and Perspectives. Kiekrz -- Poznań, Poland, September 2-4, 1992.
19. W. Ziarko (Ed.): Proceedings of the Second International Workshop on Rough Sets and Knowledge Discovery (RSKD'93). Banff, Alberta, Canada, October 12-15, 1993.
20. T.Y. Lin (Ed.): Proceedings of the Third International Workshop on Rough Sets and Soft Computing (RSSC'94). San Jose State University, San Jose, California, USA, November 10-12, 1994.
21. T.Y. Lin (Ed.): Proceedings of the Workshop on Rough Sets and Data Mining at 23rd Annual Computer Science Conference, Nashville, Tennessee, March 2, 1995.
22. P.P. Wang (Ed.): Proceedings of the International Workshop on Rough Sets and Soft Computing at Second Annual Joint Conference on Information Sciences (JCIS'95), Wrightsville Beach, North Carolina, 28 September -- 1 October, 1995.
23. S. Tsumoto, S. Kobayashi, T. Yokomori, H. Tanaka, and A. Nakamura (Eds.): Proceedings of the Fourth International Workshop on Rough Sets, Fuzzy Sets and Machine Discovery (RSFD'96). The University of Tokyo, November 6-8, 1996.
24. P.P. Wang (Ed.): Proceedings of the Fifth International Workshop on Rough Sets and Soft Computing (RSSC'97) at Third Annual Joint Conference on Information Sciences (JCIS'97). Duke University, Durham, NC, USA, Rough Set and Computer Science 3, March 1-5, 1997.
25. S. Hirano, M. Inuiguchi, S. Tsumoto (Eds.): Proceedings of International Workshop on Rough Set Theory and Granular Computing (RSTGC'2001), Matsue, Shimane, Japan, May 20-22, 2001. Bulletin of International Rough Set Society 5/1-2 (2001).
26. Z. Suraj (Ed.): Proceedings of the Sixth International Conference on Soft Computing and Distributed Processing (SCDP 2002), June 24-25, 2002, Rzeszow, Poland, University of Information Technology and Management Publisher, Rzeszow 2002.
27. M. Inuiguchi, S. Miyamoto (Eds.): Proceedings of the First Workshop on Rough Sets and Kansei Engineering in Japan, December 14-15, 2002, Tokyo, Bulletin of International Rough Set Society 7/1-2 (2003).
28. A. Skowron, M. Szczuka (Eds.): Proceedings of an International Workshop on Rough Sets in Knowledge Discovery and Soft Computing, RSDK, Warsaw, Poland, April 5-13, 2003, Warsaw University, 2003.
29. L. Czaja (Ed.): Proceedings of the Workshop on Concurrency, Specification and Programming, CS&P'2003, Vol. 1-2, Czarna, Poland, September 25-27, 2003, Warsaw University, 2003.
30. G. Lindemann, H.-D. Burkhard, L. Czaja, A. Skowron, H. Schlingloff, Z. Suraj (Eds.): Proceedings of the Workshop on Concurrency, Specification and Programming, CS&P'2004, Vol. 1-3, Caputh, German, September 24-26, 2004, Humboldt University, Berlin 2004.

Conference proceedings

RSCTC

31. L. Polkowski, A. Skowron (Eds.): Proceedings of the First International Conference on Rough Sets and Current Trends in Computing (RSCTC'98), Warsaw, Poland, 1998, Lecture Notes in Artificial Intelligence 1424, Springer-Verlag, Berlin 1998.

32. W. Ziarko, Y.Y. Yao (Eds.): Rough Sets and Current Trends in Computing. Second International Conference, RSCTC 2000, Banff, Canada, October 16-19, 2000, Lecture Notes in Artificial Intelligence 2005, Springer-Verlag, Berlin 2001.
33. J.J. Alpigini, J.F. Peters, A. Skowron, N. Zhong (Eds.): Rough Sets and Current Trends in Computing. Third International Conference, RSCTC 2002, Malvern, PA, USA, October 14-16, 2002, Lecture Notes in Artificial Intelligence 2475, Springer-Verlag, Berlin 2002.
34. S. Tsumoto, R. Slowinski, J. Komorowski, J.W. Grzymala-Busse (Eds.): Rough Sets and Current Trends in Computing. Fourth International Conference, RSCTC 2004, Uppsala, Sweden, June 1-5, 2004, Lecture Notes in Artificial Intelligence 3066, Springer-Verlag, Berlin 2004.

RSFDGrC

35. G. Wang, Q. Liu, Y.Y. Yao, A. Skowron (Eds.). Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing. 9th International Conference, RSFDGrC 2003, Chongqing, China, May 26-29, 2003, Lecture Notes in Artificial Intelligence 2639, Springer-Verlag, Berlin 2003.
36. W. Ziarko, Y.Y. Yao, Xiaohua Hu, D. Ślęzak (Eds.). Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing. 9th International Conference, RSFDGrC 2005, Regina, Canada, September 1-3, 2005.

Special issues

37. R. Slowinski, J. Stefanowski (Eds.), Foundations of Computing and Decision Sciences 18/3-4 (1993) 155-396.
38. W. Ziarko (Ed.): Computational Intelligence: An International Journal 11/2 (1995).
39. T.Y. Lin (Ed.): Journal of the Intelligent Automation and Soft Computing 2/2 (1996).
40. T.Y. Lin (Ed.): International Journal of Approximate Reasoning 15/4 (1996).
41. W. Ziarko (Ed.): Fundamenta Informaticae 27/2-3 (1996).
42. Skowron, S.K. Pal (Eds.): Pattern Recognition Letters 24/6 (2003).
43. Burkhard, H.D., Lindemann, G., Czaja, L., Suraj, Z. (Eds.): Fundamenta Informaticae 60/1-4 (2004).
44. Suraj, Z. (Ed.): Fundamenta Informaticae 61/2 (2004).
45. A. ella Hassanien, H. Own (Eds.): International Journal of Artificial Intelligence and Machine Learning 2005.

Tutorials

46. J. Komorowski, Z. Pawlak, L. Polkowski, A. Skowron: Rough Sets: A Tutorial, in S.K. Pal, A. Skowron (Eds.): Rough Fuzzy Hybridization. A New Trend in Decision-Making, Springer-Verlag, Singapore 1999, 3-98.
47. A. Skowron, Z. Pawlak, J. Komorowski, L. Polkowski: A rough set perspective on data and knowledge, in W. Kloesgen, J. Żytkow (Eds.): Handbook of KDD. Oxford University Press, Oxford (2002), 134-149.
48. Z. Pawlak: Some Issues on Rough Sets, in J.F. Peters, A. Skowron (Eds.): Transactions on Rough Sets I, LNCS 3100 journal subline, Springer-Verlag, Germany, 2004, 1-58.

Bibliographic databases

The Rough Set Database System <http://www.rsds.wsiz.rzeszow.pl>

49. Z. Suraj, P. Grochowalski: The Rough Sets Database System: An Overview in: S. Tsumoto, Y.Y. Yao (Eds.): Bulletin of International Rough Set Society 1/2 (2003). First Workshop on Rough Sets and Kansei Engineering, Tokyo, Japan, December 14-15, 2002.
50. Z. Suraj, P. Grochowalski: The Rough Sets Database System: An Overview in: S. Tsumoto, R. Slowinski, J. Komorowski, J.W. Grzymala-Busse (Eds.): Rough Sets and Current Trends in Computing. Fourth International Conference, RSCTC 2004, Uppsala, Sweden, June 1-5, 2004, Lecture Notes in Artificial Intelligence 3066, Springer-Verlag, Berlin 2004.

Bulletins

The International Rough Set Society, Japan: <http://www.roughsets.org>

51. S. Tsumoto (Ed.): Bulletin of International Rough Set Society 1/1 (1996).
52. S. Tsumoto (Ed.): Bulletin of International Rough Set Society 1/2 (1997).
53. S. Tsumoto, Y.Y. Yao, and M. Hadjimichael (Eds.): Bulletin of International Rough Set Society 2/1 (1998).
54. S. Tsumoto, and Y.Y. Yao (Eds.): Bulletin of International Rough Set Society 1/2 (2003).

Journals

55. J.F. Peters, A. Skowron (Eds.): Transactions on Rough Sets I, LNCS 3100 journal subline, Springer-Verlag, Germany, 2004.

Selected software systems

56. Rough Set Exploration System (RSES)
57. Rosetta

<http://logic.mimuw.edu.pl>



Zdzisław Pawlak received his M.Sc. in 1951 in Electronics from the Warsaw University of Technology, Ph.D. in 1958 and D.Sc. in 1963 in the Theory of Computation from the Polish Academy of Sciences. He is a Professor of the Institute of Theoretical and Applied Informatics, the Polish Academy of Sciences and the University of Information Technology and Management and a Member of the Polish Academy of Sciences. His current research interests include intelligent systems and cognitive sciences, in particular, decision support systems, knowledge representation, reasoning about knowledge, machine learning, inductive reasoning, vagueness, uncertainty and decision support.

He is an author of a new mathematical tool, called rough set theory, intended to deal with vagueness and uncertainty. Over two thousand and three hundred papers have been published by now on rough sets and their applications world wide. Several international workshops and conferences on rough sets have been held in recent years.

He is a recipient of many awards among others the State Award in Computer Science in 1978, the Hugo Steinhaus award for achievements in applied mathematics in 1989; Doctor Honoris Causa of Poznań University of Technology in 2002; a member or past member of editorial boards of several dozens international journals. Program committee member of many international conferences on computer science. He held over forty visiting university appointments in Europe, USA and Canada, about fifty invited international conference talks, and over one hundred seminar talks given in about fifty universities in Europe, USA, Canada, China, India, Japan, Korea, Taiwan, Australia and Israel. He published around two hundred articles in international journals and several books on various aspects on computer science and application of mathematics and supervised thirty Ph.D. theses in computer science and applied mathematics.



Zbigniew Suraj received his M.Sc. in 1971 from the Pedagogical University in Rzeszów, Ph.D. in 1985 from the Warsaw University in Mathematics, and D.Sc. (habilitation) in 2000 in Computer Science from the Institute of Computer Science, the Polish Academy of Sciences. He is a Professor at the University of Information Technology and Management in Rzeszów, Poland, Head of the Chair of Computer Science Foundations at the University. He has held visiting appointments at many universities in Europe and Canada. He received a number of awards for his scientific achievements in computer science including an individual award of the Polish Ministry of Science, High Education and Technology (1985) and the team-award of the Polish Ministry of Education (1993).

Currently, he is a member of the Editorial Board of the Transactions on Rough Sets published by Springer-Verlag. He was the editor and co-editor of special issues of Fundamenta Informaticae published by IOS Press. He is an author and co-author of 4 books in computer science. He has published over 140 research papers in prestigious journals and conference proceedings. He has developed both software packages in rough sets and Petri nets areas as well as the original rough set bibliography database system.

His recent research interests focus on rough set theory and its applications, knowledge discovery and data mining, approximate reasoning, discovering concurrent data models and decision algorithms from data, modeling with Petri nets, selected aspects of mobile robotics.

He has organized many domestic and international conferences. He is a member of program committee of many international conferences on rough sets and related domains including RSCTC, RSFDGrC, AICCSA and others. His research is supported by the State Committee for Scientific Research in Poland.

He is a member of the International Rough Set Society, the Polish Information Processing Society, the Vice-President of the Polish Mathematical Society – Branch in Rzeszów, the Vice-President of the Scientific Society in Rzeszów.

1. Basic Concepts of Rough Sets

1.1 Approximation Spaces and Set Approximations

Rough set theory [1] proposes a new mathematical approach to imperfect knowledge, i.e. to vagueness (or imprecision). In this approach, vagueness is expressed by a boundary region of a set.

Rough set concept can be defined by means of topological operations, *interior* and *closure*, called *approximations*.

Now, we describe this problem more precisely. Let a finite set of objects U and a binary relation $R \subseteq U \times U$ be given. The sets U, R are called the *universe* and an *indiscernibility relation*, respectively. The discernibility relation represents our lack of knowledge about elements of U . For simplicity, we assume that R is an equivalence relation. A pair (U, R) is called an *approximation space*, where U is the universe and R is an equivalence relation on U .

Let X be a subset of U , i.e. $X \subseteq U$. Our goal is to characterize the set X with respect to R .

In order to do it, we need additional notation and basic concepts of rough set theory which are presented below.

By $R(x)$ we denote the equivalence class of R determined by element x . The indiscernibility relation R describes - in a sense - our lack of knowledge about the universe U . Equivalence classes of the relation R , called *granules*, represent an elementary portion of knowledge we are able to perceive due to R . Using only the indiscernibility relation, in general, we are not able to observe individual objects from U but only the accessible granules of knowledge described by this relation.

- The set of all objects which can be with *certainty* classified as members of X with respect to R is called the *R-lower approximation* of a set X with respect to R , and denoted by $R_*(X)$, i.e.

$$R_*(X) = \{x : R(x) \subseteq X\}.$$

- The set of all objects which can be only classified as *possible* members of X with respect to R is called the *R-upper approximation* of a set X with respect to R , and denoted by $R^*(X)$, i.e.

$$R^*(X) = \{x : R(x) \cap X \neq \emptyset\}.$$

- The set of all objects which can be decisively classified neither as members of X nor as members of $-X$ with respect to R is called the *boundary region* of a set X with respect to R , and denoted by $RN_R(X)$, i.e.

$$RN_R(X) = R^*(X) - R_*(X).$$

Now we are ready to formulate the definition of the rough set notion.

- A set X is called *crisp (exact)* with respect to R if and only if the boundary region of X is empty.

- A set X is called *rough (inexact)* with respect to R if and only if the boundary region of X is nonempty.

The definitions of set approximations presented above can be expressed in terms of granules of knowledge in the following way. The lower approximation of a set is union of all granules which are entirely included in the set; the upper approximation – is union of all granules which have non-empty intersection with the set; the boundary region of a set is the difference between the upper and the lower approximation of the set.

Figure 1 presents the graphical illustration of the set approximations defined above.

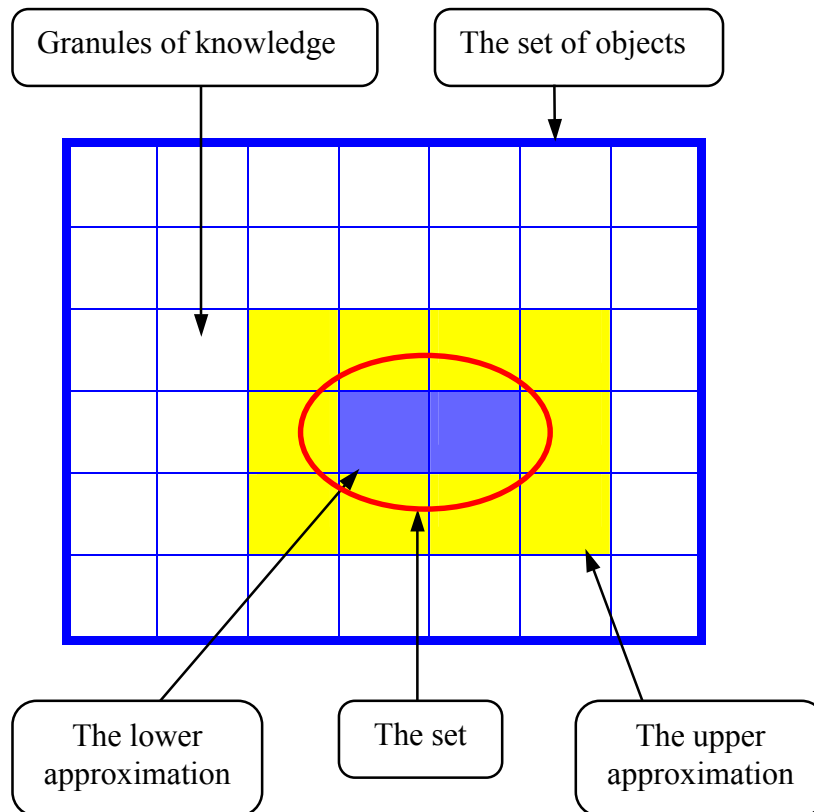


Fig. 1

It is interesting to compare definitions of classical sets, fuzzy sets and rough sets. Classical set is a primitive notion and is defined intuitively or axiomatically. Fuzzy sets are defined by employing the fuzzy membership function, which involves advanced mathematical structures, numbers and functions. Rough sets are defined by approximations. Thus this definition also requires advanced mathematical concepts.

One can easily prove the following properties of approximations:

1. $R_*(X) \subseteq X \subseteq R^*(X)$
2. $R_*(\emptyset) = R^*(\emptyset) = \emptyset$; $R_*(U) = R^*(U) = U$
3. $R^*(X \cup Y) = R^*(X) \cup R^*(Y)$
4. $R_*(X \cap Y) = R_*(X) \cap R_*(Y)$
5. $R_*(X \cup Y) \supseteq R_*(X) \cup R_*(Y)$

6. $R^*(X \cap Y) \subseteq R^*(X) \cap R^*(Y)$
7. $X \subseteq Y \rightarrow R_*(X) \subseteq R_*(Y) \& R^*(X) \subseteq R^*(Y)$
8. $R_*(-X) = -R^*(X)$
9. $R^*(-X) = -R_*(X)$
10. $R_*R_*(X) = R^*R_*(X) = R_*(X)$
11. $R^*R^*(X) = R_*R^*(X) = R^*(X)$

It is easily seen that the lower and the upper approximations of a set are, respectively the interior and closure of this set in the topology generated by the indiscernibility relation.

One can define the following four basic classes of rough sets, i.e., four categories of vagueness:

1. A set X is *roughly R -definable*, iff $R_*(X) \neq \emptyset$ and $R^*(X) \neq U$.
2. A set X is *internally R -undefinable*, iff $R_*(X) = \emptyset$ and $R^*(X) \neq U$.
3. A set X is *externally R -undefinable*, iff $R_*(X) \neq \emptyset$ and $R^*(X) = U$.
4. A set X is *totally R -undefinable*, iff $R_*(X) = \emptyset$ and $R^*(X) = U$.

The intuitive meaning of this classification is the following.

A set X is *roughly R -definable* means that with respect to R we are able to decide for some elements of U that they belong to X and for some elements of U that they belong to $-X$.

A set X is *internally R -undefinable* means with respect to R we are able to decide for some elements of U that they belong to $-X$, but we are unable to decide for any element of U whether it belongs to X .

A set X is *externally R -undefinable* means that with respect to R we are able to decide for some elements of U that they belong to X , but we are unable to decide for any element of U whether it belongs to $-X$.

A set X is *totally R -undefinable* means that with respect to R we are unable to decide for any element of U whether it belongs to X or $-X$.

A rough set X can be also characterized numerically by the following coefficient

$$\alpha_R(X) = \frac{|R_*(X)|}{|R^*(X)|}$$

called the *accuracy of approximation*, where $|X|$ denotes the cardinality of $X \neq \emptyset$.

Obviously $0 \leq \alpha_R(X) \leq 1$. If $\alpha_R(X) = 1$ then X is *crisp* with respect to R (X is *precise* with respect to R), and otherwise, if $\alpha_R(X) < 1$, X is *rough* with respect to R (X is *vague* with respect to R).

1.2 Rough Membership Function

Rough sets can be also defined by using, instead of approximations, a rough membership function proposed in [2].

In classical set theory, either an element belongs to a set or it does not. The corresponding membership function is the characteristic function for the set, i.e. the function takes values 1 and 0, respectively. In the case of rough sets, the notion of membership is different. The *rough membership function* quantifies the degree of relative overlap between the set X and the equivalence class $R(x)$ to which x belongs. It is defined as follows:

$$\mu_x^R : U \rightarrow \langle 0, 1 \rangle$$

where

$$\mu_x^R(x) = \frac{|X \cap R(x)|}{|R(x)|}$$

and $|X|$ denotes the cardinality of X .

The rough membership function expresses conditional probability that x belongs to X given R and can be interpreted as a degree that x belongs to X in view of information about x expressed by R .

The meaning of rough membership function can be depicted as shown in Fig.2.

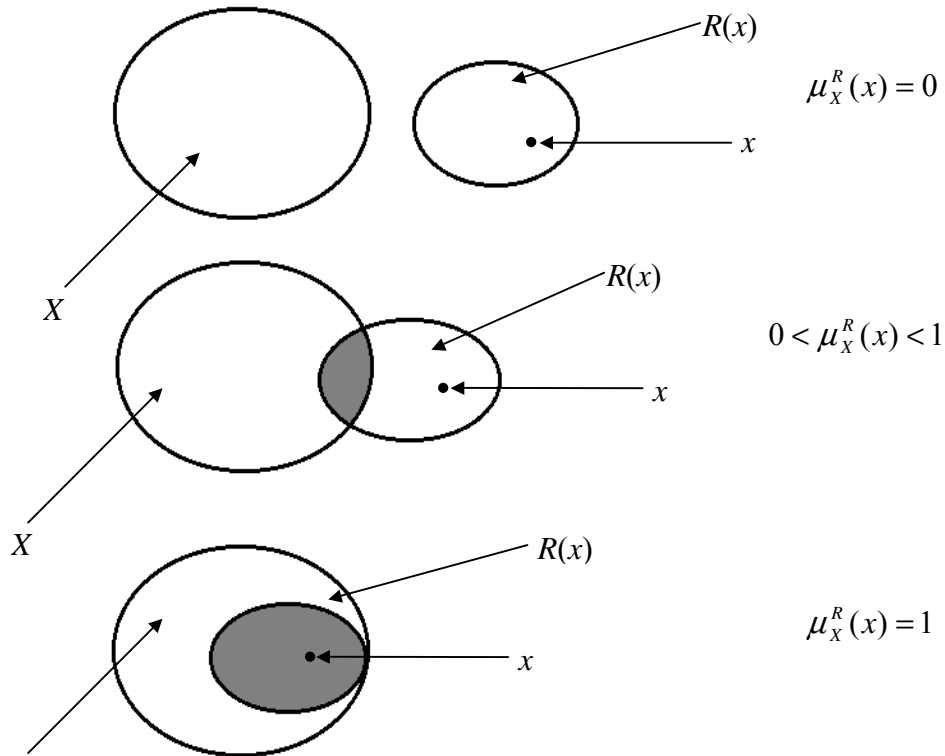


Fig. 2

The rough membership function can be used to define approximations and the boundary region of a set, as shown below:

$$R_*(X) = \{ x \in U : \mu_x^R(x) = 1 \},$$

$$R^*(X) = \{ x \in U : \mu_x^R(x) > 0 \},$$

$$RN_R(X) = \{ x \in U : 0 < \mu_x^R(x) < 1 \}.$$

It can be shown that the rough membership function has the following properties [2]:

1. $\mu_X^R(x) = 1$ iff $x \in R_*(X)$
2. $\mu_X^R(x) = 0$ iff $x \in U - R^*(X)$
3. $0 < \mu_X^R(x) < 1$ iff $x \in RN_R(X)$
4. $\mu_{U-X}^R(x) = 1 - \mu_X^R(x)$ for any $x \in U$
5. $\mu_{X \cup Y}^R(x) \geq \max(\mu_X^R(x), \mu_Y^R(x))$ for any $x \in U$
6. $\mu_{X \cap Y}^R(x) \leq \min(\mu_X^R(x), \mu_Y^R(x))$ for any $x \in U$

From the above properties it follows that the rough membership differs essentially from the fuzzy membership, for properties 5 and 6 show that the membership for union and intersection of sets, in general, cannot be computed – as in the case of fuzzy sets – from their constituents membership. Thus formally the rough membership is a generalization of the fuzzy membership. Besides, the rough membership function, in contrast to fuzzy membership function, has a probabilistic flavour.

The formulae for the lower and upper set approximations can be generalized to some arbitrary level of precision $\pi \in \left(\frac{1}{2}, 1\right]$ by means of the rough membership function in the following way:

$$R_\pi^*(X) = \{ x \in U : \mu_X^R(x) \geq \pi \}$$

$$R_\pi^*(X) = \{ x \in U : \mu_X^R(x) > 1 - \pi \}$$

Note that the lower and upper approximations as originally formulated are obtained as a special case with $\pi = 1.0$.

Approximations of concepts are constructed on the basis of background knowledge. Obviously, concepts are also related to unseen so far objects. Hence it is very useful to define parameterized approximations with parameters tuned in the searching process for approximations of concepts. This idea is crucial for construction of concept approximations using rough set methods. For more information about the parameterized approximation spaces the reader is referred to [3].

Rough sets can thus approximately describe sets of patients, events, outcomes, etc. that may be otherwise difficult to circumscribe.

References

- [1] Z. Pawlak: Rough sets, Int. J. of Information and Computer Sciences, 11, 5, 341-356, 1982.
- [2] Z. Pawlak, A. Skowron: Rough membership function, in: R. E Yeager, M. Fedrizzi and J. Kacprzyk (eds.), Advances in the Dempster-Schafer of Evidence, Wiley, New York, 1994, 251-271.
- [3] A. Skowron, Z. Pawlak, J. Komorowski, L. Polkowski, A rough set perspective on data and knowledge, in W. Kloesgen, J. Żytkow (Eds.): Handbook of KDD. Oxford University Press, Oxford (2002), 134-149.

2. Rough Sets in Data Analysis

In this section we define basic concepts of rough set theory in terms of data, in contrast to general formulation presented in Section 1. This is necessary if we want to apply rough sets in data analysis.

2.1 Information Systems

A data set is represented as a table, where each row represents a case, an event, a patient, or simply an object. Every column represents an attribute (a variable, an observation, a property, etc.) that can be measured for each object; the attribute may be also supplied by a human expert or the user. Such table is called an *information system*. Formally, an *information system* is a pair $S = (U, A)$ where U is a non-empty finite set of *objects* called the *universe* and A is a non-empty finite set of *attributes* such that $a:U \rightarrow V_a$ for every $a \in A$. The set V_a is called the *value set* of a .

Example 2.1. Let us consider a very simple information system shown in Table 1. The set of objects U consists of seven objects: $x_1, x_2, x_3, x_4, x_5, x_6, x_7$, and the set of attributes includes two attributes: *Age* and *LEMS* (Lower Extremity Motor Score).

	Age	LEMS
x_1	16-30	50
x_2	16-30	0
x_3	31-45	1-25
x_4	31-45	1-25
x_5	46-60	26-49
x_6	16-30	26-49
x_7	46-60	26-49

Table 1

One can easily notice that objects x_3 and x_4 as well as x_5 and x_7 have exactly the same values of attributes. The objects are (pairwise) *indiscernible* using the available attributes.

In many applications there is an outcome of classification that is known. This *a posteriori* knowledge is expressed by one distinguished attribute called decision attribute; the process is known as supervised learning. Information systems of this kind are called *decision systems*. A *decision system* (a decision table) is any information system of the form $S = (U, A \cup \{d\})$, where $d \notin A$ is the *decision attribute*. The elements of A are called *conditional attributes* or simply *conditions*. The decision attribute may take several values though binary outcomes are rather frequent.

Example 2.2. Consider a decision system presented in Table 2. The table includes the same seven objects as in Example 2.1 and one decision attribute (*Walk*) with two values: Yes, No.

	<i>Age</i>	<i>LEMS</i>	<i>Walk</i>
x_1	16-30	50	Yes
x_2	16-30	0	No
x_3	31-45	1-25	No
x_4	31-45	1-25	Yes
x_5	46-60	26-49	No
x_6	16-30	26-49	Yes
x_7	46-60	26-49	No

Table 2

One may again notice that cases x_3 cases x_4 as well as x_5 and x_7 still have exactly the same values of conditions, but the first pair has different value of the decision attribute while the second pair has the same value.

2.2 Indiscernibility Relation

A decision system expresses all the knowledge about the model. This table may be unnecessarily large in part because it is redundant in at least two ways. The same or indiscernible objects may be represented several times, or some of the attributes may be superfluous. We shall look into these issues now.

Let $S = (U, A)$ be an information system, and $B \subseteq A$. A binary relation $IND_S(B)$ defined in the following way

$$IND_S(B) = \{(x, x') \in U^2 \mid \forall a \in B \ a(x) = a(x')\}$$

is called the *B-indiscernibility relation*. It is easy to see that $IND_S(B)$ is equivalence relation. If $(x, x') \in IND_S(B)$, then objects x and x' are indiscernible from each other by attributes from B . The equivalence classes of the *B-indiscernibility relation* are denoted $[x]_B$. The subscript S in the indiscernibility relation is usually omitted if it is clear which information system is meant.

Some extensions of standard rough sets do not require from a relation to be transitive (see, for instance, [8]). Such a relation is called tolerance relation or similarity.

Example 2.3. In order to illustrate how a decision system from Table 2 defines an indiscernibility relation, we consider the following three non-empty subsets of the conditional attributes: $\{Age\}$, $\{LEMS\}$ and $\{Age, LEMS\}$.

If we take into consideration the set $\{LEMS\}$ then objects x_3 and x_4 belong to the same equivalence class; they are indiscernible. From the same reason, x_5, x_6 and x_7 belong to another indiscernibility class. The relation IND defines three partitions of the universe.

$$\begin{aligned} IND(\{Age\}) &= \{\{x_1, x_2, x_6\}, \{x_3, x_4\}, \{x_5, x_7\}\}, \\ IND(\{LEMS\}) &= \{\{x_1\}, \{x_2\}, \{x_3, x_4\}, \{x_5, x_6, x_7\}\}, \\ IND(\{Age, LEMS\}) &= \{\{x_1\}, \{x_2\}, \{x_3, x_4\}, \{x_5, x_7\}, \{x_6\}\}. \end{aligned}$$

2.3. Set Approximation

In this subsection, we define formally the approximations of a set using the discernibility relation.

Let $S = (U, A)$ be an information system and let $B \subseteq A$, and $X \subseteq U$. Now, we can approximate a set X using only the information contained in the set of attributes B by constructing the B -lower and B -upper approximations of X , denoted $\underline{B}X$ and $\overline{B}X$ respectively, where $\underline{B}X = \{x \mid [x]_B \subseteq X\}$ and $\overline{B}X = \{x \mid [x]_B \cap X \neq \emptyset\}$.

Analogously as in a general case, the objects in $\underline{B}X$ can be with certainty classified as members of X on the basis of knowledge in B , while the objects in $\overline{B}X$ can be only classified as possible members of X on the basis of knowledge in B . The set $BN_B(X) = \overline{B}X - \underline{B}X$ is called the B -boundary region of X , and thus consists of those objects that we cannot decisively classify into X on the basis of knowledge in B . The set $U - \overline{B}X$ is called the B -outside region of X and consists of those objects which can be with certainty classified as do not belonging to X (on the basis of knowledge in B). A set is said to be *rough* (respectively *crisp*) if the boundary region is non-empty (respectively empty).

Example 2.4. Let $X = \{x \mid Walk(x) = Yes\}$, as given by Table 2. In fact, the set X consists of three objects: x_1, x_4, x_6 . Now, we want to describe this set in terms of the set of conditional attributes $A = \{Age, LEMS\}$. Using the above definitions, we obtain the following approximations: the A -lower approximation $\underline{A}X = \{x_1, x_6\}$, the A -upper approximation $\overline{A}X = \{x_1, x_3, x_4, x_6\}$, the A -boundary region $BN_A(X) = \{x_3, x_4\}$, and the A -outside region $U - \overline{A}X = \{x_2, x_5, x_7\}$. It is easy to see that the set X is rough since the boundary region is not empty. The graphical illustration of approximations of the set X together with the equivalence classes contained in the corresponding approximations are shown in Fig. 3.

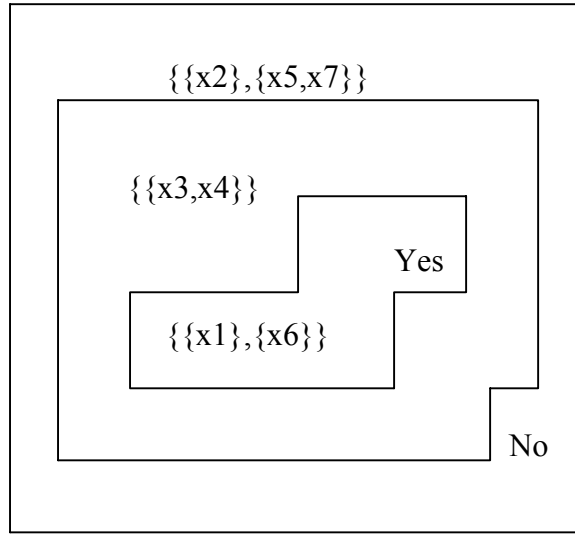


Fig. 3

One can easily the following properties of approximations:

1. $\underline{B}(X) \subseteq X \subseteq \overline{B}(X)$
2. $\underline{B}(\phi) = \overline{B}(\phi) = \phi, \underline{B}(U) = U$
3. $\overline{B}(X \cup Y) = \overline{B}(X) \cup \overline{B}(Y)$
4. $\underline{B}(X \cap Y) = \underline{B}(X) \cap \underline{B}(Y)$
5. $X \subseteq Y$ implies $\underline{B}(X) \subseteq \underline{B}(Y)$ and $\overline{B}(X) \subseteq \overline{B}(Y)$
6. $\underline{B}(X \cup Y) \supseteq \underline{B}(X) \cup \underline{B}(Y)$
7. $\overline{B}(X \cap Y) \subseteq \overline{B}(X) \cap \overline{B}(Y)$
8. $\underline{B}(-X) = -\overline{B}(X)$
9. $\overline{B}(-X) = -\underline{B}(X)$
10. $\underline{B}(\underline{B}(X)) = \overline{B}(\underline{B}(X)) = \underline{B}(X)$
11. $\overline{B}(\overline{B}(X)) = \underline{B}(\overline{B}(X)) = \overline{B}(X)$

where $-X$ denotes $U - X$.

It is easily seen that the lower and the upper approximations of a set, are respectively, the interior and the closure of this set in the topology generated by the indiscernibility relation.

One can define the following four basic classes of rough sets, i.e., four categories of vagueness:

1. A set X is *roughly B-definable*, iff $\underline{B}(X) \neq \phi$ and $\overline{B}(X) \neq U$,
2. A set X is *internally B-undefinable*, iff $\underline{B}(X) = \phi$ and $\overline{B}(X) \neq U$,

3. A set X is *externally B-undefinable*, iff $\underline{B}(X) \neq \emptyset$ and $\overline{B}(X) = U$,
4. A set X is *totally B-undefinable*, iff $\underline{B}(X) = \emptyset$ and $\overline{B}(X) = U$.

The intuitive meaning of this classification is the following.

A set X is *roughly B-definable* means that with the help of B we are able to decide for some elements of U that they belong to X and for some elements of U that they belong to $\neg X$.

A set X is *internally B-undefinable* means that using B we are able to decide for some elements of U that they belong to $\neg X$, but we are unable to decide for any element of U whether it belongs to X .

A set X is *externally B-undefinable* means that using B we are able to decide for some elements of U that they belong to X , but we are unable to decide for any element of U whether it belongs to $\neg X$.

A set X is *totally B-undefinable* means that using B we are unable to decide for any element of U whether it belongs to X or $\neg X$.

A rough set can be also characterized numerically by the following coefficient

$$\alpha_B(X) = \frac{|\underline{B}(X)|}{|\overline{B}(X)|}$$

called the *accuracy of approximation*, where $|X|$ denotes the cardinality of $X \neq \emptyset$.

Obviously $0 \leq \alpha_B(X) \leq 1$. If $\alpha_B(X) = 1$, X is *crisp* with respect to B (X is *precise* with respect to B), and otherwise, if $\alpha_B(X) < 1$ X is *rough* with respect to B (X is *vague* with respect to B).

Example 2.5. Let us consider a decision system shown in Table 3 in order to explain the above definitions.

<i>Patient</i>	<i>Headache</i>	<i>Muscle-pain</i>	<i>Temperature</i>	<i>Flu</i>
<i>p1</i>	<i>no</i>	<i>yes</i>	<i>high</i>	<i>yes</i>
<i>p2</i>	<i>yes</i>	<i>no</i>	<i>high</i>	<i>yes</i>
<i>p3</i>	<i>yes</i>	<i>yes</i>	<i>very high</i>	<i>yes</i>
<i>p4</i>	<i>no</i>	<i>yes</i>	<i>normal</i>	<i>no</i>
<i>p5</i>	<i>yes</i>	<i>no</i>	<i>high</i>	<i>no</i>
<i>p6</i>	<i>no</i>	<i>yes</i>	<i>very high</i>	<i>yes</i>

Table 3

Let $X = \{x : Flu(x) = Yes\} = \{p1, p2, p3, p6\}$ and the set of attributes $B = \{Headache, Muscle-pain, Temperature\}$. The set X is *roughly B-definable*, because $\underline{B}(X) = \{p1, p3, p6\} \neq \emptyset$ and $\overline{B}(X) = \{p1, p2, p3, p5, p6\} \neq U$. For this case we get $\alpha_B(X) = 3/5$. It means that the set X can be characterized partially employing symptoms (attributes) *Headache*, *Muscle-pain* and *Temperature*. Taking only one symptom $B = \{Headache\}$ we get $\underline{B}(X) = \emptyset$ and $\overline{B}(X) = U$, which means that the set X is *totally undefinable* in terms of attribute *Headache*, i.e., this attribute is not characteristic for the set X whatsoever. However,

taking single attribute $B = \{Temperature\}$ we get $\underline{B}(X) = \{p3, p6\}$ and $\overline{B}(X) = \{p1, p2, p3, p5, p6\}$, thus the set X is again roughly definable, but in this case we obtain $\alpha_B(X) = 2/5$, which means that the single symptom *Temperature* is less characteristic for the set X , than the whole set of symptoms, and patient $p1$ cannot be now classified as having flu in this case.

2.4 Rough Sets and Membership Function

In classical set theory, either an element belongs to a set or it does not. The corresponding membership function is the characteristic function for the set, i.e. the function takes values 1 and 0, respectively. In the case of rough sets, the notion of membership is different. The *rough membership function* quantifies the degree of relative overlap between the set X and the equivalence $[x]_B$ class to which x belongs. It is defined as follows:

$$\mu_x^B : U \rightarrow [0,1] \text{ and } \mu_x^B(x) = \frac{|[x]_B \cap X|}{|[x]_B|}.$$

Obviously $\mu_x^B(x) \in [0,1]$. A value of the membership function $\mu_x^B(x)$ is a kind of conditional probability, and can be interpreted as a degree of *certainty* to which x belongs to X (or $1 - \mu_x^B(x)$), as a degree of *uncertainty*).

The rough membership function can be used to define approximations and the boundary region of a set, as shown below:

$$\begin{aligned} \underline{B}(X) &= \{x \in U : \mu_x^B(x) = 1\}, \\ \overline{B}(X) &= \{x \in U : \mu_x^B(x) > 0\}, \\ BN_B(X) &= \{x \in U : 0 < \mu_x^B(x) < 1\}. \end{aligned}$$

The rough membership function has the following properties [6]:

1. $\mu_x^B(x) = 1$ iff $x \in \underline{B}(X)$,
2. $\mu_x^B(x) = 0$ iff $x \in \overline{B}(X)$,
3. $0 < \mu_x^B(x) < 1$ iff $x \in BN_B(X)$,
4. If $IND_s(B) = \{(x, x) : x \in U\}$, then $\mu_x^B(x)$ is the characteristic function of X ,
5. If $x IND_s(B) y$, then $\mu_x^B(x) = \mu_x^B(y)$ provided $IND_s(B)$,
6. $\mu_{U-x}^B(x) = 1 - \mu_x^B(x)$ for any $x \in U$,
7. $\mu_{X \cup Y}^B(x) \geq \max(\mu_x^B(x), \mu_Y^B(x))$ for any $x \in U$,
8. $\mu_{X \cap Y}^B(x) \leq \min(\mu_x^B(x), \mu_Y^B(x))$ for any $x \in U$.

The rough membership function can be interpreted as a frequency-based estimate of $\Pr(x \in X | u)$, the conditional probability that object x belongs to set X , given knowledge u of the information signature of x with respect to attributes B .

The formulae for the lower and upper set approximations can be generalized to some arbitrary level of precision $\pi \in \left(\frac{1}{2}, 1\right]$ by means of the rough membership function [6], as shown below.

$$\begin{aligned}\underline{B}_\pi X &= \{x : \mu_X^B(x) \geq \pi\} \\ \overline{B}_\pi X &= \{x : \mu_X^B(x) > 1 - \pi\}\end{aligned}$$

Note that the lower and upper approximations as originally formulated are obtained as a special case with $\pi = 1.0$.

Approximations of concepts are constructed on the basis of background knowledge. Obviously, concepts are also related to unseen so far objects. Hence, it is very useful to define parameterized approximations with parameters tuned in the searching process for approximations of concepts. This idea is crucial for construction of concept approximations using rough set methods.

2.5 Dependency of Attributes

An important issue in data is discovering dependencies between attributes. Intuitively, a set of attributes D depends totally on a set of attributes C , denoted $C \Rightarrow D$, if all values of attributes from D are uniquely determined by values of attributes from C . In other words, D depends totally on C , if there exists a functional dependency between values of D and C .

Formally, a functional dependency can be defined in the following way. Let D and C be subsets of A .

We will say that D depends on C in a degree k ($0 \leq k \leq 1$), denoted $C \Rightarrow D_k$

If

$$k = \gamma(C, D) = \frac{|POS_C(D)|}{|U|}$$

where $POS_C(D) = \bigcup_{X \in U/D} \underline{C}(X)$ called a *positive region* of the partition U/D with respect to C , is the set of all elements of U that can be uniquely classified to blocks of the partition U/D , by means of C .

Obviously

$$\gamma(C, D) = \sum_{x \in U/D} \frac{|\underline{C}(x)|}{|U|}$$

If $k=1$ we say that D depends totally on C , and if $k < 1$, we say that D depends partially (in a degree k) on C .

The coefficient k expresses the ratio of all elements of the universe, which can be property classified to blocks of the partition U/D , employing attributes C and will be called the *degree of the dependency*.

Example 2.6. Let us consider again a decision system shown in Table 3. For example, for dependency $\{Headache, Muscle-pain, Temperature\} \Rightarrow \{Flu\}$ we get $k = 4/6 = 2/3$, because

four out of six patients can be uniquely classified as having flu or not, employing attributes *Headache*, *Muscle-pain* and *Temperature*.

If we were interested in how exactly patients can be diagnosed using only the attribute *Temperature*, that is – in the degree of the dependence $\{Temperature\} \Rightarrow \{Flu\}$, we would get $k = 3/6 = 1/2$, since in this case only three patients $p3$, $p4$ and $p6$ out of six can be uniquely classified as having flu. In contrast to the previous case patient $p4$ cannot be classified now as having flu or not. Hence the single attribute *Temperature* offers worse classification than the whole set of attributes *Headache*, *Muscle-pain* and *Temperature*. It is interesting to observe that neither *Headache* nor *Muscle-pain* can be used to recognize flu, because for both dependencies $\{Headache\} \Rightarrow \{Flu\}$ and $\{Muscle-pain\} \Rightarrow \{Flu\}$ we have $k = 0$.

It can be easily seen that if D depends totally on C then $IND(C) \subseteq IND(D)$. This means that the partition generated by C is finer than the partition generated by D . Let us notice that the concept of dependency discussed above corresponds to that considered in relational databases.

Summing up: D is *totally (partially) dependent* on C , if employing C all (possibly some) elements of the universe U may be uniquely classified to blocks of the partition U/D .

2.6 Reduction of Attributes

In the Section 2.2 we investigated one of the natural aspects of reducing data which concerns identifying equivalence classes, i.e. objects that are indiscernible using the available attributes. In order to make some savings only one element of the equivalence class is needed to represent the entire class. The other aspect in data reduction is to keep only those attributes that preserve the indiscernibility relation and, consequently, set approximation. The rejected attributes are redundant since their removal cannot worsen the classification.

In order to express the above idea more precisely we need some auxiliary notions.

Let $S=(U,A)$ be an information system, $B \subseteq A$, and let $a \in B$.

We say that a is *dispensable* in B if $IND_S(B) = IND_S(B - \{a\})$; otherwise a is *indispensable* in B .

A set B is called *independent* if all its attributes are indispensable.

Any subset B' of B is called a *reduct* of B if B' is independent and $IND_S(B') = IND_S(B)$.

Hence, a reduct is a set of attributes that preserves partition. It means that a reduct is the minimal subset of attributes that enables the same classification of elements of the universe as the whole set of attributes. In other words, attributes that do not belong to a reduct are superfluous with regard to classification of elements of the universe.

There is usually several such subsets of attributes and those which are minimal are called reducts. Computing equivalence classes is straightforward. Finding a minimal reduct (i.e. reduct with a minimal number of attributes) among all reducts is *NP-hard* [6]. It is easy to see that the number of reducts of an information system with m attributes may be equal to

$$\left(\left\lfloor \frac{m}{2} \right\rfloor \right)$$

This means that computing reducts is not a trivial task. This fact is one of the bottlenecks of the rough set methodology. Fortunately, there exist good heuristics (e.g. [1],[7],[9],[10]) based on genetic algorithms that compute sufficiently many reducts in often acceptable time, unless the number of attributes is very high.

The reducts have several important properties. In what follows we will present two of them. First, we define a notion of a *core of attributes*.

Let B be a subset of A . The *core* of B is the set of all indispensable attributes of B .

The following is an important property, connecting the notion of the core and reducts

$$Core(B) = \bigcap Red(B),$$

where $Red(B)$ is the set of all reducts of B .

Because the core is the intersection of all reducts, it is included in every reduct, i.e., each element of the core belongs to some reduct. Thus, in a sense, the core is the most important subset of attributes, for none of its elements can be removed without affecting the classification power of attributes.

To further simplification of an information table we can eliminate some values of attribute from the table in such a way that we are still able to discern objects in the table as the original one. To this end we can apply similar procedure as to eliminate superfluous attributes, which is defined next.

We will say that the value of attribute $a \in B$, is *dispensable* for x , if $[x]_B = [x]_{B - \{a\}}$; otherwise the value of attribute a is *indispensable* for x .

If for every attribute $a \in B$ the value of a is indispensable for x , then B will be called *orthogonal* for x .

A subset $B' \subseteq B$ is a *value reduct* of B for x , iff B' is orthogonal for x and $[x]_B = [x]_{B'}$.

The set of all indispensable values of attributes in B for x will be called the *value core* of B for x , and will be denoted $CORE^x(B)$.

Also in this case we have

$$CORE^x(B) = \bigcap Red^x(B),$$

where $Red^x(B)$ is the family of all reducts of B for x .

Suppose we are given a dependency $C \Rightarrow D$. It may happen that the set D depends not on the whole set C but on its subset C' and therefore we might be interested to find this subset. In order to solve this problem we need the notion of a *relative reduct*, which will be defined and discussed next.

Let $C, D \subseteq A$. Obviously, if $C' \subseteq C$ is a D -reduct of C , then C' is a minimal subset of C such that $\gamma(C, D) = \gamma(C', D)$.

We will say that attribute $a \in C$ is *D-dispensable* in C , if $POS_C(D) = POS_{(C-\{a\})}(D)$; otherwise the attribute a is *D-indispensable* in C .

If all attributes $a \in C$ are C -indispensable in C , then C will be called *D-independent*.

A subset $C' \subseteq C$ is a *D-reduct* of C , iff C' is D -independent and $POS_C(D) = POS_{C'}(D)$.

The set of all D -indispensable attributes in C will be called *D-core* of C , and will be denoted by $CORE_D(C)$. In this case we have also the property

$$CORE_D(C) = \bigcap Red_D(C),$$

where $Red_D(C)$ is the family of all D -reducts of C . If $D = C$ we will get the previous definitions.

Example 2.7. In Table 3 there are two relative reducts with respect to Flu , $\{Headache, Temperature\}$ and $\{Muscle-pain, Temperature\}$ of the set of condition attributes $\{Headache, Muscle-pain, Temperature\}$. That means that either the attribute $Headache$ or $Muscle-pain$ can be eliminated from the table and consequently instead of Table 3 we can use either Table 4

<i>Patient</i>	<i>Headache</i>	<i>Temperature</i>	<i>Flu</i>
<i>p1</i>	<i>no</i>	<i>high</i>	<i>yes</i>
<i>p2</i>	<i>yes</i>	<i>high</i>	<i>yes</i>
<i>p3</i>	<i>yes</i>	<i>very high</i>	<i>yes</i>
<i>p4</i>	<i>no</i>	<i>normal</i>	<i>no</i>
<i>p5</i>	<i>yes</i>	<i>high</i>	<i>no</i>
<i>p6</i>	<i>no</i>	<i>very high</i>	<i>yes</i>

Table 4

or Table 5

<i>Patient</i>	<i>Muscle-pain</i>	<i>Temperature</i>	<i>Flu</i>
<i>p1</i>	<i>yes</i>	<i>high</i>	<i>yes</i>
<i>p2</i>	<i>no</i>	<i>high</i>	<i>yes</i>
<i>p3</i>	<i>yes</i>	<i>very high</i>	<i>yes</i>
<i>p4</i>	<i>yes</i>	<i>normal</i>	<i>no</i>
<i>p5</i>	<i>no</i>	<i>high</i>	<i>no</i>
<i>p6</i>	<i>yes</i>	<i>very high</i>	<i>yes</i>

Table 5

For Table 3 the relative core of with respect to the set $\{Headache, Muscle-pain, Temperature\}$ is the $Temperature$. This confirms our previous considerations showing that $Temperature$ is the only symptom that enables, at least, partial diagnosis of patients.

We will need also a concept of a *value reduct* and *value core*. Suppose we are given a dependency $C \Rightarrow D$ where C is relative D -reduct of C . To further investigation of the dependency we might be interested to know exactly how values of attributes from D depend on values of attributes from C . To this end we need a procedure eliminating values of attributes form C which does not influence on values of attributes from D .

We say that value of attribute $a \in C$, is D -dispensable for $x \in U$, if $[x]_C \subseteq [x]_D$ implies $[x]_{C-\{a\}} \subseteq [x]_D$; otherwise the value of attribute a is D -indispensable for x .

If for every attribute $a \in C$ value of a is D -indispensable for x , then C will be called D -independent (*orthogonal*) for x .

A subset $C' \subseteq C$ is a D -reduct of C for x (a value reduct), iff C' is D -independent for x and $[x]_{C'} \subseteq [x]_D$ implies $[x]_{C'} \subseteq [x]_D$.

The set of all D -indispensable for x values of attributes in C will be called the D -core of C for x (the value core), and will be denoted $CORE_D^x(C)$.

We have also the following property

$$CORE_D^x(C) = \bigcap Red_D^x(C),$$

where $Red_D^x(C)$ is the family of all D -reducts of C for x .

Using the concept of a value reduct, Table 4 and Table 5 can be simplified as follows:

<i>Patient</i>	<i>Headache</i>	<i>Temperature</i>	<i>Flu</i>
<i>p1</i>	<i>no</i>	<i>high</i>	<i>yes</i>
<i>p2</i>	<i>yes</i>	<i>high</i>	<i>yes</i>
<i>p3</i>	–	<i>very high</i>	<i>yes</i>
<i>p4</i>	–	<i>normal</i>	<i>no</i>
<i>p5</i>	<i>yes</i>	<i>high</i>	<i>no</i>
<i>p6</i>	–	<i>very high</i>	<i>yes</i>

Table 6

<i>Patient</i>	<i>Muscle-pain</i>	<i>Temperature</i>	<i>Flu</i>
<i>p1</i>	<i>yes</i>	<i>high</i>	<i>yes</i>
<i>p2</i>	<i>no</i>	<i>high</i>	<i>yes</i>
<i>p3</i>	–	<i>very high</i>	<i>yes</i>
<i>p4</i>	–	<i>normal</i>	<i>no</i>
<i>p5</i>	<i>no</i>	<i>high</i>	<i>no</i>
<i>p6</i>	–	<i>very high</i>	<i>yes</i>

Table 7

The following important property

$$B' \Rightarrow B - B', \text{ where } B' \text{ is a reduct of } B,$$

connects reducts and dependency.

Besides, we have:

$$\text{If } B \Rightarrow C, \text{ then } B \Rightarrow C', \text{ for every } C' \subseteq C,$$

in particular

$$\text{If } B \Rightarrow C, \text{ then } B \Rightarrow \{a\}, \text{ for every } a \in C.$$

Moreover, we have:

If B' is a reduct of B , then neither $\{a\} \Rightarrow \{b\}$ nor $\{b\} \Rightarrow \{a\}$ holds, for every $a, b \in B'$, i.e., all attributes in a reduct are pairwise independent.

2.7 Discernibility Matrices and Functions

In order to compute easily reducts and the core we can use discernibility matrix [6], which is defined below.

Let $S=(U,A)$ be an information system with n objects. The discernibility matrix of S is a symmetric $n \times n$ matrix with entries c_{ij} as given below.

$$c_{ij} = \{a \in A \mid a(x_i) \neq a(x_j)\} \text{ for } i, j = 1, \dots, n$$

Each entry thus consists of the set of attributes upon which objects x_i and x_j differ. Since discernibility matrix is symmetric and $c_{ii} = \emptyset$ (the empty set) for $i=1, \dots, n$. Thus, this matrix can be represented using only elements in its lower triangular part, i.e. for $1 \leq j < i \leq n$.

With every discernibility matrix one can uniquely associate a discernibility function defined below.

A *discernibility function* f_s for an information system S is a Boolean function of m Boolean variables a_1^*, \dots, a_m^* (corresponding to the attribute a_1, \dots, a_m) defined as follows.

$$f_s(a_1^*, \dots, a_m^*) = \bigvee \{ \exists c_{ij}^* \mid 1 \leq j < i \leq n, c_{ij} \neq \emptyset \}$$

where $c_{ij}^* = \{a^* \mid a \in c_{ij}\}$. The set of all prime implicants² of f_s determines the set of all reducts of A .

Example 2.8 Using the above definitions for the information system S from Example 2.5 (Table. 3), we obtain the following discernibility matrix presented in Table 8 and discernibility function presented below.

	$p1$	$p2$	$p3$	$p4$	$p5$	$p6$
$p1$						
$p2$	H, M					
$p3$	H, T	M, T				
$p4$	T, F	H, M, T, F	H, T, F			
$p5$	H, M, F	F	M, T, F	H, M, T		
$p6$	T	H, M, T	H	T, F	H, M, T, F	

Table 8

² An implicant of a Boolean function f is any conjunction of literals (variables or their negations) such that if the values of these literals are true under an arbitrary valuation v of variables then the value of the function f under v is also true. A prime implicant is a minimal implicant (with respect to the number of its literals). Here we are interested in implicants of monotone Boolean functions only, i.e. functions constructed without negation (see Subsection ???).

In this table H, M, T, F denote *Headache*, *Muscle-pain*, *Temperature* and *Flu*, respectively.

The discernibility function for this table is

$$\begin{aligned}
 f_s(H, M, T, F) = & (H \vee M)(H \vee T)(T \vee F)(H \vee M \vee F)T \\
 & (M \vee T)(H \vee M \vee T \vee F)F(H \vee M \vee T) \\
 & (H \vee T \vee F)(M \vee T \vee F)H \\
 & (H \vee M \vee T)(T \vee F) \\
 & (H \vee M \vee T \vee F)
 \end{aligned}$$

where \vee denotes the disjunction and the conjunction is omitted in the formula.

Let us also notice that each row in the above discernibility function corresponds to one column in the discernibility matrix. This matrix is symmetrical with the empty diagonal. Each parenthesized tuple is a conjunction in the Boolean expression, and where the one-letter Boolean variables correspond to the attribute names in an obvious way. After simplification, the discernibility function using laws of Boolean algebra we obtain the following expression

HTF,

which says that there is only one reduct $\{H, T, F\}$ in the data table and it is the core.

Relative reducts and core can be computed also using the discernibility matrix, which needs slight modification

$$c_{ij} = \{a \in C : a(x_i) \neq a(x_j) \text{ and } w(x_i, x_j)\},$$

where $w(x_i, x_j) \equiv x_i \in POS_C(D)$ and $x_j \notin POS_C(D)$ or

$x_i \notin POS_C(D)$ and $x_j \in POS_C(D)$ or

$x_i, x_j \in POS_C(D)$ and $(x_i, x_j) \notin IND(D)$

for $i, j = 1, 2, \dots, n$.

If the partition defined by D is definable by C then the condition $w(x_i, x_j)$ in the above definition can be reduced to $(x_i, x_j) \notin IND(D)$.

Thus, entry c_{ij} is the set of all attributes which discern objects x_i and x_j that do not belong to the same equivalence class of the relation $IND(D)$.

If we instead construct a Boolean function by restricting the conjunction to only run over column k in the discernibility matrix (instead of over all columns), we obtain the so-called *k-relative discernibility function*. The set of all prime implicants of this function determines the set of all *k-relative reducts* of A . These reducts reveal the minimum amount of information needed to discern $x_k \in U$ (or, more precisely, $[x_k] \subseteq U$) from all other objects.

Example 2.9 Considering the information system S from Example 2.5 as a decision system, we can illustrate the above considerations by computing relative reducts for the set of attributes $\{Headache, Muscle-pain, Temperature\}$ with respect to *Flu*. The corresponding discernibility matrix is shown in Table 9.

	$p1$	$p2$	$p3$	$p4$	$p5$	$p6$
$p1$						
$p2$						
$p3$						
$p4$	T	H, M, T				
$p5$	H, M		M, T			
$p6$				T	H, M, T	

Table 9

The discernibility function for this table is

$$T(H \vee M)(H \vee M \vee T)(M \vee T).$$

After simplification the discernibility function we obtain the following expression

$$TH \vee TM,$$

which represents two reducts TH and TM in the data table and T is the core.

2.8 Decision Rule Synthesis

The reader has certainly realized that the reducts (of all the various types) can be used to synthesize *minimal* decision rules. Once the reducts have been computed, the rules are easily constructed by overlaying the reducts over the originating decision table and reading off the values.

Example 2.10 Given the reduct $\{Headache, Temperature\}$ in Table 4, the rule read off the first object is "if *Headache* is *no* and *Temperature* is *high* then *Flu* is *yes*".

We shall make these notions precise.

Let $S = (U, A \cup \{d\})$ be a decision system and let $V = \bigcup \{V_a : a \in A\} \cup V_d$. Atomic formulae over $B \subseteq A \cup \{d\}$ and V are expressions of the form $a = v$; they are called *descriptors* over B and V , where $a \in B$ and $v \in V_a$. The set $F(B, V)$ of formulae over B and V is the least set containing all atomic formulae over B and V and closed with respect to the propositional connectives \wedge (conjunction), \vee (disjunction) and \neg (negation).

Let $\varphi \in F(B, V)$. $\|\varphi_A\|$ denotes the meaning of φ in the decision table A which is the set of all objects in U with the property φ . These sets are defined as follows:

1. if φ is of the form $a = v$ then $\|\varphi_A\| = \{x \in U \mid a(x) = v\}$
2. $\|\varphi \wedge \varphi'_A\| = \|\varphi_A\| \cap \|\varphi'_A\|$; $\|\varphi \vee \varphi'_A\| = \|\varphi_A\| \cup \|\varphi'_A\|$; $\|\neg \varphi_A\| = U - \|\varphi_A\|$

The set $F(B, V)$ is called the set of *conditional formulae* of A and is denoted $C(B, V)$.

A *decision rule* for A is any expression of the form $\varphi \Rightarrow d = v$, where $\varphi \in C(B, V)$, $v \in V_d$ and $\|\varphi_A\| \neq \emptyset$. Formulae φ and $d = v$ are referred to as the predecessor and the successor of decision rule $\varphi \Rightarrow d = v$.

Decision rule $\varphi \Rightarrow d = v$ is *true* in A if, and only if, $\|\varphi_A\| \subseteq \|d = v_A\|$; $\|\varphi_A\|$ is the set of objects *matching* the decision rule; $\|\varphi_A\| \cap \|d = v_A\|$ is the set of objects *supporting* the rule.

Example 2.11 Looking again at Tab. 4, some of the rules are, for example:

$(Headache = no) \wedge (Temperature = high) \Rightarrow (Flu = yes)$,

$(Headache = yes) \wedge (Temperature = high) \Rightarrow (Flu = yes)$,

The first rule is true in Tab. 4 while the second one is not true in that table.

Several numerical factors can be associated with a synthesized rule. For example, the support of a decision rule is the number of objects that match the predecessor of the rule. Various frequency-related numerical quantities may be computed from such counts like the *accuracy coefficient* equal to

$$\frac{\|\|\varphi_A\| \cap \|d = v_A\|\|}{\|\|\varphi_A\|\|}$$

For a systematic overview of rule synthesis see e.g. [1], [9], [10].

Remark. The rough set theory has found many applications in medical data analysis, finance, voice recognition, image processing and others. However the approach presented in this section is too simple to many real-life applications and was extended in many ways by various authors. The detailed discussion of the above issues can be found in [5], [7] and the internet (e.g., <http://www.rsds.wsiz.rzeszow.pl>)

References

- [1] J.W.Grzymała-Busse (1997): A new version of the rule induction system LERS. *Fundamenta Informaticae* 31, pp. 27-39.
- [2] Z. Pawlak: Rough sets, *International Journal of Computer and Information Sciences*, 11, 341-356, 1982.
- [3] Z. Pawlak: *Rough Sets – Theoretical Aspects of Reasoning about Data*, Kluwer Academic Publishers, Boston, London, Dordrecht, 1991.
- [4] Z. Pawlak, A. Skowron: Rough membership functions, in: R. R Yaeger, M. Fedrizzi and J. Kacprzyk (eds.), *Advances in the Dempster Shafer Theory of Evidence*, John Wiley & Sons, Inc., New York, Chichester, Brisbane, Toronto, Singapore, 1994, 251-271.
- [5] L. Polkowski: *Rough Sets – Mathematical Foundations*, *Advances in Soft Computing*, Physica-Verlag, Springer-Verlag Company, 2002, 1-534.
- [6] A. Skowron, C. Rauszer: The discernibility matrices and functions in information systems, in: R. Słowiński (ed.), *Intelligent Decision Support. Handbook of Applications and Advances of the Rough Set Theory*, Kluwer Academic Publishers, Dordrecht, 1992, 311-362.
- [7] A. Skowron et al: Rough set perspective on data and knowledge, *Handbook of Data Mining and Knowledge Discovery* (W. Klösgen, J. Żytkow eds.), Oxford University Press, 2002, 134-149.
- [8] A. Skowron, L. Polkowski, J. Komorowski: Learning Tolerance Relations by Boolean Descriptors, *Automatic Feature Extraction from Data Tables*. In S. Tsumoto, S. Kobayashi, T. Yokomori, H. Tanaka, and A. Nakamura (Eds.) (1996), *Proceedings of the Fourth International Workshop on Rough Sets, Fuzzy Sets, and Machine Discovery (RSFD'1996)*, The University of Tokyo, November 6-8, pp. 11-17.

- [9] A. Skowron (1995): Synthesis of adaptive decision systems from experimental data. In A. Aamodt and J. Komorowski (Eds.)(1995), Proc. Fifth Scandinavian Conference on Artificial Intelligence. Trondheim, Norway, May 29-31, Frontiers. In: Artificial Intelligence and Applications 28, IOS Press, pp. 220-238.
- [10] J. Stefanowski (1998): On rough set based approaches to induction of decision rules. In L. Polkowski, A. Skowron (Eds.): Rough Sets in Knowledge Discovery 1. Methodology and Applications. Physica-Verlag, Heidelberg 1998, pp. 500-529.

3. Rough Sets and Concurrency

Knowledge discovery and data mining [3],[5] is one of important and current research problems. Discovering relationships between data is the main task of so called machine learning [12].

The aim of this section is to present an overview of a new methodology of discovering concurrent data models and decision algorithms from experimental data tables. This work is a continuation of a new research direction concerning relationships between rough set theory and concurrency. Such research direction was started by Z. Pawlak in 1992 [17] and later developed intensively by many authors (see e.g. [6],[8],[13-15],[20],[23-24],[25-29],[30]). In the section, we consider the following two main problems: (1) discovering concurrent data models from data tables; (2) discovering decision algorithms from data tables. Foundation of the proposed approach are rough sets [16], fuzzy sets [1] and Petri nets [11]. The preprocessing of data [3] and Boolean reasoning [2] are also used. The proposed methodology is a result of the author and his coauthors long term research [6],[8],[13-15],[20],[23-24],[25-29],[30].

These problems are current and very important not only with the respect to cognitive aspect but also to its possible applications. Discovering concurrent systems models from experimental data tables is very interesting and useful for a number of application domains, in particular, in Artificial Intelligence, e.g. by speech recognition [3], in biology and molecular biology, for example, for obtaining the answer concerning the following question: How is the model of cell evolution dependent on changes of the gene code (see e.g. [31], pp. 780-804)? Discovering decision algorithms from data tables is very important for real-time applications in such areas as real-time decision making by groups of intelligent agents (e.g. robots) [19], navigation of intelligent mobile robots [4], searching information in centralized or distributed data bases [5] and, in general, in real-time knowledge-based control systems [21].

3.1 Discovering concurrent data models from data tables

This subsection includes an informal description of the first problem in question, together with a general scheme of its solution. The examples of this problem's possible applications are also indicated. We assume that the reader is familiarized with the basic notions of rough set theory [16] and Petri nets [11].

Problem 1. *Let an experimental data table be given. Construct a concurrent data model on the base of knowledge extracted from a given data table in such a way that its global states are consistent with the extracted knowledge.*

This problem we can interpret as the synthesis problem of concurrent systems specified by experimental data tables. We assume that the knowledge extracted from a given data table includes information on the structure as well as the behavior of the modeled system.

The solution

In order to solve this problem, we propose to realize the following three main stages:

Stage 1. Data Representation.

We assume that the data table S (an information system in Pawlak's sense [16]) representing experimental knowledge is given. It consists of a number of rows labeled by elements from the set of objects U , which contain the results of sensor measurements

represented by the value vector of attributes from A . Values of attributes we interpret as states of local processes in the modeled system. However, the rows of data table we interpret as global states of the system. Sometimes, it is necessary to transform the given experimental data table by taking into account other relevant features (new attributes) instead of the original ones. This step is necessary when the result concurrent model constructed directly from the original data table yields a very large model or when the complexity of model synthesis from the original table is too high. In this case some additional time is necessary to compute the values of new features, after the results of sensor measurements are given. The input for our methodology consists of the data table (if necessary, preprocessed in a way as described above).

In order to represent data, apart from information systems, it is also possible to use dynamical information systems introduced in [28] and specialized tables (i.e., matrices of forbidden states and matrices of forbidden transitions) [14] in our considerations. We assume that information systems include knowledge on global states of modeled concurrent systems only, however, dynamical information systems additionally accumulate the knowledge on the next state relation (i.e., transitions between global states in a given system). The specialized tables include information both on which global states of the modeled system and which transitions between global states are forbidden. This representation is, in a sense, a dual representation of dynamical information systems. In the following, we use only information systems as models for experimental data tables. The proposed generalizations can be also applied to the remaining data representations mentioned above.

In general, data tables obtained from the measurements are relatively large. Besides, such data is usually incomplete, imprecise and vague. Rough set methods allow to reduce the large amounts of source data, as well as pre-process and analyze it. Moreover, computer tools for data analysis based on rough set methodology are available (see e.g. [9]). This is the reason, why we use these methods.

Stage 2. *Knowledge representation.*

First, we generate a set of rules (deterministic and non-deterministic) by applying rough set methods in the form if ... then from a given data table S . These rules represent the knowledge (usually partial) on the modeled system. We consider rules true in a degree CF ($0 < CF \leq 1$) in S , where CF is the number equal to the ratio of the number of objects from U matching the left and right hand side of the rule, and the number of objects from U matching only the left hand side of the rule. The number CF is called the certainty factor of the rule. We assume that there is an object u matching the left hand side of the rule. If $CF = 1$ for a given rule, then we say that the rule is deterministic; otherwise, it is non-deterministic. Next, using the result set of rules (or the result concurrent model described below) we can get much more information on the behavior of the modeled system, constructing so called an extension (in a degree CF) of a given information system. An extension S' (in a degree CF) of a given information system S is created by adding to S all new global states (rows) of S which are consistent with all rules true (in a degree CF) in S . If we use only a set of deterministic rules generated from S , then we can construct so called maximal consistent extension of S .

Stage 3. *Transformation of data tables into colored Petri nets.*

Now, we transform a set of rules obtained in Stage 2 into a concurrent model represented in the form of a colored Petri net with the following property: the reachability set of the result net corresponds one-to-one to an extension of a given information system. The construction of a colored Petri net for the given data table consists of three steps (levels). Firstly, a net representing the set of components (functional modules) of a given information is constructed. Any component (a subset of rules or an information subsystem) in a sense

represents the strongest functional module of the system. The components of the system are computed means of its reducts [25]. Secondly, a net defined by the set of rules of a given information system, corresponding to all nontrivial dependencies (connections or links) between the values of attributes belonging to different components of the information system, to the net obtained in the first step is added. The connections between components represent constraints which must be satisfied when these functional modules coexist in the system. The components together with the connections define so called covering of the system. In general, there are many coverings of a given information system. It means that for a given information system we can construct its several concurrent models presenting the same behavior but a different internal structure. Thirdly, the elements (places, transitions and arcs) of the net defined in steps 1-2 are additionally described according to the definition of a colored Petri net. The modular approach described above makes the appropriate construction of a net much clearer. Moreover, the application of colored Petri nets to represent concurrent systems allows to obtain coherent and clear models (also hierarchical ones – this aspect is omitted in the paper) suitable for further computer analysis and verification [10].

Such approach allows the adaptation of structures of complex concurrent systems to the new conditions, changing in time, re-engineering (reconstruction) of structures of systems organization together with optimization of reconstruction costs, and adapting the systems organization to new requirements [29].

Example 3.1. Consider an information system $S = (U, A)$ where a set of objects $U = \{u_1, u_2, u_3, u_4\}$, a set of attributes $A = \{a, b\}$ and the values of the attributes are defined as in Table 10.

$U \setminus A$	a	b
u_1	0	1
u_2	1	0
u_3	0	2
u_4	2	0

Table 10

By applying methods for generating rules in minimal form, i.e., with a minimal number of descriptors on its left hand side, described in [24], we obtain the following set of rules for the system S : if $a=1$ then $b=0$, if $a=2$ then $b=0$, if $b=1$ then $a=0$, if $b=2$ then $a=0$, all of them are true in degree $CF=1$ in S . For generating rules from the given information system, we can also use specialized computer tools, e.g. [9].

After applying the standard Boolean algebra laws to the given set of rules, we obtain the following Boolean expression: $(a=0 \text{ AND } b=0) \text{ OR } (a=0 \text{ AND } b=1) \text{ OR } (a=0 \text{ AND } b=2) \text{ OR } (a=1 \text{ AND } b=0) \text{ OR } (a=2 \text{ AND } b=0)$. Using the computed Boolean expression, we can construct the guard expression corresponding to rules presented above. It has the following form: $[ya = (a=0) \text{ AND } yb = (b=0) \text{ OR } ya = (a=0) \text{ AND } yb = (b=1) \text{ OR } ya = (a=0) \text{ AND } yb = (b=2) \text{ OR } ya = (a=1) \text{ AND } yb = (b=0) \text{ OR } ya = (a=2) \text{ AND } yb = (b=0)]$.

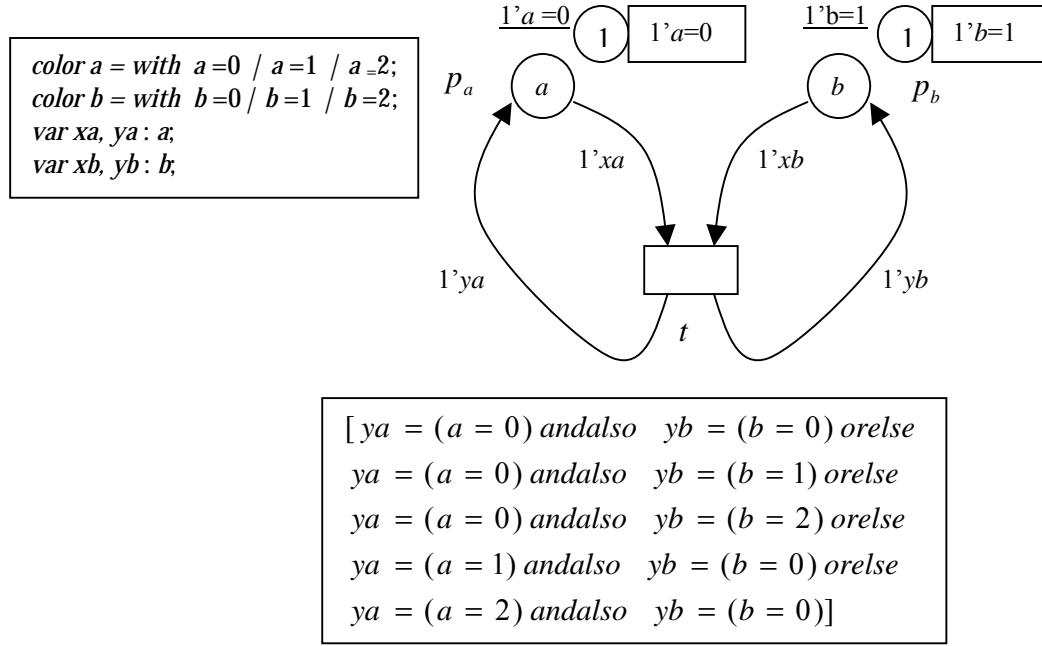


Fig. 4

The concurrent model of S in the form of colored Petri net constructed by using our approach is shown in Fig. 4. The guard expression form associated with the transition t (see Fig. 4) differs slightly from the one presented above. It follows the formal requirements imposed by the syntax of the CPN ML language implemented into the Design/CPN system [10].

The set of markings of the constructed net corresponds to all global states consistent with all rules true in degree $CF=1$ in S . It is easy to see that the maximal consistent extension S' of S additionally includes the value vector $(0,0)$. This vector is consistent with all rules true in S . Thus, the set of all value vectors corresponding to S is equal to $\{(0,1), (1,0), (0,2), (2,0), (0,0)\}$.

3.2 Discovering decision algorithms from data tables

This section presents an informal description of the second problem considered here together with the general scheme of its solution. The examples of possible applications of this problem are also indicated.

Problem 2. *Let a decision data table be given. Construct a concurrent decision algorithm on the base of the knowledge extracted from a given decision data table in such a way that: (1) its computation leading to decision making has minimal length and/or satisfies other additional analysis criteria.*

In the following, we consider this problem as the modeling problem of approximate reasoning process on the base of knowledge included into experimental decision tables with uncertain, imprecise and vague information. We choose the timed approximate Petri nets defined in [8] as a concurrent model for the constructed decision algorithm.

The solution

The proposed solution of this problem is obtained by realizing the following three stages:

Stage 1. Data Representation.

We assume that the decision table S (a decision system in Pawlak's sense [16]) representing experimental knowledge is given. It consists of a number of rows labeled by elements from a set of objects U which contain the results of sensor measurements represented by the value vector of conditional attributes (conditions) from A together with the decision d corresponding to this vector. The decision is given by a domain expert. Values of conditions are identified by sensors in a finite, but unknown number of time units. Sometimes, it is necessary to transform the given experimental decision table in a similar way as in Problem 1, Stage 1. This step is necessary when the decision algorithm constructed directly from the original decision table yields an inadequate classification of unseen objects, or when the complexity of decision algorithm synthesis from the original decision table is too high. The input for our algorithm consists of the decision table (preprocessed, if necessary). In the paper, we consider the values of attributes to be crisp [16] or fuzzy [1].

Stage 2. Knowledge representation.

We assume that the knowledge encoded in S is represented by rules automatically extracted from S , using the standard rough set methods for rules generation. We consider two kinds of rules: conditional and decision ones. Rules of the first kind express some relationships between values of conditions. However, rules of the second kind express some relationships between values of conditions and decision. Besides, each of such rules can be deterministic or non-deterministic. The rule is active if the values of all attributes on its left hand side have been measured. An active rule which is true (in a degree CF) in S can be used to predict the value of the attribute on its right hand side even if its value has not been measured, yet. Our concurrent model of a decision algorithm propagates information from sensors (attributes) to other attributes, as soon as possible. It is done by using rules in minimal form, i.e. with a minimal number of descriptors on its left hand side. We use the method for generating rules minimal and true in S , described in [24].

Stage 3. Transformation of decision tables into timed approximate Petri nets.

The construction of a timed approximate Petri net for the given decision table consists of four steps (levels). Each step of the net construction provides one module of the constructed net. At first, the places representing the set of all conditions of the given decision table are constructed. Then, the fragments of the net, defined by a set of conditional rules generated from the given decision table are added, to the places obtained in the first step. Next, the net obtained in the second step is extended by adding fragments of the net defined by the set of decision rules generated from the given decision table. Finally, the elements (places, transitions and arcs) of the net defined in steps 1-3 are additionally described according to the definition of a timed approximate Petri net. The modular approach described above makes the appropriate construction of a net much clearer.

The constructed timed approximate Petri net allows to make a decision, as soon as a sufficient number of attribute values is known as a result of measurements and conclusions drawn from the knowledge encoded in S . It is easy to prove that any of its computations leading to decision making has minimal length, i.e., no prediction of the proper decision based on the knowledge encoded in S and the measurement of attribute values is possible before the end of the computation. Each step of the computation of the constructed timed approximate Petri net consists of two phases. In the first phase, checking is performed to see whether some new values of conditions have been identified by sensors, and in the second phase, new information about values is transmitted through the net at high speed. The whole process is realized by implementation of the rules true (in a degree CF) in the given decision table.

It is worth to point out that the proposed methodology can be used to model the decision processes considered by Pawlak in [18], in an easy way.

Example 3.2. Consider a decision system $S = (U, A \cup \{e\})$ where the set of objects $U = \{u_1, u_2, u_3, u_4, u_5, u_6\}$, and the set of conditions $A = \{a, b, c, d\}$. The decision is denoted by e . The possible values of conditions and the decision from S are defined as in Table 11.

$U \setminus A \cup \{e\}$	a	b	c	d	e
u_1	1	1	1	1	0
u_2	1	1	2	1	1
u_3	2	0	1	2	0
u_4	2	0	2	2	0
u_5	2	0	1	1	0
u_6	2	0	2	1	0

Table 11

Using standard rough set methods for generating rules and computing certainty factors of rules, we compute all (deterministic and non-deterministic) decision and conditional rules from S together with appropriate certainty factors. For the sake of the paper's length, we consider only a sample of rules for the decision system S . It is as follows: **r1**: if $a=1$ then $e=0$ ($CF=0.5$, non-deterministic decision rule), **r2**: if $b=1$ then $e=0$ ($CF=0.5$, non-deterministic decision rule), **r3**: if $a=1$ AND $c=2$ then $e=1$ ($CF=1$, deterministic decision rule), **r4**: if $a=2$ OR $b=0$ OR $c=1$ OR $d=2$ then $e=0$ ($CF=1$, deterministic decision rule), **r5**: if $b=1$ AND $c=2$ then $e=1$ ($CF=1$, deterministic decision rule), **r6**: if $d=1$ then $e=0$ ($CF=0.75$, non-deterministic decision rule), **r7**: if $a=1$ OR $b=1$ then $d=1$ ($CF=1$, deterministic conditional rule).

The timed approximate Petri net corresponding to these rules is presented in Fig. 5. The net construction is realized according to the approach described above. More detailed information on the algorithm for transforming rules representing the given decision table into an approximate Petri net can be found in [6]. Complete description of the method for transforming rules into the timed approximate Petri net will be presented in the full version of this paper. At present, we give an intuitive explanation of such a net construction, taking into account our example.

In the timed approximate Petri net from Fig. 5, places p_a, p_b, p_c, p_d represent the conditions a, b, c, d from S , respectively. However, the place p_e represents the decision e . The transitions t_1, \dots, t_6 represent the rules **r1**, ..., **r6**, respectively. Transition t_7 represents the rule **r7**.

Time values associated with the transitions of the net are defined as follows: 1 corresponds to transitions t_1, t_2, t_6 ; 2 corresponds to transitions t_3, t_5, t_7 ; 4 corresponds to the transition t_4 . Bi-directional input/output arcs in the net check only whether a suitable transition in the net is enabled at a given marking. After firing that transition the marking of input/output places of transition does not change. The color sets (types) corresponding to places p_a, p_b, p_c, p_d are defined as follows: $a = \{a=1, a=2\}$, $b = \{b=0, b=1\}$, $c = \{c=1, c=2\}$, $d = \{d=1, d=2\}$, $e = \{e=0, e=1\}$, respectively.

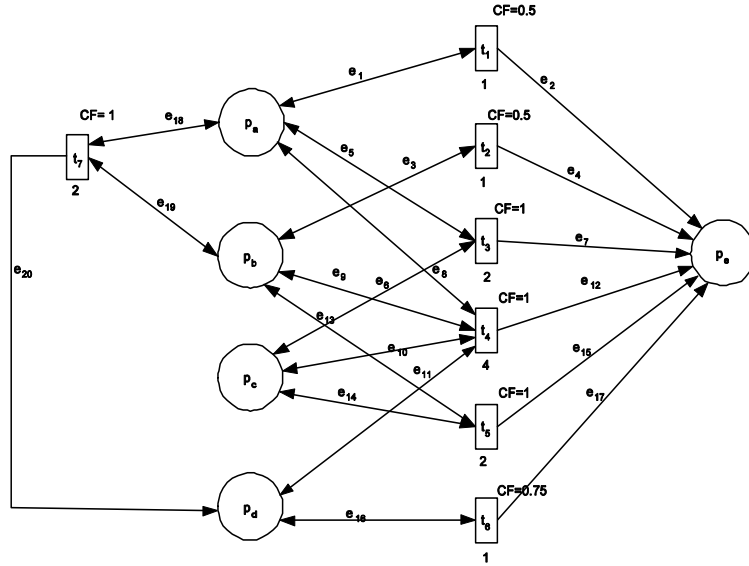


Fig. 5

The time stamps carried by tokens for places p_a, p_b, p_c, p_d are equal to 7,7,12,10, respectively. The operator AND is associated with transitions t_3, t_5 , and OR – with transitions t_4, t_7 . Time stamps and operators are omitted in the figure. Additionally, in our net model we assume that the truth values for all propositions (descriptors) associated with places are equal to 1 (true), as well as the threshold values are set to 0 for all transitions. Moreover, in the net any transition can fire only once at a given simulation session.

Place\Time	$g=0s$	$g=8s$	$g=9s$	$g=10s$	$g=14s$
p_a	\emptyset	$1/a_1@7$	$1/a_1@7$	$1/a_1@7$	$1/a_1@7$
p_b	\emptyset	$1/b_1@7$	$1/b_1@7$	$1/b_1@7$	$1/b_1@7$
p_c	\emptyset	\emptyset	\emptyset	\emptyset	$1/c_2@12$
p_d	\emptyset	\emptyset	$1/d_1@9$	$1/d_1@10$	$1/d_1@10$
p_e	\emptyset	$0.5/e_0@8$	$0.5/e_0@8$	$0.75/e_0@11$	$0.75/e_0@11$ $+1/e_1@14$

Table 12

We assume that the initial marking of each place in the net at a global clock $g = 0s$ is equal to the empty set. The example of an approximate reasoning process realized in the net model presented in Fig. 5 is shown in Table 12. The sign @ in the marking of a place denotes a time stamp. From Fig. 5 and Table 12 it follows that, for example, when the value of global clock is equal to $g = 8s$, then places p_a , and p_b are marked. At the same time, transitions t_1 and t_2 are ready and they fire. Further, at the time $g = 8s$, we can make the first decision, i.e., the place p_e is marked. In Table 12, we can also observe that for different time moments we obtain different markings of the place p_e . Analyzing the approximate reasoning process in the net model, we can consider different paths of its execution. As a consequence, we can choose the most appropriate path of the reasoning process in the net model, i.e., satisfying some imposed requirements (criteria).

Remarks. The described data model of concurrent systems discovered from a given information system allows to understand better the structure and behavior of the modeled

system. Due to this approach, it is possible to represent the dependencies between the processes in information system and their dynamic interactions in graphical way. The presented approach can be treated as a kind of decomposition of a given information system. Besides, our methodology can be applied for automatic feature extraction. The components and the connections between components in the system can be interpreted as new features of the modeled system. Properties of the constructed concurrent systems (e.g. their invariants) can be understood as higher level laws of experimental data. As a consequence, this approach seems to be useful also for state identification in real-time.

The presented concurrent model of a decision algorithm allows a comprehensive analysis of approximate reasoning process described by the given decision table, among others, a very quick identification of objects in the given decision table. The proposed net model allows also to analyze the performance of approximate reasoning process. Using the time approximate nets we can answer such questions as, how much time we need to make a decision, or which execution path in the approximate reasoning process should be chosen in order to make a decision as quick as it is possible. We have two self-implemented computer tools used for the verification of proposed methods and algorithms [7],[13].

References

- [1] Bandemer, H., Gottwald, S.: Fuzzy Sets, Fuzzy Logic, Fuzzy Methods with Applications, Wiley, New York 1995.
- [2] Brown, E.M.: Boolean Reasoning, Kluwer Academic Publishers, Dordrecht 1990.
- [3] Cios, J.K., Pedrycz, W., Świniarski, R.W.: Data Mining. Methods for Knowledge Discovery, Kluwer Academic Publishers, Boston 1998.
- [4] Crowley, J.L.: Navigation for an Intelligent Mobile Robot, IEEE Journal of Rob. Auto. RA-1, 1985, 31-41.
- [5] Fayyad, Usama M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, Ramasamy (Eds.), Advances in Knowledge Discovery and Data Mining, The AAAI Press, Menlo Park, CA, 1996.
- [6] Fryc, B., Pancierz, K., Suraj, Z.: Approximate Petri Nets for Rule-Based Decision Making, in: [31], pp. 733-742.
- [7] Fryc, B., Makara, Z., Pancierz, K., Suraj, Z.: A Petri Net System, in: Proceedings of the Workshop on Theory and Applications of Soft Computing (TASC'04), L. Polkowski (Ed.), Warsaw, Poland, November 26, 2004, Polish-Japanese Institute of Information Technology, Warsaw 2004 (to appear).
- [8] Fryc, B., Suraj, Z.: Timed Approximate Petri Nets, in: Proceedings of the Workshop on Theory and Applications of Soft Computing (TASC'04), L. Polkowski (Ed.), Warsaw, Poland, November 26, 2004, Polish-Japanese Institute of Information Technology, Warsaw 2004 (to appear).
- [9] <http://logic.mimuw.edu.pl/rses>
- [10] <http://www.daimi.au.dk/designCPN/>
- [11] Jensen, K.: Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use, Vol. 1, Springer-Verlag, Berlin 1997.
- [12] Kodratoff, Y., Michalski, R. (Eds.): Machine Learning, Vol. 3, Morgan Kaufmann Publishers, San Mateo, CA, 1990.
- [13] Pancierz, K., Suraj, Z.: Discovering Concurrent Models from Data Tables with the ROSECON System. Fundamenta Informaticae, Vol. 60 (1-4), IOS Press, Amsterdam 2004, 251-268.

- [14] Pancierz, K., Suraj, Z.: On Some Approach to Restricted-Based Concurrent System Design, in: Proceedings of the International Workshop on Concurrency, Specification and Programming (CS&P'2004), Vol. 1, H.D. Burkhard, L. Czaja, G. Lindemann, A. Skowron, H. Schlingloff, Z. Suraj (Eds.), Caputh, Germany, September 24-26, 2004, Humboldt University, Berlin 2004, 112-123..
- [15] Pancierz, K., Suraj, Z.: Automated Discovering of Concurrent Models from Data Tables: An Overview, in: Proceedings of the 3rd ACS/IEEE International Conference on Computer Systems and Applications (AICCSA-05), Cairo, Egypt, January 3-6, 2005, American University in Cairo, 2005 (to appear).
- [16] Pawlak, Z.: Rough Sets - Theoretical Aspects of Reasoning About Data, Kluwer Academic Publishers, Dordrecht 1991.
- [17] Pawlak, Z.: Concurrent Versus Sequential the Rough Sets Perspective, Bulletin of the EATCS, 48, 1992, 178-190.
- [18] Pawlak, Z.: Flow Graphs, their Fusion, and Data Analysis, in: A. Jankowski, A. Skowron, M. Szczuka (Eds.), Proceedings of the International Workshop on Monitoring, Security and Rescue Techniques in Multiagent Systems (MSRAS 2004), Płock, Poland, June 7-9, 2004, 3-4.
- [19] Payton, D.W., Bihari, T.E.: Intelligent Real-Time Control of Robotic Vehicles, Comm. ACM, 1991, Vol. 34-8, 48-63.
- [20] Peters, J.F., Skowron, A., Suraj, Z., Pedrycz, W., Ramanna, S.: Aproximate Real-Time Decision Making: Concepts and Rough Fuzzy Petri Net Models, International Journal of Intelligent Systems, Vol. 14, John Wiley & Sons, Inc., New York 1999, 805-839.
- [21] Schoppers, M.: Real-Time Knowledge-Based Control Systems, Comm. ACM, Vol. 34- 8, 1991, 26-30.
- [22] Skowron, A., Rauszer, C.: The discernibility matrices and functions in information systems, in: R. Słowiński (Ed.), Intelligent Decision Support. Handbook of Applications and Advances of Rough Set Theory, Kluwer Academic Publishers, Dordrecht 1992, 331-362.
- [23] Skowron, A., Suraj, Z.: Rough Sets and Concurrency, Bulletin of the Polish Academy of Sciences, Vol. 41, No. 3, 1993, 237-254.
- [24] Skowron, A., Suraj, Z. (1996): A Parallel Algorithm for Real-Time Decision Making: A Rough Set Approach, Journal of Intelligent Information Systems 7, Kluwer Academic Publishers, Dordrecht, 5-28.
- [25] Suraj, Z.: Discovery of Concurrent Data Models from Experimental Tables: A Rough Set Approach. Fundamenta Informaticae, Vol. 28, No. 3-4, IOS Press, Amsterdam 1996, 353-376.
- [26] Suraj, Z.: An Application of Rough Set Methods to Cooperative Information Systems Reengineering, in: S. Tsumoto, S. Kobayashi, T. Yokomori, H. Tanaka (Eds.), Proceedings of the Fourth International Workshop on Rough Sets, Fuzzy Sets, and Machine Discovery (RSFD'96), Tokyo, Japan, November 6-8, 1996, 364-371.
- [27] Suraj, Z.: Reconstruction of Cooperative Information Systems under Cost Constraints: A Rough Set Approach, Information Sciences: An International Journal 111, Elsevier Inc., New York 1998, 273-291.
- [28] Suraj, Z.: The Synthesis Problem of Concurrent Systems Specified by Dynamic Information Systems, in: L. Polkowski, A. Skowron (Eds.), Rough Sets in Knowledge Discovery, 2, Physica-Verlag, Berlin 1998, 418-448.
- [29] Suraj, Z.: Rough Set Methods for the Synthesis and Analysis of Concurrent Processes, in: L. Polkowski, S. Tsumoto, T.Y. Lin (Eds.), Rough Set Methods and Applications, Springer, Berlin 2000, 379-488.

- [30] Suraj, Z., Pancierz, K.: A Synthesis of Concurrent Systems: A Rough Set Approach, in: G. Wang, Q. Liu, Y. Yao, A. Skowron (Eds.), Proceedings of the 9th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing (RSFDGrC'2003), Chongqing, China, May 26-29, 2003, Lecture Notes in Artificial Intelligence, Vol. 2639, Springer-Verlag, Berlin-Heidelberg 2003, 299-302.
- [31] Tsumoto, S., Słowiński, R., Komorowski, J., Grzymala-Busse, J.W. (Eds.), Proceedings of the 4th International Conference on Rough Sets and Current Trends in Computing (RSCTC'2004), Uppsala, Sweden, June 1-5, 2004, Lecture Notes in Artificial Intelligence, Vol. 3066, Springer-Verlag, Berlin Heidelberg, 2004, 733-742.