BIRZIET UNIVERSITY- FACULTY OF INFORMATION TECHNOLOGY
MASTER OF COMPUTING PROGRAM

Feature Selection Using Tabu Search Algorithm

Prepared by
Muayyad Sharif
1095296

Seminar Report Submitted To The Computing Program

Supervised by
Dr. Wasel Ghanem & Dr. Majdi Mafarja

• ABSTRACT

In data preprocessing , knowledge discovery and data mining , Feature Selection plays a big role. To find the minimal subset of high dimensional data and make developments in terms of time complexity and accuracy, we remove irrelevant and misleading features. Computer intelligence is employed to solve the problem of feature selection.

In this paper, I introduce a modified memory-based heuristic of tabu search to solve the attribute reduction problem in rough set theory. The proposed method, called modified tabu search attribute reduction (mTSAR), is a high-level TS with long-term memory. Therefore, mTSAR invokes diversification and intensification search schemes besides the TS neighborhood search methodology applied on well-known data sets compared with other heuristic solutions. mTSAR shows promising and competitive performance compared with some other CI tools in terms of solution qualities. Moreover, mTSAR shows a superior performance in saving the computational costs.

The modification which I proposed concerned with starting with random big subsets of attributes ,many iterations for each subset ,computing the dependency degree (dd) for each solution. If a solution got dd==1 then the subset is decreased by one ,until no improvement got and the cardinality of each solution equals one.
The above is faster than the traditional TS and gets solutions better or equal saving time and complexity.

CONTENTS

• Introduction

Computational Intelligence (CI) tools and applications have grown rapidly since its inception in the early 1990s of the last century . CI tools were firstly limited to fuzzy logic, neural networks and evolutionary computing as well as their hybrid methods.

Nowadays, the definition of CI tools has been extended to cover many of other machine learning tools [2]. One of the main CI classes is Granular Computing (GrC) [3], which has recently been developed to cover all tools that mainly invoke computing with fuzzy and rough sets.

Calculation of reducts of an information system is a key problem in RS theory [1]. We need to get reducts of an information system in order to extract rule-like knowledge from an information system. Reduct is a minimal attribute subset of the original data which has the same discernibility power as all of the attributes in the rough set framework.

Obviously, reduction is an attribute subset selection process, where the selected attribute subset not only retains the representational power, but also has minimal redundancy.

Actually, the problem of attribute reduction (AR) of an information system has made great gain from rapid development of CI tools [4].

One class of the promising CI tools is memory-based heuristics, like Tabu Search (TS), which have shown their successful performance in solving many combinatorial search problems [5].

In this paper, I propose a modifiedTS-based method, called tabu search for attribute reduction (mTSAR), to solve the problem of attribute reduction of an information system. mTSAR uses a 0-1 variable representation of solutions in searching for reducts. A rough set dependency degree function is invoked to measure the solution qualities.

The search process in mTSAR is a high-level TS with long term memory. Therefore, mTSAR invokes diversification and intensification search schemes besides the TS neighborhood search methodology.

In the literature, much effort has been made to deal with the AR problem.

The mTSAR method proposed in this paper uses the TS neighborhood search methodology for searching reducts of an information system.

TS neighborhood search is based on two main concepts; avoiding return to a recently visited solution, and accepting downhill moves to escape from local maximum information. Some search history information is reserved to help the search process to behave more intelligently. Specifically, the best reducts found so far are saved to provide the diversification and intensification schemes with more promising solutions.

The modification which I worked on depends on generating random big subsets of attributes near two thirds of |C| ,many iterations for each subset are executed ,computing the dependency degree (dd) for each solution. If a solution got dd==1 then the subset cardinality is decreased by one ,until no improvement is got and the cardinality of each solution equals one.

The above modification is faster than the traditional TS and gets solutions better or equal ,saving time and computing complexity.

The paper is organized as follows:

In the next section, we briefly give the principles of rough set and attribute reduction ,and tabu search as preliminaries needed throughout the paper.

In Sect. 3, we highlight the main components of mTSAR and present the algorithm.

In Sect. 4, we report numerical results with mTSAR using some well-known datasets. Finally, the conclusion makes up Sect. 5.

• Feature Selection

Feature Selection (FS) is one of the data reduction techniques that can be defined as the process of choosing a subset of the original features according to certain criteria .
Consequently, irrelevant, redundant, and noisy data is reduced to the minimum, and then effects appear immediately for applications: speeding up data mining algorithm, and improving predictive accuracy.

There are 3 strategies for searching: complete: in which we take all possible subsets and find the optimal ones, heuristic: a sort of depth-first search guided by heuristics and nondeterministic: random search-current set does not get big or small like previous.

There are four phases for FS as shown in figure:

(a) Generation phase, to generate the candidate subsets (solution). There are three main schemes for generating subsets (sequential forward, sequential backward , bidirectional and random). The generation can start with no feature or with all features or with random subset of features.
(b) Evaluation function measures the goodness of a subset produced by some generation procedure and this value is compared with the previous best, if it’s found to be better then it’s replaced with previous best subset.
(c) stopping criterion: without stopping criterion feature selection process may run exhaustively or forever through the space of subsets. Stopping criteria can be: predefined number of iterations, predefined number of features or when the optimal subset is obtained according to some evaluation function.
(d) validation procedure : to check whether the optimal subset (solution) produced is valid or not, by testing the results of this subset and comparing it with previously established results .

• Significance of Attributes

Attributes cannot be equally important, and some of them can be eliminated from an information table without losing information contained in the table. The idea of attribute reduction can be generalized by introducing a concept of *significance of attributes*, which enables us evaluation of attributes not only by two-valued scale, *dispensable* or *indispensable*, but by assigning to an attribute a real number from the closed interval [0,1], expressing how important is an attribute in an information table.
Significance of an attribute can be evaluated by measuring effect of removing the attribute from an information table on classification defined by the table.

Table 1 An example of reducts

U	(i) A dataset				(ii) A reduced dataset			(iii) A reduced dataset		
	a	*b*	*c*	*d*	*a*	*c*	*d*	*b*	*c*	*d*
*e*1	0	0	0	1	0	0	1	0	0	1
*e*2	0	0	1	0	0	1	0	0	1	0
*e*3	0	0	2	0	0	2	0	0	2	0
*e*4	1	0	0	1	1	0	1	0	0	1
*e*5	1	1	1	1	1	1	1	1	1	1
*e*6	1	0	2	0	1	2	0	0	2	0

$$\gamma_P(Q) = \frac{|POS_P(Q)|}{|U|},$$

We can define dependency degree as follows[16]**:**

If we calculate the dd for the subsets in the table above we get:

$$\gamma_{\mathbb{C}}(\mathbb{D}) = 1,$$

$$\gamma_{\{a,b\}}(\mathbb{D}) = \frac{1}{6}, \gamma_{\{b,c\}}(\mathbb{D}) = 1, \gamma_{\{a,c\}}(\mathbb{D}) = 1,$$

$$\gamma_{\{a\}}(\mathbb{D}) = 0, \gamma_{\{b\}}(\mathbb{D}) = \frac{1}{6}, \gamma_{\{c\}}(\mathbb{D}) = \frac{2}{3}.$$

$$\mathfrak{R}_{\min} = \{\{a, c\}, \{b, c\}\}.$$

Comparing two solution x and x' ,we say that x is better than x' if one of the following conditions holds[16]:

- $\gamma_x(\mathbb{D}) > \gamma_{x'}(\mathbb{D})$,
- $\sum_i x_i < \sum_i x'_i$ if $\gamma_x(\mathbb{D}) = \gamma_{x'}(\mathbb{D})$.

• Metaheuristics

Metaheuristics iteratively apply the generation and replacement procedures from the current single solution. In the generation phase, a set of candidate solutions are generated from the current solution s. This set C(s) is generally obtained by local transformations of the solution. In the replacement phase, a selection is performed from the candidate solution set C(s) to replace the current solution; that is, a solution s′∈ C(s) is selected to be the new solution. This process iterates until a given stopping criteria. The generation and the replacement phases may be memory less. In this case, the two procedures are based only on the current solution. Otherwise, some history of the search stored in a memory can be used in the generation of the candidate list of solutions and the selection of the new solution. Popular examples of such S-metaheuristics are local search, simulated annealing, and tabu search [15].

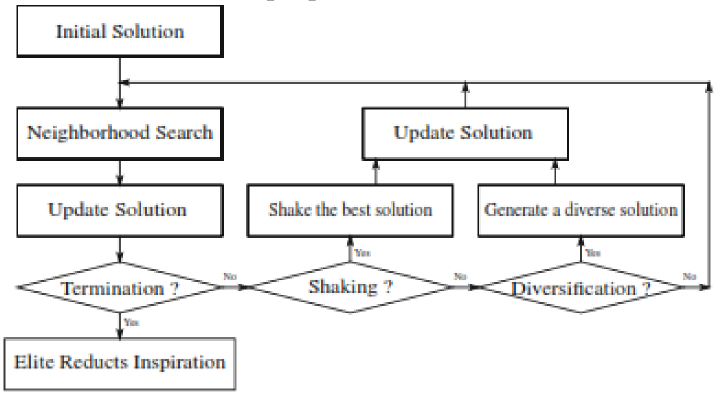
• Tabu search[16]:

It is a heuristic method originally proposed by Glover in 1986. TS has primarily been proposed and developed for combinatorial optimization problems [11], and has shown its capability of dealing with various difficult problems [12], Moreover, there have been some attempts to develop TS for continuous optimization problems [13].

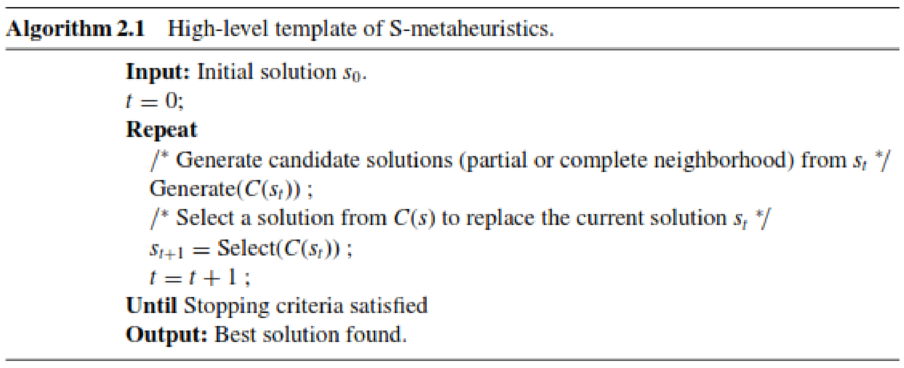
The main feature of TS is its use of an adaptive memory and responsive exploration. A simple TS combines a local search procedure with anti-cycling memory-based rules to prevent the search from getting trapped in local optimal solutions. Specifically, TS avoids returning to recently visited solutions by constructing a list of them called Tabu List (TL). In each iteration of the simple TS illustrated in Algorithm 2.1 below, TS generates many trial solutions in a neighborhood of the current solution. The trial solutions generation process is composed to avoid generating any trial solution that has already been visited recently. The best trial solution in the generated solutions will become the next solution. Therefore, TS can accept downhill movements to avoid getting trapped in local maxima. TS can be terminated if the number of iterations without any improvement exceeds a predetermined maximum iteration number.

Tabu search for attribute reduction (TSAR): This solution representation uses a binary representation for solutions (attribute subsets). Therefore, a trial solution x is a 0–1 vector with dimension equal to the number of conditional attributes |C| .If a component x of $x_i = 1,...,|C|$, has the value 1, then the *ith* attribute is contained in the attribute subset represented by the trial solution x. Otherwise, the solution x does not contain the *ith* attribute as shown in the diagram below.

Figure 1 shows traditional TSAR flow-chart[16]



Algorithm-1 : illustrates the high-level template of S-metaheuristics[16].

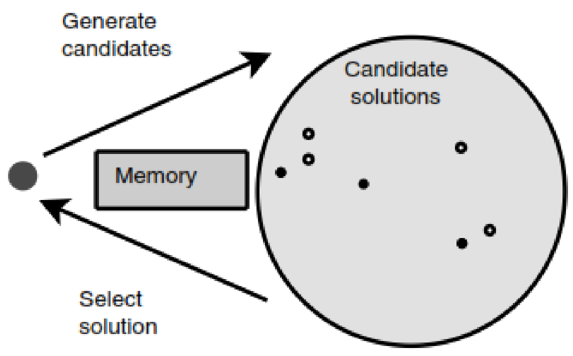

Algorithm 2.1 High-level template of S-metaheuristics.

```
Input: Initial solution $s_0$.
$t = 0$;
Repeat
    /* Generate candidate solutions (partial or complete neighborhood) from $s_t$ */
    Generate($C(s_t)$) ;
    /* Select a solution from $C(s)$ to replace the current solution $s_t$ */
    $s_{t+1}$ = Select($C(s_t)$) ;
    $t = t + 1$ ;
Until Stopping criteria satisfied
Output: Best solution found.
```

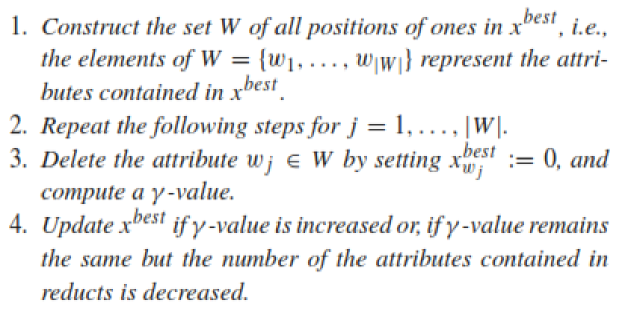
Unlike exact methods, metaheuristics allow to tackle large-size problem instances by delivering satisfactory solutions in a reasonable time[15]. There is no guarantee to find global optimal solutions or even bounded solutions. Metaheuristics have received more and more popularity in the past 20 years. Their use in many applications shows their efficiency and effectiveness to solve large and complex problems.

- **Initial Solution**[15]

Two main strategies are used to generate the initial solution: a random and a greedy approach. There is always a trade-off between the use of random and greedy initial solutions in terms of the quality of solutions and the computational time. The best answer to this trade-off will depend mainly on the efficiency and effectiveness of the random and greedy algorithms at hand, and the S-metaheuristic properties. For instance, the larger is the neighborhood, the less is the sensitivity of the initial solution to the performance of the S-metaheuristics [15].
Generating a random initial solution is a quick operation, but the metaheuristic may take much larger number of iterations to converge. To speed up the search, a greedy heuristic may be used. Indeed, in most of the cases, greedy algorithms have a reduced polynomial-time complexity. Using greedy heuristics often leads to better quality local optima. Hence, the S-metaheuristic will require, in general, less iterations to converge toward a local optimum. Some approximation greedy algorithms may also be used to obtain a bound guarantee for the final solution. However, it does not mean that using better solutions as initial solutions will always lead to better local optima.



- **Shaking**[16]



1. *Construct the set W of all positions of ones in x^{best}, i.e., the elements of $W = \{w_1, \ldots, w_{|W|}\}$ represent the attributes contained in x^{best}.*
2. *Repeat the following steps for $j = 1, \ldots, |W|$.*
3. *Delete the attribute $w_j \in W$ by setting $x^{best}_{w_j} := 0$, and compute a γ-value.*
4. *Update x^{best} if γ-value is increased or, if γ-value remains the same but the number of the attributes contained in reducts is decreased.*

- **Proposed Modification**

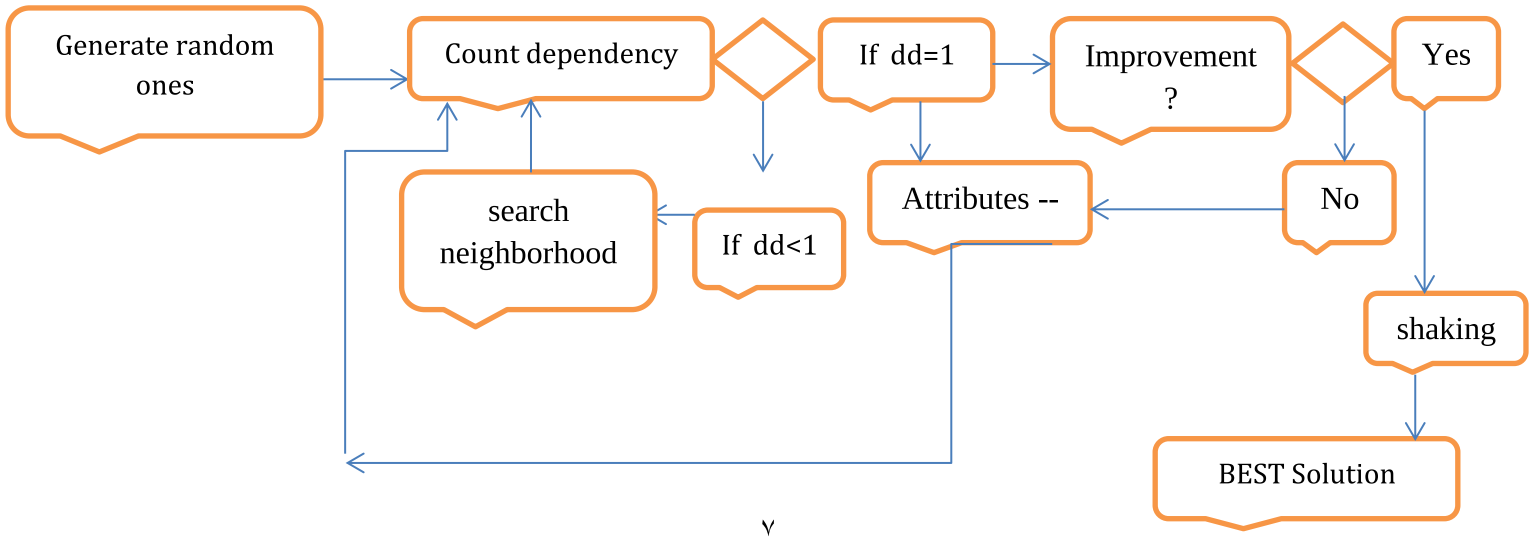
My modification on the TS which I called *mTSAR* can be viewed in the following procedures:

1. Generate random solution of ones round 2/3 of |C|.
2. Count dependency degree.

$$\gamma_P(Q) = \frac{|POS_P(Q)|}{|U|},$$

3. If dd<>1 then search neighbor of the solution by changing random ones to zeros.
4. If dd=1 then decrease the number of ones by one, calculate dd.
5. Continue decreasing until no improvement occurs.
6. The best solution got with dd=1 which has the least attributes is imposed now to shaking which implies changing all the ones with zeros in order.
7. If a new solution got dd=1 ,it is the new best solution.

Figure 2 shows the modification flow-chart



Improvements on TS

I can summarize the improvements made on TS as follows:

1. Diversification was replaced by generating random solutions.
2. Intensification was replaced by decreasing the attributes by one in each iteration.
3. No need to many steps and concepts in regular TSAR like core attributes and predefined iterations such as I_{div} ,I_{shake} ,I_{imp} ,I_{max}.
4. Using excel sheets in one file called (tabu.xls) views a better picture compared to the traditional text way of storing data.

Methods used

1. Main method: an object created represents the excel input file ,runs other methods.
2. Sum method: takes any matrix as an argument ,returns its sum or the number of ones (attributes).
3. Count method: calculates the dependency degree for each matrix containing ones representing any solution as follows:

The method compares each record with the others ,if it is similar to any in conditional attributes and differs in class ,the record is discarded and if they belong to the same class the counter is increased by one. The attributes considered are those who are represented by one in the solution matrix so this method costs O(n^2) where n is the number of records.

Comparison in results

I compare the above modification with other three algorithm in addition to regular TS and got the following results:

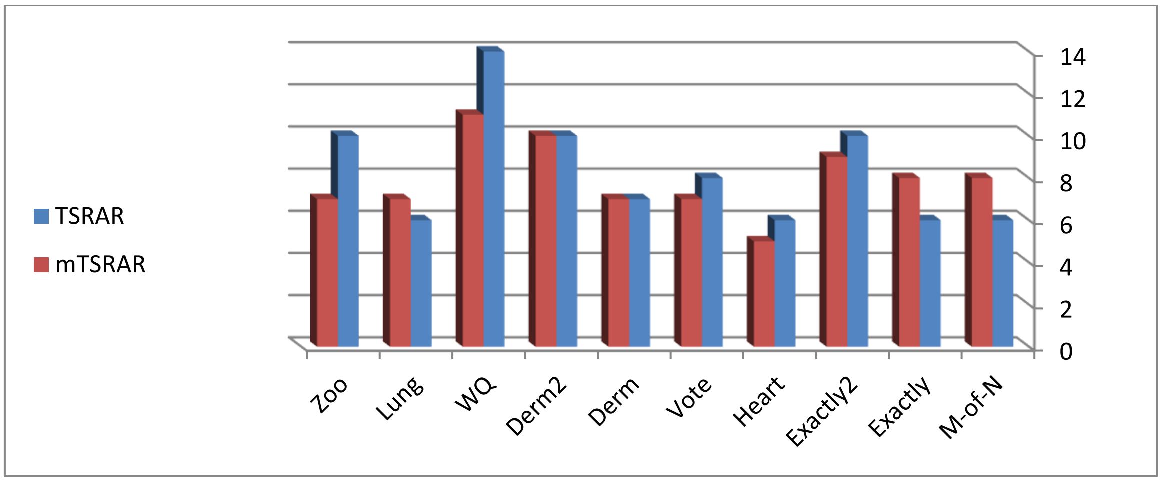
Data set	Number of attributes	AntRSAR	SimRSAR	GenRSAR	TSRAR	**mTSRAR**
M-of-N	13	6	6	6	6	**8**
Exactly	13	6	6	6	6	**8**
Exactly2	13	10	10	10	10	**9**
Heart	13	6	6	6	6	**5**
Vote	16	8	8	8	8	**7**
Derm	34	6	6	10	7	**7**
Derm2	34	8	8	10	10	**10**
WQ	38	12	13	16	14	**11**
Lung	56	4	4	6	6	**7**
Zoo	16				10	**7**

1. mTSRAR produced the best results in general ,except in 2 data sets.
2. mTSRAR required less iterations in common.
3. The following figure shows the comparison.

Comparison in procedures

In terms of	TSRAR	mTSRAR
Diversification	Count the times in each the attribute is used in a solution	Randomize ones
Intensification	Using core attributes concept and best reducts	Reduce attributes one by one in each iteration
Iterations	More iterations: I_{div} ,I_{shake} ,I_{imp} ,I_{max}	Less iterations
Data format	Text data file	Excel sheet

Figure 3 : The effect of modification



- ## Conclusion

We noticed from the previous study that features selection is a sensitive procedure required to take a decision from data, using RST theory and wrapper model helped us in that procedure.
The attribute reduction problem in rough set theory has been studied in this paper. A modified TS-based modification, called mTSAR, has been proposed to solve the problem. New diversification and intensification improvements have been inlaid in mTSAR to achieve better performance and to fit the problem. Comparisons with other CI tools have revealed that mTSAR is promising and it is less expensive in computing the dependency degree function γ . Generally, we may conclude that modified Tabu Search could be successfully applied to the rough set attribute reduction problem and obtain reasonable results that are comparable to other methods in terms of solution qualities. However, the proposed method has shown superior performance in saving the computational cost and time. Manipulating in diversification and intensification procedures by the decreasing randomizing got best and faster results than the regular procedures in TSAR.

- **REFERENCES:**

[1] Z. Pawlak: Rough sets, International Journal of Computer and Information Sciences, 11,

[2] Burke EK, Kendall G (2005) Search methodologies: introductory tutorials in optimization and decision support techniques. Springer,Berlin

[3] Bargiela A, Pedrycz W (2002) Granular computing: an introduction.Springer, Berlin

[4] Jensen R, Shen Q (2003) Finding rough set reducts with ant colony optimization. In: Proceedings of the 2003 UK workshop on computational intelligence, pp 15–22

[5] Glover F, Laguna M (1997) Tabu search. Kluwer, Boston

[6] L. Zadeh: Fuzzy sets, Information and Control, 8, 338-353, 19651982

[7] Han, Jiawei, and Micheline Kamber. *Data mining: concepts and techniques – Third edition-*. Morgan Kaufmann, 2006.

[8] Liu, Huan. *Feature Selection for Knowledge Discovery and Data Mining*. Norwell, Massachusetts 02061 USA: Kluwer Academic Publishers Group, 1998.

[9] B. Russell: The Principles of Mathematics, London, George Allen & Unwin Ltd., 1st Ed. 1903 (2nd Edition in 1937)

[10] Dash, Manoranjan, and Huan Liu. "Feature selection for classification." *Intelligent data analysis* 1, no. 1-4 (1997): 131-156.

[11] Koller, D. and Sahami, M., Toward optimal feature selection. In: Proceedings of International Conference on Machine Learning, 1996.

[12] Langley, p. Selection of relevant features in machine learning. In: Proceedings of the AAAI Fall Symposium on Relevance, l-5, 1994.

[13] Erik Schaffernicht1, Volker Stephan2, and Horst-Michael GroB, An Efficient Search Strategy for Feature Selection Using Chow-Liu Trees : Springer-Verlag Berlin Heidelberg 2007.

[14] NING ZHONG, JUZHEN DONG, SETSUO OHSUGA. Using Rough Sets with Heuristics for Feature Selection: Kluwer Academic Publishers. Manufactured in The Netherlands 2001.

[15] METAHEURISTICS FROM DESIGN TO IMPLEMENTATION El-Ghazali Talbi
University of Lille – CNRS – INRIA 2009.

[16] Abdel-Rahman Hedar , Jue Wang , Masao Fukushima, Tabu search for attribute reduction in rough set theory, Springer-Verlag 2007