© International Journal of Combinatorial Optimization Problems and Informatics, Vol. 6, No. 3, Sep-Dec 2015, pp. 5-21. ISSN: 2007-1558.

Reduction of Decision Rules for Project Explanation on Public Project Portfolio

Laura Cruz-Reyes[1], César Medina-Trejo[2], Fernando Lopez-Irarragorri[3], Gilberto Rivera[4], Claudia G. Gómez S.[1], Mercedes Pérez-Villafuerte[1].
[1]Tecnológico Nacional de México: Instituto Tecnológico de Ciudad Madero, México, [2]Tecnológico Nacional de México: Instituto Tecnológico de Tijuana, México, [3]Universidad Autónoma de Nuevo León, México, [4]Universidad Autónoma de Sinaloa, México.

lauracruzreyes@itcm.edu.mx, cesarmedinatrejo@gmail.com, flopez65@gmail.com, riveragil@gmail.com, cggs71@hotmail.com, pvmercedes@gmail.com

Abstract. The selection of a public project portfolio among a set of good portfolios directly impacts public organizations in a decisive manner. This task depends on the criteria of the decision maker and is especially hard when the solution is evaluated on many objectives. This combinational optimization problem has been addressed by multicriteria optimization algorithms which do not generate a single solution, but instead they generate a set of good solutions in the Pareto frontier. In this paper we propose to aid the recommendation of public projects with the utilization of simplified decision rules to explain the construction of the recommended project portfolio. Due to most decision problems can be represented in decision tables, we can form a decision table — in portfolio selection problem — with rules defined by the features of the projects (condition attributes) and the decision of support (the decision attribute). To facilitate the recommendation process, we simplify the decision rules via a hybrid rough set-based method that combines a genetic algorithm with an exact method. Less decision rules help the decision maker to faster analyze why some projects are accepted or rejected in a particular portfolio and —by this mean— get more certainty about his/her decision. With the proposed approach, there is a significant reduction of the attributes of the decision rules with a high accuracy of classification, which was proven on a set of the UCI repository for machine learning test.

Keywords: Project Portfolio Problem, Decision Table, Decision Rules, Hybrid System.

1. Introduction

The selection of the best public project portfolio is critical, the reason is that public projects help society (e.g. the financing of street lighting, the construction of a hospital or highway repairs, to name a few) and a bad choice can have a negative impact on the portfolio. The portfolio selection is a complex combinatorial optimization problem because of its multiobjective characteristics. By this reason, multicriteria algorithms are used to generate a set of good solutions, where each solution is a portfolio considered for recommendation.

The best portfolio is selected by an entity that can be a person or a group of people, called the Decision Maker (DM). For simplicity, we consider hereinafter that the DM is only one person. (S)He selects the portfolio depending on the cognitive thinking his/her preferences. Thus, different DMs are likely to select different portfolios Therefore, it is important to explain — in a simple form— the criterion used for the construction of the set of recommended portfolios. There is a rising demand to explain and justify a set of solutions to a DM in *decision support systems* [1], which focus on guiding the user to make a decision that satisfies him/her.

A DM needs a way to justify his/her decision, it require easy to understand explanations to eliminate the uncertainty generated by having more than one portfolio presented in addition to an explication that contains the resume of the information of the projects in the construction of the portfolios. The generation of explanations is an important feature of the decision support tools [2].

Received Oct 20, 2015 / Accepted Dec 17, 2015

The decision tables are a way of organizing information, where each row is a decision rule. Here, every project of a portfolio can be represented as a decision rule, composed of the characteristics of the project and the decision on supporting it or not. These rules are usually expressed in the form of "if-then-else" rules that can help people to perform or understand decisions. The number of resulting decision rules after the analysis of the input instance can be very large If so, it is extremely important to have only a small set of decision rules with reduced clauses (attributes) that only contains the information needed to appreciate more rapidly the characteristics of the projects in the portfolio (either rejected or accepted).

The decision rule reduction in the area of rough set theory is conceptually similar to attribute selection in Machine Learning. There has been a growing demand on attribute reduction as a consequence of the high dimensionality of the data in different fields. For example, there are applications on the health department [3] and chemical industry [4] that generate decision rules using rough set theory to assist in the correct attribute selection. There are also works [5] that do not focus on a specific application but on a general framework to be used on different scenarios.

In this paper we propose a method to assist the DM in the task of selecting only one public project portfolio. We present the DM the recommended solution (from the set of good solutions) and an explanation generated with the use of rough set theory to reduce the decision rules derived from decision tables. This set of simplified rules will be used as a knowledge base to generate generic arguments to interact with the DM in a dialog game as proposed in Cruz-Reyes et al. [6]. To best of our knowledge, the use of decision rules for the portfolio selection problem has been discussed only for R&D projects [7], where these rules are used to describe the funding assignment policy of a portfolio.

This paper is organized as follow: Section 2 shows an introduction of some basic concepts needed to understand the public portfolio problem and the proposed approach to address it. Section 3 describes the proposed methodology to generate simplified decision rules based on rough sets. In Section 4 we show the experimental results against the ones from the literature, and finally, in Section 5, our conclusions and future work are presented.

2. Background

In this section is shown the theoretical foundations on which this paper is sustained.

2.1 Public Portfolio Problem

A project is a unique, unrepeatable and temporal process, which seeks to accomplish a specific set of objectives. A set of projects that can be done in the same time period is called a portfolio [8]. However, institutions or organizations usually do not have sufficient resources to support all projects that are proposed. In such circumstances, the difficulty is to choose the set of projects that provide the greatest benefit.

Projects on the same portfolio share the available funds of the organization. The DM is responsible for the selection of the portfolio among the set of portfolios that were recommended.

The public portfolio problem (PPP) is defined as follows [9]: a total budget to fund projects represented by B, a set of projects denoted by N, the portfolio is modeled as $x = \langle x_1, x_2, ..., x_N \rangle$ which is a binary vector where a value of 1 in the i-th project means it will be supported, otherwise the project is not supported. Each project has an associated cost and area (e.g. education, health) denoted by c_i and a_i respectively, where every area j has a lower budget limit and an upper one, which are represented by L_j and U_j respectively. Every project has a corresponding geographical region, which also have the lower and upper budget limits.

Each project i is represented by a p-dimensional vector $f(i) = \langle f_1(i), f_2(i), f_3(i), ..., f_p(i) \rangle$, each $f_k(i)$ represents the impact of project i to the objective k in a problem with p objectives. A portfolio is feasible if it does not exceed the entire budget and the constraint for each area j.

The public projects need special handling because they have certain specific characteristics, Fernandez et al. [10][11] list the following:

- The exact requirements for a project to be financed are not known with certainty, there are estimates only.

- The effects of the impact of these projects on society generally are not on the short term, because of this characteristic, their benefits are difficult to qualify and measure. Thus, one must consider the impact on the individuals of the society and keep in mind the economic benefits that would provide to the society.
- Projects are qualified regularly taking into account multiple conflicting criteria.

The selection of a public project portfolio has many attributes in conflict. Because of the nature of the problem, it has been approached by multicriteria algorithms, but most of these algorithms generates a set of solutions that are on the *Pareto frontier*, which is the collection of non-dominated optimal portfolio in PPP, The DM must choose only one of the set of good solutions (this problem is the choice problematic of a single action (P.α) formulated by Bernard Roy in [12]). Such a decision depends on the base of the DM's preferences Here, an explication is necessary to justify the automatic selection of a portfolio and verify if this selection satisfies the DM criteria.

2.2 Genetic Algorithms

The genetic algorithms belong to the evolutionary algorithms, which generate solutions —or individuals— to optimization problems using techniques based on natural evolution. This particular algorithm mimics natural selection of species, in which the fittest survive and the unfit die. In these algorithms, the fittest individuals have a high probability to find another individual to produce an offspring to pass on a future generation; whilst the less fit individuals have little to almost no probability of being chosen. This probability results on the genetics of the best individuals pass on directly to the new generation, these new generations —according to natural selection— are increasing their fitness in order to adapt to the environment. In genetic algorithms, individuals are representations of possible solutions. The basic principles for a genetic algorithm to function properly are described by Beasley [13]:

1) **Coding.** The codification of the chromosome, represented by genes, need to have the parameters to construct the possible candidate solution.
2) **Fitness function.** A fitness function is assigned to each chromosome for classification, which is a numeric value that represents how suitable is the individual for the problem to be solved.
3) **Reproduction.** On the reproduction phase, individuals are selected to be recombined and populate the next generation. For the recombination a crossover between two individuals is performed, e.g., *one-point crossover* (in which a random point is selected to cut the chromosome creating two segments, a head and a tail, an exchange is made to generate a new individual, from genetic information of both parents). To introduce new genetic information a *mutation* is produced with a low probability, changing a gene on the chromosome.
4) **Convergence.** In each generation as the population evolves with the successive generations, the fitness function of the best chromosomes will increase toward the global optimum. A gene has reached convergence, i.e., if certain percentage of the current population has this gene in the same position.

2.3 Rough Set Theory

Rough set theory is a tool to address the imperfect, vague and imprecise knowledge [14]. It has characteristics in common with the Dempster-Shafer theory of evidence [15]. Rough sets complement the fuzzy sets, as they address the uncertainty in different ways. The difference between rough sets and the other approaches is that the rough sets use the upper and lower approximations as the main tool and no prior information of the data is needed. Dempster-Shafer theory uses belief functions and the probability distribution of the data is required. In addition, the probability value that the fuzzy sets theory needs is not necessary in rough set theory [16].

In the rough set theory all necessary information is obtained from the original data, no additional information is necessary. This feature is supported in the conjecture that the available information contains additional knowledge about the universe of discourse. The basis of the whole rough set theory is founded on the *indiscernibility relation*, which describes the objects that cannot be discerned with the available information: if the objects have the same information in theirs attributes, they cannot be distinguished between them. These indiscernible objects are grouped into sets which constitute elementary sets, and the knowledge of the universe is constructed based on such elementary sets.

In an information table there may exist objects which cannot be classified into one concept of interest (it is unknown if these objects belong or not to the concept). Rough set theory addresses the uncertainty of the concepts by proposing two accurate concepts known as approximations, the *upper* and *lower approximations*. The former includes objects that probably can be classified into the concept of interest, and the latter cover the objects that clearly are classified on the concept of interest. Rough

set theory address the vagueness of information with a *boundary region* (according to the difference between the upper approximation with the lower approximation), unlike fuzzy sets that approach the vagueness of information with a membership function.

The major employment of the rough set theory includes the following [17]:

a) Attribute reduction, which is an important data preprocessing step;
b) rule generation, a process to obtain relevant criteria for decision; and
c) prediction and recommendation based on historical data.

Given the characteristics of the rough set theory, Pawlak [18] concluded that it can help with:

a) Introducing efficient algorithms to locate hidden patterns in the data.
b) Evaluation of the meaning of the data.
c) A simple formulation to understand.
d) Direct interpretation of the obtained results.

The basic definitions are outlines in the next paragraphs, based on the definitions described by Pawlak [18,19,20].

Indiscernibility relation. Denotes a relationship between objects that we cannot discern with the available information because the set of objects to compare have the same information in the considered attributes. For an easier comprehension, the basic theory of rough sets is defined in the next sections with respect of a data example shown in Table 1.

A data set is the foundation of the data analysis based on rough sets. This data set is called an information system, which is a data table with columns form by attributes and rows consisting of objects of interest. In the particular case of a project portfolio problem, the rows are the projects requiring evaluation to determine if they are part of the portfolio, whilst the columns are formed by the attributes that describe the features of the projects.

In a formal way, an information system is the pair *IS*= (*U*, *A*), where *U* is the universe of finite and non-empty sets (the projects in PPP) and *A* is a finite set of attributes [19]. For any attribute $a \in A$, there is an associate set V_a of its attribute values. The table entries are in pairs (*x*, *a*), where $x \in U$ and $a \in A$. For any subset *B* of *A*, there is a binary relation *I*(*B*) on *U*, called an indiscernibility relation, represented by:

$$(x, y) \in I(B) \text{ if and only if } a(x) = a(y) \text{ for every } a \in B, \quad \textbf{(1)}$$

where $a(x) \in V_a$, is the attribute value *a* of the object of interest *x* on the information system. *I*(*B*) is an equivalence relation, if (*x*, *y*) belongs to *I*(*B*) then *x* and *y* are indiscernible in relation to *B*. The equivalence classes formed by *I*(*B*) are the basic knowledge about the information system. They are called *B*-elementary sets or *B*-granules.

Table 1. An inconsistent decision table for project support.

Objects U (Projects)	Attributes A				
	Impact	Poverty	Middle Class	High Class	Supported
1	High	Yes	No	Yes	Yes
2	Medium	Yes	No	Yes	Yes
3	Low	Yes	Yes	No	No
4	High	No	No	Yes	Yes
5	Low	Yes	Yes	No	Yes
6	High	Yes	Yes	Yes	Yes

Using data from Table 1, calculating $I(B) = I(\{Impact, Poverty, Middle\ Class, High\ Class\})$ yields the elementary sets {1}, {2}, {3, 5}, {4}, {6}, we cannot discern the project that belongs in the same elementary sets, i.e., from the elementary set {3, 5} the project 3 and 5 cannot be differentiated using the information obtained from every attribute of *B*.

Reducts. On the information systems, there are very often attributes that can be removed and still conserve the same primary properties of the original data, formally a *reduct* is defined by the subset of attributes *B* such that $B \subseteq A$ and $I(B) = I(A)$, where $I(B)$ and $I(A)$ are the indiscernibility relations established by *B* and *A*. The reducts are used to reduce the information systems by the removal of unessential attributes. Finding a minimal reduct (that is, finding the reduct of minimal length) is an NP-hard problem, and the problem of generating all reducts is exponential [21].

Using the indiscernibility relation the redundant attributes can be determined. From the data in the example shown in Table 1, we have removed one attribute at a time and calculated the indiscernibility relation, resulting in the following:

a) $I(\{Impact, Poverty, Middle\ Class, High\ Class\})=\{1\}, \{2\}, \{3, 5\}, \{4\}, \{6\}$;
b) $I(\{Poverty, Middle\ Class, High\ Class\}) = \{1, 2\}, \{3, 5\}, \{4\}, \{6\}$;
c) $I(\{Impact, Middle\ Class, High\ Class\}) = \{1, 4\}, \{2\}, \{3, 5\}, \{6\}$;
d) $I(\{Impact, Poverty, High\ Class\}) = \{1, 6\}, \{2\}, \{3, 5\}, \{4\}$;
e) $I(\{Impact, Poverty, Middle\ Class\}) = \{1\}, \{2\}, \{3, 5\}, \{4\}, \{6\}$.

In this case, the only subset of attributes whose indiscernibility relation is the same as the set of a) is the subset of e), that means we can remove the attribute High Class because it is redundant and the resulting reduct is shown on Table 2.

Table 2. A reduct obtained from Table 1.

Objects U (Projects)	Attributes A			
	Impact	Poverty	Middle Class	Supported
1	High	Yes	No	Yes
2	Medium	Yes	No	Yes
3	Low	Yes	Yes	No
4	High	No	No	Yes
5	Low	Yes	Yes	Yes
6	High	Yes	Yes	Yes

The decision table from Table 1 is inconsistent since the projects 3 and 5 are conflicting, having the same attributes values and different decision, rough sets deals with the problem of inconsistencies with a concept called approximation.

Aproximations. The partition constructed from *B*, represented by U / B, is the family of equivalence classes $I(B)$, and the block of the partition where *x* is contained is represented by $B(x)$.

There exist instances in which the information at hand is not sufficient to classify all the objects with certainty due of the granularity of the knowledge from the information system. For these situations there are two precise crisp sets, defined on sets $X \subseteq U$ [18]:

$$B^*(X) = \{x \in U : B(x) \cap X \neq \varnothing\}, \quad (2)$$

$$B_*(X) = \{x \in U : B(x) \subseteq X\}. \quad (3)$$

$B^*(X)$ and $B_*(X)$ are known as the *B*-upper and *B*-lower approximation of *X*, respectively. The lower approximation is the set of all objects which surely can be classified into the concept of interest with certainty, and the upper approximation and the set

of all objects that can possibly belong to the concept of interest. With these two approximations the *B*-boundary region of *X* can be defined as the difference between the upper and lower approximations, formally:

$$BN_B(X) = B^*(X) - B_*(\mathrm{X}). \tag{4}$$

The boundary region is the set of all the objects which cannot be classified in either the object of interest or its complement; in the case —when $BN_B(X) \neq \varnothing$ — the set is called rough for being inexact, and the case when $BN_B(X) = \varnothing$ the set is called crisp because it is exact.

Returning to Table 1, we calculate the lower and upper approximation and the boundary region with respect to $X = (\{1,\ 2,\ 4,\ 5,\ 6\})$ because the attribute of interest is *Support* = Yes:

a) $B^*(X) = B^*(\{1,\ 2,\ 4,\ 5,\ 6\}) = \{1,\ 2,\ 3,\ 4,\ 5,\ 6\}$;

b) $B_*(X) = B_*(\{1,\ 2,\ 4,\ 5,\ 6\}) = \{1,\ 2,\ 4,\ 6\}$;

c) $BN_B(X) = BN_B(\{1,\ 2,\ 4,\ 5,\ 6\}) = \{1,\ 2,\ 3,\ 4,\ 5,\ 6\} - \{1,\ 2,\ 4,\ 6\} = \{3,\ 5\}$ }, the boundary region is not empty; ergo, this is a rough set because it is inexact.

Decision Tables. A decision table is an information system which clearly can be divided into two disjoint classes of attributes: the condition and decision attributes. It is formally represented by *IS* (*U*, *C*, *D*), where *U* is the universe of discourse, *C* is the set of condition attributes and *D* is the set of decision attributes, where $C \cup D = A$.

Table 1 shows an example of a decision table that have to do with project support. *Impact*, *Poverty*, *Middle Class* and *High Class* are the condition attributes, and *Support* is the decision attribute. In this case, the object of interest as before is if the project is supported, namely the attribute *Support* with the value "Yes". We want to give an explanation of characteristics of the favored projects, in this case the set of objects {1, 2, 4, 5, 6}.

With the current decision table, the projects 3 and 5 cannot be classified with certainty as supported or not. Because those projects have the same condition attributes and a different decision attribute, they can possibly be classified as supported or not, and such a condition entails an inconsistent data table. Whilst projects 1, 2, 4 and 6 can be classified —with certainty— as supported.

Decision Rules. The decision rules are used to outline the correspondence between the condition attributes with the decision attributes. The decision rules are frequently used to show the knowledge derive from a decision table.

Each object or fact is taken as a decision rule, which are frequently stated as logical expressions [22] of the form IF-THEN-ELSE. The first part of the expression contains the decision attributes with its respective values, and the THEN expression is the decision attributes, i.e., the decision rule of the first object of Table 1 is: "IF (*Impact*=High and *Poverty*=Yes and *Middle Class*=No and *High Class*=Yes) THEN (*Support*=Yes)". Decision rules are easy to understand; thus they are used commonly as a technique of knowledge representation.

However, the attributes of an information system can be frequently removed. By such a reduction, the decision rules can be easier to understand as a result of a less number of "IF" clauses in each rule. Here, the most challenging issue is to find the minimum reduct that helps to further reduce the clauses.

Rough Set Frameworks. Several frameworks for the study of the rough set theory have been developed, like ROSE (Rough Set Data Explorer), ROSETTA (Rough Set Toolkit for Analysis of Data) and RSES (Rough Set Exploration System). The last framework is a free software tool that is considered to provide a stable platform for experimentation with data, such as exploration, classification support and knowledge discovery. Riza et.al [23] made an overview and comparison of these frameworks, and we briefly describe below one of them.

The framework RSES contains a library of methods based on the rough set theory. RSES algorithms are categorized into two groups: (1) algorithms to handle data structures of the library, and (2) algorithms focused on operations based on the rough set theory. The main types of algorithm — for experimentation — in this software are [24]:

- Reduction algorithms to calculate the reducts for a given decision table. The reducts can be exhaustively computed or by means of approximation algorithms and heuristic approaches. With the calculated reducts, the decision rules can be constructed.
- Discretization algorithms to transform the values of a decision without losing its classification characteristic.
- Classification algorithms that use the decision rules to obtain a decision value.

3. Proposed Methodology

In this section, the proposed methodology for obtaining project explanations on public project portfolio is detailed. Figure 1 shows the complete methodology, ranging from obtaining preferential information to the calculation of the reduced decision rules. An overview of the process is presented as follow:

1. First, we have to obtain the preferential information of the organization, these preferences are necessary for the selection of the appropriate multicriteria algorithm.
2. With the correct selection of the multicriteria method using the preferential information, we obtain the set of portfolios to be presented, which is reduced to a set that matches the DM's preferences. One of these multicriteria methods is presented by Fernandez [25], which generates a set of good solutions in the region of interest of the DM.
3. In the recommendation subsystem, there is a set containing the possible portfolios to recommend for the DM, such portfolios are used by the reduct generation algorithm.
4. A transformation of the portfolios to a decision table is performed. This tabular representation is necessary to work with the rough set method. The calculation of the reducts is made from the decision table in order to generate a reduced decision table through reducts generated with a genetic algorithm.
5. By means of an exact method, we work with the new reduced decision table to make the number of attributes in the rows decrease (value reduction). The number of attributes is diminished, rule by rule, reducing the quantity of clauses in each rule.
6. Using a small set of rules on the projects conforming the portfolio (which projects were supported and which were not), we can summarize the information about the construction of the portfolio and verify how close or far the portfolio is to the policies of the organization. This is a critical issue in helping the DM to make a decision.
7. The process of recommending a justified portfolio may terminate when the DM is faced with the justification. At this point, (s)he can make an introspection by seeing the summary of the characteristics of the projects in the recommended portfolio, and —on the basis of such information— there is a possibility that the DM is not satisfied with this portfolio. The DM may update his/her preferences and make reassessment the attribute values of some projects. The option to perform these updates is currently a work in progress, the process should be repeated if the DM changes his/her preferences, and —by this mean— a new justification (on how the new recommended portfolio is conformed) is obtained.

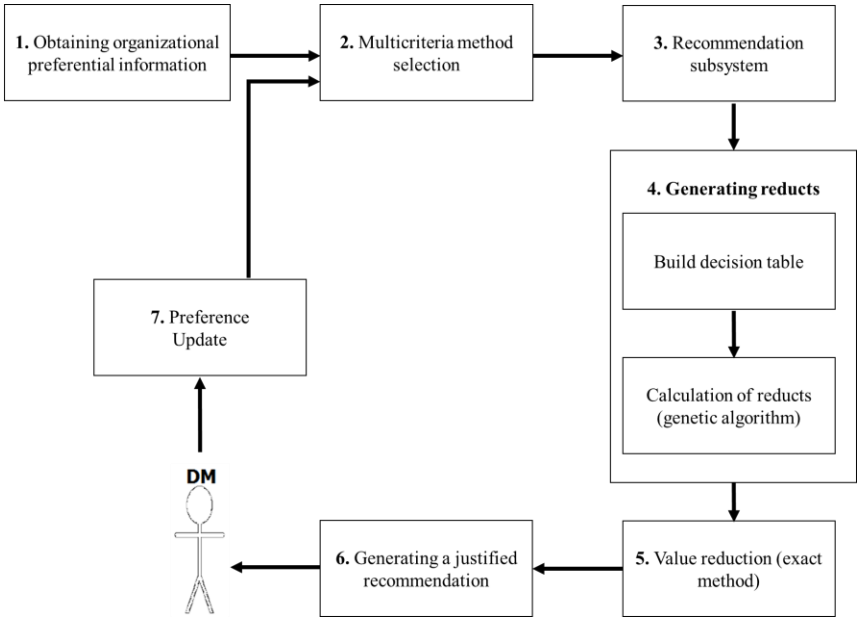

Fig. 1. Proposed methodology for a project portfolio recommendation with simplified decision rules.

The main contribution of this methodology is the simplification of decision rules via a hybrid algorithm that consists of two phases:

a) a genetic algorithm to generate the reducts, and
b) an exhaustive calculation of the decision rules with an exact method, using the reduced decision table generated by the genetic algorithm.

3.1 Generating Reducts

An information table with all its attributes can be very extensive and —as a consequence— hard to follow. To mitigate this drawback, it is important to identify those attributes whose information could be removed without affecting the effectiveness for classifying. With this objective in mind, a genetic algorithm is used to the calculation of reduced decision tables. Both the fitness function and the concept of distinction table were taken from Wróblewski [26], which performs a transformation of an information table into a new table with a binary structure in order to formulate the fitness function of the genetic algorithm, as described in Section 3.2.

Table 3. A consistent information table.

Projects	Attributes				
	Impact	Poverty	Middle Class	High Class	Support
P_1	High	Yes	No	Yes	Yes
P_2	Medium	Yes	No	Yes	Yes
P_3	Low	Yes	Yes	No	No
P_4	High	No	No	Yes	Yes

The distinction table is a binary matrix of dimensions $\left(\frac{m^2-m}{2}\right)\times(N+1)$; here, the first dimension refers to all possible pairs of projects to compare, where *m* is the number of projects in the information table; and the second dimension refers to the number of attributes, where *N* is the number of condition attributes (the decision attribute is also considered in the term "plus one"). A distinction table entry is of the form $\mathrm{a}((j,k),i)$, which is the comparison for the pair of projects (x_j, x_k) in its attribute a_i. Thus, the entries are formulated as follows:

$$\text{for } i \in \{1,2,3,...,N\}: \tag{5}$$

$$\mathrm{a}((j,k),i) = \begin{cases} 1 & \text{if } a_i(x_j) \neq a_i(x_k), \\ 0 & \text{if } a_i(x_j) = a_i(x_k), \end{cases}$$

$$\mathrm{a}((j,k),N+1) = \begin{cases} 1 & \text{if } a_i(x_j) = a_i(x_k), \\ 0 & \text{if } a_i(x_j) \neq a_i(x_k). \end{cases}$$

An example of the structure of a distinction table is shown in Table 4, which is the result of using Equation 5 on the information table presented in Table 3. The constructed genetic algorithm works with the newly created distinction table of binary inputs.

Table 4. The resulting distinction table of transforming the decision table shown on Table 1.

Projects Pairs	Attributes				
	Impact	Poverty	Middle Class	High Class	Support
$P_{1,}\ P_{2}$	1	0	0	0	1
$P_{1,}\ P_{3}$	1	0	1	1	0
$P_{1,}\ P_{4}$	0	1	0	0	1
$P_{2,}\ P_{3}$	1	0	1	1	0
$P_{2,}\ P_{4}$	1	1	0	0	1
$P_{3,}\ P_{4}$	1	1	1	1	0

3.2 Genetic Algorithm

The genetic algorithm used has the following characteristics:

- a static number of generations (100),
- an initial population of the 10% of the total size of the distinction table,
- an one-point crossover operator applied to 10% of the population, and
- a mutation rate of 1% (bit-flip mutation).

Coding. Figure 2 shows the representation of a candidate solution *r*, which is a binary representation, in which the attributes considered within the reduct are represented by 1s; whilst the attributes that are not considered are represented by 0s.

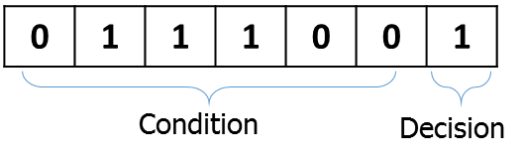

Fig. 2. Representation of a candidate solution.

Fitness Function. The fitness function we used is the expression proposed by Wróblewski [26]:

$$F(r) = \frac{N - L_r}{N} + \frac{C_r}{K}. \quad (6)$$

This fitness function search for a compromise between the number of attributes and the covering of rows of the reduct (classification power) The first term of Eq. (6) focuses on minimizing the number of attributes and the second term tries to maximize the accuracy of the decision (or classification in machine learning). In Eq. (6), N is the total number of attributes in the candidate solution, L_r is the number of 1s in the solution, C_r is the quantity of covered rows according to the candidate solution, and K is the number of combinations of pairs of projects (number of rows).

Genetic operators. The initial population is taken randomly from the distinction table, 10% of the total number of pairs of projects is used. The individuals in the population are ranked —in descending order—according to their score on the fitness function We applied the one-point crossover on the parents, the selection of these individuals is made in two parts:(1) to add

diversity, the first individual is selected at random, and (2) the second parent is selected by strictly following the order of the population ranking, i.e., the first pair of individuals are the first element of the population (the one with the best fitness function at that time) and another one picked at random; for the second pair: the first parent is the second best individual according to the ranking and the other parent is one that is randomly selected, and so on. The resultant offspring is mutated (using bit-flip) with a probability of 1% .

In each generation, all of the fitness-function results —from the offspring— are maintained to compare them with the current generation; if necessary, solutions with a lower score than the offspring are replaced, by removing them and adding the best members of the offspring. At the end of the evolutionary process, the reduct solution is the candidate with the highest score.

3.2 Value Reduction

The simplification of a decision table is regularly done in three steps: the calculation of reducts, the elimination of equivalent rows, and the reduction of rules, which is called value reduction. We want to produce the minimum number of clauses (conditions), so the same decisions can be concluded on a smaller number of conditions. Through this value reduction, we generate a set of relative reducts —not by columns— but for each individual row with respect to decision classes. The method we used for the value reduction is called semantic [27] and is presented in Algorithm 1.

Algorithm 1. Algorithm for value reduction

```
INPUT:      A decision table: IS(U,C,D)
OUTPUT:     A minimal decision algorithm MD(U,C,D)
1. Set F_x = ∅, G_x = ∅, Red_x = ∅
2. For each project x ∈ U do
3.    G_xd=calculate_family(x, d)
4.          for each attribute c ∈ C do
5.                F_xc = F_xc ∪ calculate_family(x, c)
6.          for n=1 to C.size
7.                for all combinations y = C – n_attributes do
8.                      F'_x = F_x
9.                      F'_x = F'_x - F'_xy
10.                     if compare(F'_x, G_xd)
11.                           Red_x = C - y
12. For each project x ∈ U do
13.   MD=MD (Select_Random (Redx))
// Select only one reduct alternative per decision rule
14. Group_Equivalent (MD)
// For simplification, equivalent rules can be grouped
15. return MD
```

Let suppose a decision table already reduced with the calculation of reducts generated by the genetic algorithm, and further reduced with the elimination of equivalent rows.

Table 5. An example of a decision table for project support.

Objects U (Projects)	Attributes			
	Funding	Poverty Impact	Middle Class Impact	Support
1	High	High	Low	No
2	High	Low	High	No
3	High	Low	Low	No
4	Low	High	High	Yes
5	Low	High	Low	No
6	Low	Low	High	No

7	Low	Low	Low	No

In Table 5 the set of condition attributes is {*Funding*, *Poverty Impact*, *Middle Class Impact*} and the decision attribute is {*Support*}. We begin the reduction of redundant values of the set of condition attributes for every decision rule. The problem can be formulated as follows:

Let $F = \{X_1, ..., X_n\}$, the family of attributes of the object *X*, where $X_i \subseteq U$, we need to find all subfamilies *G* of attributes, $G \subseteq F$. We generate the combinations of attributes removing one attribute at a time, such that $\bigcap G \subseteq X_n$ where X_n is the family of attributes of the decision attribute on object *X*. For the sake of illustration, we below compute all subfamilies *G* for the first decision rule (object) of Table 4.

First, we describe *F* for the first decision rule: $F = \{[1]_{Funding}, [1]_{Poverty}, [1]_{Middle\_C}\}$ (e.g., $[1]_{Funding} = \{1,2,3\}$), which is the set of all the objects that have the same attribute value of the pair (1, *Fund*), comparing only the attribute "*Fund*". For all the condition attributes on the first decision rule, the family is $F = \{\{1,2,3\},\{1,4,5\},\{1,3,5\}\}$. The next step is find all the subfamilies G by removing one attribute at a time, such that $\bigcap G \subseteq [1]_{Support} = \{1,2,3,5,6,7\}$, which is the decision attribute. In this case, there are six subfamilies:

G1. $[1]_{Funding} \bigcap [1]_{Poverty} = \{\{1,2,3\} \bigcap \{1,4,5\}\} = \{1\} \subseteq [1]_{Support}$

G2. $[1]_{Funding} \bigcap [1]_{Middle\_C} = \{\{1,2,3\} \bigcap \{1,3,5,7\}\} = \{1,3\} \subseteq [1]_{Support}$

G3. $[1]_{Poverty} \bigcap [1]_{Middle\_C} = \{\{1,4,5\} \bigcap \{1,3,5,7\}\} = \{1,5\} \subseteq [1]_{Support}$

G4. $[1]_{Funding} = \{1,2,3\} \subseteq [1]_{Support}$

G5. $[1]_{Poverty} = \{1,4,5\} \not\subset [1]_{Support}$

G6. $[1]_{Middle\_C} = \{1,3,5,7\} \subseteq [1]_{Support}$

Once all the subfamilies of the first decision rule have been evaluated, five subsets are considered reducts; in this case, the reducts with less attributes are taking into consideration, because these subsets are contained into the reducts with more attributes (we are searching for the minimum number of rules). In a similar way, the value reduction for the remaining decision rules of the table can be performed. Table 6 is the result of the calculation on all the decision rules. It is important to note that the minimal decision rules (1 and 1') were determined with the value reduction of G4 and G5, respectively.

Table 6. The reduced decision rules from using Table 5.

	Objects *U* (Projects)	Attributes			
		Fund	Poverty Impact	Middle Class Impact	Support
Decision Rules	1	High	-	-	No
	1'	-	-	Low	No
	2	High	-	-	No
	2'	-	Low	-	No
	3	High	-	-	No
	3'	-	Low	-	No
	3"	-	-	Low	No
	4	-	High	High	Yes
	5	-	-	Low	No
	6	-	Low	-	No
	7	-	Low	-	No
	7'	-	-	Low	No

We can see from Table 6, that decision rules 4, 5, 6 only have one reduct, decision rules 1, 2 have two reducts and decision rule 3 has three reducts. We must choose only one reduct per rule; but, in this case, there are $2\times2\times3\times1\times1\times1\times2=24$ alternative combinations. In Table 7, to further explain the value reduction process, we show one alternative solution for this example.

Table 7. An example of a decision algorithm chosen from Table 6.

Objects *U* (Projects)	Attributes			
	Fund	Poverty Impact	Middle Class Impact	Support
1	-	-	Low	No
2	-	Low	-	No
3	-	Low	-	No
4	-	High	High	Yes
5	-	-	Low	No
6	-	Low	-	No
7	-	-	Low	No

Table 7 can be reduced by grouping equivalent rules to generate a minimal *decision algorithm*, which is a set of decision rules. Decision rules 1, 5, 7 are equivalent as well as decision rules 2, 3 and 6. Therefore, we grouped equivalent rules, and the resultant groups are displayed in Table 8.

Table 8. The reduced decision algorithm from Table 7.

Objects *U* (Projects)	Attributes			
	Fund	Poverty Impact	Middle Class Impact	Support
1,5,7	-	-	Low	No
4	-	High	High	Yes
2,3,6	-	Low	-	No

With this reduction, in Table 8 we have only three rules, with seven clauses (attributes) in total. Without such a reduction, we would have seven rules with four clauses each one (see Table 5). We have used this method in conjunction with the genetic algorithm to simplify a decision table. The genetic algorithm generates the reduced decision table, whichs is the input for the exact method described in this section. The resulting minimal decision rules for the data example of Table 5 (taken from Table 8) are:

1) IF (*Middle Class Impact* = Low) THEN *Support* = No,
2) IF (*Poverty Impact* = High and *Middle Class Impact* = High) THEN *Support* = Yes,
3) IF (*Poverty Impact* = No) THEN *Support* = No.

4. Experimental Design

The methodology and the working environment used for experimentation and evaluation of our proposal are described in this section.

The characteristics of the computer equipment are the following:

- Operating system: Windows 8.
- Processor: Core i7-2600 to 3.06 GHz
- RAM: 8GB Ram,
- Programming environment: Netbeans 8.0.2.

We have divided the experimentation in two phases:

1. **Machine Learning Instances.** There are many repositories of machine learning instances available for the scientific community for experimentation. We have used these instances to evidence the effectiveness of the methodology in terms of accuracy when removing attributes and reducing clauses for the value reduction. We have also used instances that are well known in the machine learning community for attribute selection.
2. **Case of study: Project Portfolio Instance.** To our knowledge, there is no repository of standard instances for project portfolio experimentation. Once we have proved the methodology works well with the machine learning instances, we have the confidence that our proposal can be applied on an instance of project portfolio for generating the justification of the recommendation.

4.1 Experiment 1: Machine Learning Instances

The instances used for this experimentation were taken from the UCI Repository of Machine Learning Databases [28]. This repository is widely accepted in machine learning experiments by the scientific community and some instances from UCI are used on related works [29, 30, 31] on attribute reduction. For this reason, we have measured the classification accuracy of our method via these instances.

Table 9 shows three pairs of columns with the classification percentage (accuracy) and the number of attributes (size). The first pair considers the original instances (as found in the UCI repository). In the second pair the attributes that are not in the reducts (that were generated by the genetic algorithm) were removed. Finally, the third pair is similar to the second column, but the framework RSES (described in Section 2.3) is used here.

Table 9. Comparison of RSES and our genetic algorithm for reduct calculation.

Instance	Attributes					
	Originals		Reduced-Gen		Reduced-RSES	
	Accuracy	Size	Accuracy	Size	Accuracy	Size
Balance-scale.txt	76.64	4	76.64	4	76.64	4
Balloons (a+s).txt	100	4	100	2	100	2
Balloons (y-s+a-s).txt	62.5	4	56.25	3	62.5	4
Breastcancer.txt	75.5245	9	66.7832	4	75.5245	8
Car.txt	92.3611	6	93.2275	5	92.3611	6
Diabetes.txt	73.8821	8	64.7135	1	63.8021	3
Hayes-roth.txt	80.303	4	80.303	3	80.303	3
Heart.txt	76.6667	13	70.7407	2	72.5926	3
Iris.txt	96	4	94	1	96	3
Lenses.txt	79.1667	4	79.1667	3	79.1667	4
Lymphography.txt	77.027	18	77.7027	4	75.6757	6
Monks1.txt	75	6	75	3	75	3
Monks2.txt	90.0463	6	90.0463	6	90.0463	6
Monks3.txt	100	6	100	3	100	3
Satellite_tst.txt	83.65	36	76.2	3	82.4	6
Shuttle-landing.txt	53.3333	6	53.3333	2	53.3333	4
Soybean-small.txt	97.8723	35	95.7447	3	100	2
Spect.txt	82.7715	22	83.5206	10	83.5206	17
Tic-tac-toe.txt	84.5511	9	81.2109	6	83.5073	8
Vote.txt	96.3218	16	96.092	5	96.3218	9

The classification accuracy was obtained by using the Weka software in its Weka Explorer option, which has an option for the preprocessing of data; hence, attributes can be removed from instances to perform the classification of the objects. All the attributes that the genetic algorithm considered removable were selected and deleted in the preprocessing of Weka (see Fig. 3). In the same way, the preprocessing of the RSES attributes —that were found disposable— was made.

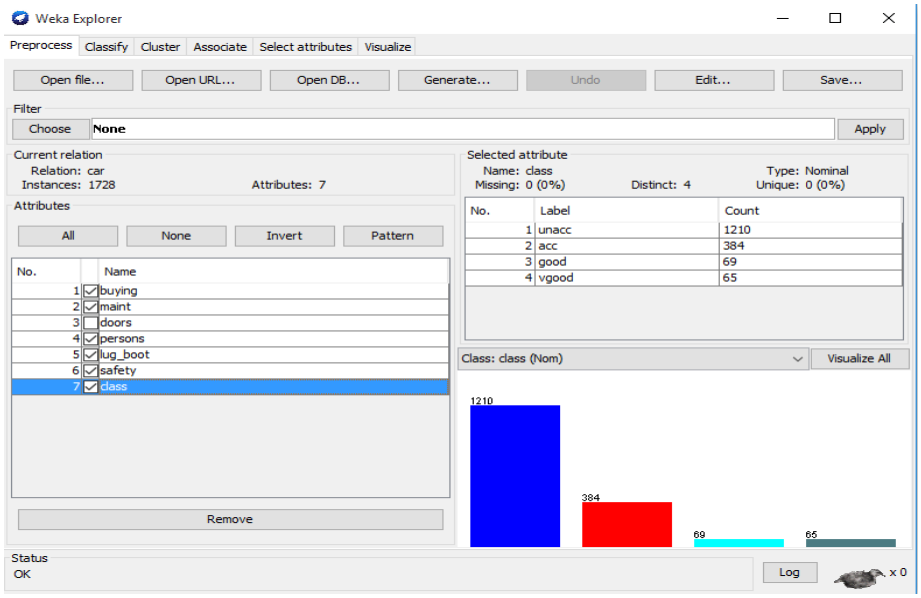

Fig. 3. Start screen of the Weka Explorer option within the Weka software; the Car instance from the UCI repository is loaded.

Table 10 shows a comparison of the average number of clauses and rules using the decision tables generated by the genetic algorithm with the value reduction method. The first two columns have the number of rules and clauses in the decision rules that were reduced only by columns with the genetic algorithm (these columns are labeled "Reducts"). The third and fourth columns shows the average number of clauses and rules of the decision tables that have been reduced by the value reduction method (we called it "Reducts + Value Reduction"). Thirty runs were performed per instance.

Table 10. Number of decision rules before and after the value reduction.

Instance	Decision Rules			
	Reducts		Reducts+Value Reduction	
	Rules	Clauses	Rules	Clause
Balance-scale.txt	625	3125	303	1335
Balloons (a+s).txt	20	60	3	7
Balloons (y-s+a-s).txt	16	64	3	9
Breastcancer.txt	286	1430	117	449
Car.txt	1728	10368	79	440
Diabetes.txt	768	1536	427	854
Hayes-roth.txt	132	528	21	78
Heart.txt	270	810	207	489
Iris.txt	150	300	38	79
Lenses.txt	24	96	4	14
Lymphography.txt	148	740	58	504
Monks1.txt	432	1728	22	83
Monks2.txt	432	3024	34	184
Monks3.txt	432	1728	12	35
Satellite_tst.txt	2000	8000	1172	3740
Shuttle-landing.txt	15	45	5	12
Soybean-small.txt	47	188	6	14
Spect.txt	267	2937	95	445
Tic-tac-toe.txt	958	7	173	924

Vote.txt	435	22185	34	125

The experiments were validated statistically with the Wilcoxon test using a significance level of 0.05. According to the experimental results reported in Table 9, there is no significant difference on the classification accuracy. However, with regard to the number of attributes there is a significant difference, the experimental results in Table 10 demonstrate that the reduction on rules and clauses are significantly different.

4.2 Project Portfolio Instance

The decision rules are used to outline information to the DM on how the recommended portfolio was constructed. These rules are not used to create knowledge but to describe it. We have used an instance with 100 projects and five atrributes of the public portfolio problem; the decision attribute is generated using the multicriteria algorithm described by Fernandez [25].

Instead of having the 100 rules (one for each project), by applying our reduction methodology the knowledge on how the construction of one portfolio is made can be explained by 48 rules only. In addition, these 48 rules are reduced in the number of clauses (attributes) having an average of 236 rules. Some of such rules are presented in Table 11. The table has five condition attributes: *Cost* (funding cost of the project), *PD* (people in poverty directly benefited), *PI* (people in poverty indirectly benefited), *LCD* (people of the low class directly benefited), *LCI* (people of the low class indirectly benefited). The value of the *Decision* attribute of this table is generated by doing an analysis of the resulting rules and inducing a conclusion.

Why certain projects are rejected or accepted is explained with these attributes. In Table 11, "Not significant" means that those attributes were removed during the attribute reduction. The code and all files of the case studies are available for reproduction.

Table 11. Example of minimal decision rules for 100 projects and 5 attributes.

Condition Attributes					Decision
Cost	PD	PI	LCD	LCI	
Not significant	Low Impact	Average Impact	Not significant	Not significant	All projects with these attributes are not supported.
High	Average Impact	Not significant	Not significant	Not significant	All projects with these attributes are not supported.
Average	High Impact	Low Impact	Low Impact	Low Impact	Half of the projects with these attributes are supported.
Average	Very High Impact	High Impact	Not significant	Not significant	Support all projects with this attributes
High	Very High Impact	Very High Impact	Not significant	Not significant	Support all projects with these attributes.

With this information we can explain how the construction of the portfolio is done. Then, we can summarize the decision policy for the supported projects and verify if these conclusions match the DM's preferences. With this knowledge the DM can infer if the recommended project portfolio (which was chosen from the set of good portfolios) satisfies his/her beliefs. By the proposed hybrid methodology, a significant reduction of the clauses of the decision rules is reached without any significant impact on classification accuracy.

We believe that the information of the decision rules would be a great asset in the dialogue framework that we proposed in [32], particularly during a direct interaction with a DM to: (1) satisfy his/her enquiries on how the portfolio was chosen, (2) explain why a particular project is supported or not, and (3) clarify his/her preferences.

5. Conclusions and Future Work

In this paper, a methodology is proposed to help the DM with the problem of choosing a recommendation among a set of alternatives. The experimental evidence shows that the decision table is effectively reduced through a hybrid rough set-based method, which makes easier to understand the summary of a public project portfolio to support.

Under the conditions of uncertainty and vagueness, it is required a model of effective justification that fits the nature of the problem and the users. With the reduced decision table —by columns and rows— we can give a valid explanation. Departing from the results, we have concluded that the hybrid combination of a heuristic genetic algorithm followed by and exact method give satisfactory results (in terms of the number of total decision rules and clauses). Moreover, an acceptable accuracy is kept, which was statistically verified.

As future work, we are going to:(a) modify the genetic algorithm (by setting other types of genetic operators), (b) implement a multi-objective function, (c) make changes in the exact method to create an approximation algorithm (with the objective of reducing the run time), and (d) improve the random selection of one recommendation among a set of good solutions (i.e. the DM can point some projects that (s)he wants to support, and the framework have to discard the solutions that do not have those projects).

Acknowledgments

This work has been partial supported by CONACYT Project.254498 and PRODEP.

References

1. Ouerdane, W., Dimopoulos, Y., Liapis, K., & Moraitis, P.: Towards automating decision aiding through argumentation. Journal of Multi-Criteria Decision (2011)
2. Labreuche, C., Maudet, N., & Ouerdane, W.: Minimal and complete explanations for critical multi-attribute decisions. In Algorithmic Decision Theory (pp. 121-134) Springer Berlin Heidelberg (2011)
3. Seethalakshmi, P., & Vengataasalam, S.: Application of Rough Set Approach in Dengue Diagnosis. Applied Mathematical Sciences, 8(127), 6313-6324 (2014)
4. Seethalakshmi, P., & Vengataasalam, S.: Attribute Selection in Product Mix: A Rough Set Approach, The International Journal of Science and Technoledge, 71-77 (2014)
5. Chen, Y., Miao, D., & Wang, R.: A rough set approach to feature selection based on ant colony optimization. Pattern Recognition Letters, 31(3), 226-233 (2010)
6. Cruz-Reyes, L., Trejo, C. M., Irrarragorri, F. L., & Santillán, C. G. G.: A Decision Support System Framework for Public Project Portfolio Selection with Argumentation Theory. In Recent Advances on Hybrid Approaches for Designing Intelligent Systems (pp. 467-479). Springer International Publishing (2014)
7. Litvinchev, I. S., López, F., Alvarez, A., & Fernández, E.: Large-scale public R&D portfolio selection by maximizing a biobjective impact measure. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, 40(3), 572-582 (2010)
8. Carazo, A. F., Gómez, T., Molina, J., Hernández-Díaz, A. G., Guerrero, F. M., & Caballero, R.: Solving a comprehensive model for multiobjective project portfolio selection. Computers & operations research, 37(4), 630-639 (2010)
9. Cruz-Reyes, L., Fernandez, E., Gomez, C., Sanchez, P., Castilla, G., & Martinez, D.: Verifying the Effectiveness of an Evolutionary Approach in Solving Many-Objective Optimization Problems. In Design of Intelligent Systems Based on Fuzzy Logic, Neural Networks and Nature-Inspired Optimization (pp. 455-464). Springer International Publishing (2015)
10. Fernández-González, E., Vega-Lopez, I., & Navarro-Castillo, J.: Public portfolio selection combining genetic algorithms and mathematical decision analysis. Bio-Inspired Computational Algorithms and Their Applications, 139-160 (2012)
11. Fernandez, E., & Navarro, J.: A genetic search for exploiting a fuzzy preference model of portfolio problems with public projects. Annals of Operations Research, 117(1-4), 191-213 (2002)
12. B. Roy.: Multicriteria methodology for decision aiding, Nonconvex Optimization and Its Applications, Springer, (1996) doi:10.1007/978-1-4757-2500-1.
13. Beasley, David, R. R. Martin, and D. R. Bull.: An overview of genetic algorithms: Part 1. Fundamentals. University computing 15 58-58 (1993)
14. Pawlak, Z.: Rough sets present state and further prospects. Intelligent Automation & Soft Computing, 2(2), 95-101 (1996)
15. Andrzej Skowron and Jerzy Grzymal: From rough set theory to evidence theory. In Advances in the Dempster-Shafer theory of evidence, Ronald R. Yager, Janusz Kacprzyk, and Mario Fedrizzi (Eds.). John Wiley \& Sons, Inc., New York, NY, USA 193-236 (1994)
16. Pawlak, Z., Grzymala-Busse, J., Slowinski, R., & Ziarko, W.: Rough sets. Communications of the ACM, 38(11), 88-95 (1995)
17. Düntsch, I., & Gediga, G.: Rough set data analysis. Encyclopedia of Computer Science and Technology, 43(28), 281-301 (2000)
18. Pawlak, Z., & Skowron, A.: Rudiments of rough sets. Information sciences, 177(1), 3-27 (2007)
19. Pawlak, Zdzisław: Rough sets and intelligent data analysis. Information sciences 147.1 1-12 (2002)
20. Pawlak, Z., & Skowron, A.: Rough sets and Boolean reasoning. Information sciences, 177(1), 41-73 (2007)

21. Skowron, A., & Rauszer, C.: The discernibility matrices and functions in information systems. In Intelligent Decision Support pp. 331-362, Springer Netherlands (1992)
22. Stefanowski, J.: Changing representation of learning examples while inducing classifiers based on decision rules. Artificial Intelligence Methods, AI-METH (2003)
23. Riza, L. S., Janusz, A., Bergmeir, C., Cornelis, C., Herrera, F., Śle, D., & Benítez, J. M.: Implementing algorithms of rough set theory and fuzzy rough set theory in the R package "roughsets". Information Sciences, 287, 68-89 (2014)
24. Bazan, Jan G., and Marcin Szczuka: RSES and RSESlib-a collection of tools for rough set computations. Rough Sets and Current Trends in Computing. Springer Berlin Heidelberg (2001)
25. Fernandez, E., Gomez, C., Rivera, G., & Cruz-Reyes, L.: Hybrid metaheuristic approach for handling many objectives and decisions on partial support in project portfolio optimisation. Information Sciences, 315, 102-122 (2015)
26. Wroblewski, J.: Finding minimal reducts using genetic algorithms. In Proccedings of the second annual join conference on infromation science pp. 186-189 (1995)
27. Pawlak, Zdzisław. Rough sets: Theoretical aspects of reasoning about data. Vol. 9. Springer Science & Business Media (1991)
28. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI repository of machine learning databases. University of California, Irvine, Department of information and computer sciences, http://www.ics.uci.edu/~mlearn/MLRepository.html (1998)
29. Moshkov, M., Piliszczuk, M., & Zielosko, B.: Partial Covers, Reducts and Decision Rules in Rough Sets. Studies in Computational Intelligence, 145 (2010)
30. Wang, X., Yang, J., Teng, X., Xia, W., & Jensen, R.: Feature selection based on rough sets and particle swarm optimization. Pattern Recognition Letters, 28(4), 459-471 (2007)
31. Caballero, Y., Bello, R., Alvarez, D., & Garcia, M.: Two new feature selection algorithms with Rough Sets Theory. Artificial Intelligence in Theory and Practice, 209-216 (2006)
32. Cruz-Reyes, L., Trejo, C. M., Irrarragorri, F. L., & Santillan, C. G. G.: Simplification of Decision Rules for Recommendation of Projects in a Public Project Portfolio. In Design of Intelligent Systems Based on Fuzzy Logic, Neural Networks and Nature-Inspired Optimization pp. 419-429, Springer International Publishing (2015)