A New Rough Set Reduct Algorithm Based on Particle Swarm Optimization

Benxian Yue[1,2], Weihong Yao[1], Ajith Abraham[3,4,⋆] and Hongbo Liu[1,4]

[1] Department of Computer, Dalian University of Technology, Dalian 116023, China
yuebenxian@vip.sina.com, ywh@dl.cn, tenghf@dlut.edu.cn, lhb@dlut.edu.cn
[2] School of Mechanical and Engineeing, Dalian Universityof Technology, Dalian 116023, China
[3] Centre for Quantifiable Quality of Service in Communication Systems, Norwegian University of Science and Technology, Trondheim, Norway
ajith.abraham@ieee.org
http://www.softcomputing.net/
[4] School of Computer Science, Dalian Maritime University, Dalian 116026, China

Abstract. Finding appropriate features is one of the key problems in the increasing applications of rough set theory, which is also one of the bottlenecks of the rough set methodology. Particle Swarm Optimization (PSO) is particularly attractive for this challenging problem. In this paper, we attempt to solve the problem using a particle swarm optimization approach. The proposed approach discover the best feature combinations in an efficient way to observe the change of positive region as the particles proceed through the search space. We evaluate the performance of the proposed PSO algorithm with Genetic Algorithm (GA). Empirical results indicate that the proposed algorithm could be an ideal approach for solving the feature reduction problem when other algorithms failed to give a better solution.

1 Introduction

Rough set theory [1,2,3] provides a mathematical tool that can be used for both feature selection and knowledge discovery. It helps us to find out the minimal attribute sets called '*reducts*' to classify objects without deterioration of classification quality and induce minimal length decision rules inherent in a given information system. The idea of reducts has encouraged many researchers in studying the effectiveness of rough set theory in a number of real world domains, including medicine, pharmacology, control systems, fault-diagnosis, text categorization, social sciences, switching circuits, economic/financial prediction, image processing, and so on [4,5,6,7,8,9,10].

Usually real world objects are the corresponding tuple in some decision tables. They store a huge quantity of data, which is hard to manage from a computational point of view. Finding reducts in a large information system is still an

⋆ Corresponding author.

J. Mira and J.R. Álvarez (Eds.): IWINAC 2007, Part I, LNCS 4527, pp. 397–406, 2007.

NP-hard problem [11,15]. The high complexity of this problem has motivated investigators to apply various approximation techniques to find near-optimal solutions. Many approaches have been proposed for finding reducts, e.g., discernibility matrices, dynamic reducts, and others [12,13]. The heuristic algorithm is a better choice. Hu [14] proposed a heuristic algorithm using discernibility matrix. The approach provided a weighting mechanism to rank attributes. Zhong [15] presented a wrapper approach using rough sets theory with greedy heuristics for feature subset selection. The aim of feature subset selection is to find out a minimum set of relevant attributes that describe the dataset as well as the original all attributes do. So finding reduct is similar to feature selection. Zhong's algorithm employed the number of consistent instances as heuristics. Banerjee [16] presented various attempts of using Genetic Algorithms (GA) in order to obtain reducts. Although several variants of reduct algorithms are reported in the literature, at the moment, there is no accredited best heuristic reduct algorithm. So far, it's still an open research area in rough sets theory.

Particle swarm algorithm is inspired by social behavior patterns of organisms that live and interact within large groups. In particular, it incorporates swarming behaviors observed in flocks of birds, schools of fish, or swarms of bees, and even human social behavior, from which the Swarm Intelligence (SI) paradigm has emerged [17]. The swarm intelligent model helps to find optimal regions of complex search spaces through interaction of individuals in a population of particles [18,19]. As an algorithm, its main strength is its fast convergence, which compares favorably with many other global optimization algorithms [20,21]. It has exhibited good performance across a wide range of applications [22,23,24,25,26]. The particle swarm algorithm is particularly attractive for feature selection as there seems to be no heuristic that can guide search to the optimal minimal feature subset. Additionally, it can be the case that particles discover the best feature combinations as they proceed throughout the search space. This paper investigates how particle swarm optimization algorithm may be applied to the difficult problem of finding optimal reducts.

The rest of the paper is organized as follows. Some related terms and theorems on rough set theory are explained briefly in Section 2. The proposed approach based on particle swarm algorithm is presented in Section 3. In Section 4, experiment results and discussions are provided in detail. Finally conclusions are made in Section 5.

2 Rough Set Reduction

The basic concepts of rough set theory and its philosophy are presented and illustrated with examples in [1,2,3,15,27,28]. Here, we illustrate only the relevant basic ideas of rough sets that are relevant to the present work.

In rough set theory, an information system is denoted in 4-tuple by $S = (U, A, V, f)$, where U is the universe of discourse, a non-empty finite set of N

objects $\{x_1, x_2, \cdots, x_N\}$. A is a non-empty finite set of attributes such that $a : U \to V_a$ for every $a \in A$ (V_a is the value set of the attribute a).

$$V = \bigcup_{a \in A} V_a$$

$f : U \times A \to V$ is the total decision function (also called the information function) such that $f(x, a) \in V_a$ for every $a \in A$, $x \in U$. The information system can also be defined as a decision table by $S = (U, C, D, V, f)$. For the decision table, C and D are two subsets of attributes. $A = \{C \cup D\}$, $C \cap D = \emptyset$, where C is the set of input features and D is the set of class indices. They are also called condition and decision attributes, respectively.

Let $a \in C \cup D$, $P \subseteq C \cup D$. A binary relation $IND(P)$, called an equivalence (indiscernibility) relation, is defined as follows:

$$IND(P) = \{(x, y) \in U \times U \quad | \quad \forall a \in P, f(x, a) = f(y, a)\} \tag{1}$$

The equivalence relation $IND(P)$ partitions the set U into disjoint subsets. Let $U/IND(P)$ denote the family of all equivalence classes of the relation $IND(P)$. For simplicity of notation, U/P will be written instead of $U/IND(P)$. Such a partition of the universe is denoted by $U/P = \{P_1, P_2, \cdots, P_i, \cdots\}$, where P_i is an equivalence class of P, which is denoted $[x_i]_P$. Equivalence classes U/C and U/D will be called condition and decision classes, respectively.

Lower Approximation: Given a decision table $T = (U, C, D, V, f)$. Let $R \subseteq C \cup D$, $X \subseteq U$ and $U/R = \{R_1, R_2, \cdots, R_i, \cdots\}$. The R-lower approximation set of X is the set of all elements of U which can be with certainty classified as elements of X, assuming knowledge R. It can be presented formally as

$$R_-(X) = \bigcup \{R_i \quad | \quad R_i \in U/R, R_i \subseteq X\} \tag{2}$$

Positive Region: Given a decision table $T = (U, C, D, V, f)$. Let $B \subseteq C$, $U/D = \{D_1, D_2, \cdots, D_i, \cdots\}$ and $U/B = \{B_1, B_2, \cdots, B_i, \cdots\}$. The B-positive region of D is the set of all objects from the universe U which can be classified with certainty to classes of U/D employing features from B, i.e.,

$$POS_B(D) = \bigcup_{D_i \in U/D} B_-(D_i) \tag{3}$$

Reduct: Given a decision table $T = (U, C, D, V, f)$. The attribute $a \in B \subseteq C$ is $D - dispensable$ in B, if $POS_B(D) = POS_{(B - \{a\})}(D)$; otherwise the attribute a is $D - indispensable$ in B. If all attributes $a \in B$ are $D - indispensable$ in B, then B will be called $D - independent$. A subset of attributes $B \subseteq C$ is a $D - reduct$ of C, iff $POS_B(D) = POS_C(D)$ and B is $D - independent$. 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.

Reduced Positive Universe and *Reduced Positive Region*: Given a decision table $T = (U, C, D, V, f)$. Let $U/C = \{[u_1^{'}]_C, [u_2^{'}]_C, \cdots, [u_m^{'}]_C\}$, Reduced Positive Universe $U^{'}$ can be written as:

$$U^{'} = \{u_1^{'}, u_2^{'}, \cdots, u_m^{'}\}. \tag{4}$$

and

$$POS_C(D) = [u_{i_1}^{'}]_C \cup [u_{i_2}^{'}]_C \cup \cdots \cup [u_{i_t}^{'}]_C. \tag{5}$$

Where $\forall u_{i_s}^{'} \in U^{'}$ and $|[u_{i_s}^{'}]_C/D| = 1 (s = 1, 2, \cdots, t)$. Reduced positive universe can be written as:

$$U_{pos}^{'} = \{u_{i_1}^{'}, u_{i_2}^{'}, \cdots, u_{i_t}^{'}\}. \tag{6}$$

and $\forall B \subseteq C$, reduced positive region

$$POS_B^{'}(D) = \bigcup_{X \in U'/B \wedge X \subseteq U_{pos}^{'} \wedge |X/D|=1} X \tag{7}$$

where $|X/D|$ represents the cardinality of the set X/D. $\forall B \subseteq C$, $POS_B(D) = POS_C(D)$ if $POS_B^{'} = U_{pos}^{'}$ [28]. It is to be noted that $U^{'}$ is the reduced universe, which usually would reduce significantly the scale of datasets. It provides a more efficient method to observe the change of positive region when we search the reducts. We didn't have to calculate U/C, U/D, U/B, $POS_C(D)$, $POS_B(D)$ and then compare $POS_B(D)$ with $POS_C(D)$ to determine whether they are equal to each other or not. We only calculate U/C, $U^{'}$, $U_{pos}^{'}$, $POS_B^{'}$ and then compare $POS_B^{'}$ with $U_{pos}^{'}$.

3 Particle Swarm Approach for Reduction

Given a decision table $T = (U, C, D, V, f)$, the set of condition attributes, C, consist of m attributes. We set up a search space of m dimension for the reduction problem. Accordingly each particle's position is represented as a binary bit string of length m. Each dimension of the particle's position maps one condition attribute. The domain for each dimension is limited to 0 or 1. The value '1' means the corresponding attribute is selected while '0' not selected. Each position can be "decoded" to a potential reduction solution, an subset of C. The particle's position is a series of priority levels of the attributes. The sequence of the attribute will not be changed during the iteration. But after updating the velocity and position of the particles, the particle's position may appear real values such as 0.4, etc. It is meaningless for the reduction. Therefore, we introduce a discrete particle swarm optimization for this combinatorial problem.

During the search procedure, each individual is evaluated using the fitness. According to the definition of rough set reduct, the reduction solution must ensure the decision ability is the same as the primary decision table and the number of attributes in the feasible solution is kept as low as possible. In our algorithm, we first evaluate whether the potential reduction solution satisfies

$POS_E^{'} = U_{pos}^{'}$ or not (E is the subset of attributes represented by the potential reduction solution). If it is a feasible solution, we calculate the number of '1' in it. The solution with the lowest number of '1' would be selected. For the particle swarm, the lower number of '1' in its position, the better the fitness of the individual is. $POS_E^{'} = U_{pos}^{'}$ is used as the criterion of the solution validity.

As a summary, the particle swarm model consists of a swarm of particles, which are initialized with a population of random candidate solutions. They move iteratively through the d-dimension problem space to search the new solutions, where the fitness f can be measured by calculating the number of condition attributes in the potential reduction solution. Each particle has a position represented by a position-vector $\boldsymbol{p}_i$ (i is the index of the particle), and a velocity represented by a velocity-vector $\boldsymbol{v}_i$. Each particle remembers its own best position so far in a vector $\boldsymbol{p}_i^{\#}$, and its j-th dimensional value is $p_{ij}^{\#}$. The best position-vector among the swarm so far is then stored in a vector $\boldsymbol{p}^*$, and its j-th dimensional value is p_j^*. When the particle moves in a state space restricted to zero and one on each dimension, the change of probability with time steps is defined as follows:

$$P(p_{ij}(t) = 1) = f(p_{ij}(t-1), v_{ij}(t-1), p_{ij}^{\#}(t-1), p_j^*(t-1)). \tag{8}$$

where the probability function is

$$sig(v_{ij}(t)) = \frac{1}{1 + e^{-v_{ij}(t)}}. \tag{9}$$

At each time step, each particle updates its velocity and moves to a new position according to Eqs.(10) and (11):

$$v_{ij}(t) = wv_{ij}(t-1) + c_1 r_1 (p_{ij}^{\#}(t-1) - p_{ij}(t-1)) + c_2 r_2 (p_j^*(t-1) - p_{ij}(t-1)). \tag{10}$$

$$p_{ij}(t) = \begin{cases} 1 & \text{if } \rho < sig(v_{ij}(t)); \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

Where c_1 is a positive constant, called as coefficient of the self-recognition component, c_2 is a positive constant, called as coefficient of the social component. r_1 and r_2 are the random numbers in the interval [0,1]. The variable w is called as the inertia factor, which value is typically setup to vary linearly from 1 to near 0 during the iterated processing. ρ is random number in the closed interval [0, 1]. From Eq.(10), a particle decides where to move next, considering its current state, its own experience, which is the memory of its best past position, and the experience of its most successful particle in the swarm. The pseudo-code for the particle swarm search method is illustrated in Algorithm 1..

4 Experiment Settings, Results and Discussions

In this experiment, Genetic algorithm (GA) was used to compare the performance with PSO. The two algorithms share many similarities [29,30]. Both

Algorithm 1. A Rough Set Reduct Algorithm Based on Particle Swarm

01.Calculate $U^{'}$, $U_{pos}^{'}$ using Eqs.(4) and (6).
02.Initialize the size of the particle swarm n, and other parameters.
03.Initialize the positions and the velocities for all the particles randomly.
04.While (the end criterion is not met) do
05. $t = t + 1$;
06.   Calculate the fitness value of each particle,
06.   if $POS_E^{'} = U_{pos}^{'}$, the fitness is punished
06.   as the total number of the condition attributes,
06.   else the fitness is the number of '1' in the position.
07. $\boldsymbol{p}^* = argmin_{i=1}^{n}(f(\boldsymbol{p}^*(t-1)), f(\boldsymbol{p}_1(t)), f(\boldsymbol{p}_2(t)), \cdots, f(\boldsymbol{p}_i(t)), \cdots, f(\boldsymbol{p}_n(t)))$;
08.   For i= 1 to n
09. $\boldsymbol{p}_i^{\#}(t) = argmin_{i=1}^{n}(f(\boldsymbol{p}_i^{\#}(t-1)), f(\boldsymbol{p}_i(t))$;
10.   For j = 1 to d
11.   Update the j-th dimension value of $\boldsymbol{p}_i$ and $\boldsymbol{v}_i$
11.   according to Eqs.(10) and (11);
12.   Next j
13.   Next i
14.End While.

methods are valid and efficient methods in numeric programming and have been employed in various fields due to their strong convergence properties. In GA, the probability of crossover is set to 0.8 and the probability of mutation is set to 0.08. In PSO, self coefficient c_1 and social coefficient c_2 both are 1.49, and the inertia weight w is decreasing linearly from 0.9 to 0.1. The size of the population in GA and the swarm size in PSO both are set to $(int)(10+2*sqrt(D))$, where D is the dimension of the position, i.e., the number of condition attributes. In each trial, the maximum number of iterations is $(int)(0.1 * recnum + 10 * (nfields - 1))$, where $recnum$ is the number of records/rows and $nfields$ is the number of condition attributes. Each experiment (for each algorithm) was repeated 3 times with different random seeds. If the standard deviation is larger than 20%, the times of trials would be set to larger, 10 or 20. We consider the datasets in Table 1 from AFS[1], AiLab[2] and UCI[3].

Figs. 1, 2 and 3 illustrate the performance of the algorithms for lung-cancer, lymphography and mofn-3-7-10 datasets, respectively. For lung-cancer dataset, the results (the number of reduced attributes) for 3 GA runs all were 10: {1, 3, 9, 12, 33, 41, 44, 47, 54, 56} (The number before the colon is the number of condition attributes, the numbers in brackets are attribute index, which represents a reduction solution). The results of 3 PSO runs were 9: { 3, 8, 9, 12, 15, 35, 47, 54, 55}, 10: {2, 3, 12, 19, 25, 27, 30, 32, 40, 56}, 8: {11, 14, 24, 30, 42, 44, 45, 50}. For lymphography datasets, the results of 3 GA runs all were 7: {2, 6, 10,

[1] http://sra.itc.it/research/afs/
[2] http://www.ailab.si/orange/datasets.asp
[3] http://www.datalab.uci.edu/data/mldb-sgi/data/

13, 14, 17, 18}, the results of 3 PSO runs were 6: {2, 13, 14, 15, 16, 18}, 7: {1, 2, 13, 14, 15, 17, 18}, 7: {2, 10, 12, 13, 14, 15, 18}. For mofn-3-7-10 datasets, the results of 3 GA runs all were 7: {3, 4, 5, 6, 7, 8, 9}, the results of 3 PSO runs all were 7: {3, 4, 5, 6, 7, 8, 9}. Other results are shown in Table 1. PSO usually can obtain a better result than GA, specially for a large scale problem. although GA and PSO both got the same results, PSO usually uses only very few iterations, as illustrated in Fig. 2.

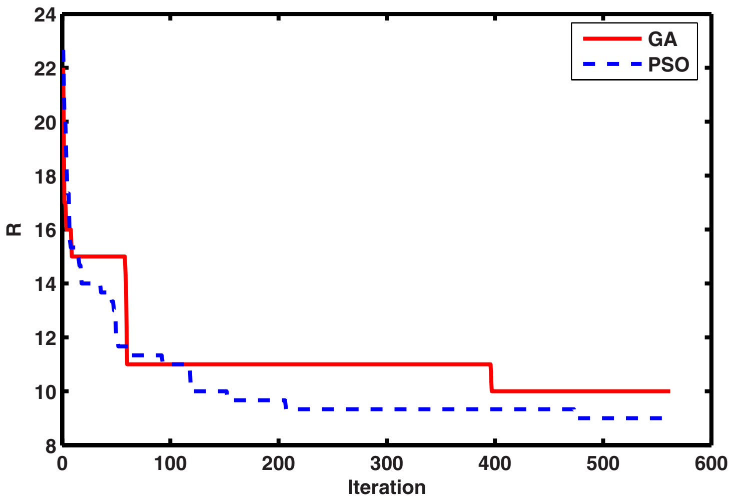

Fig. 1. Performance of rough set reduction for lung-cancer dataset

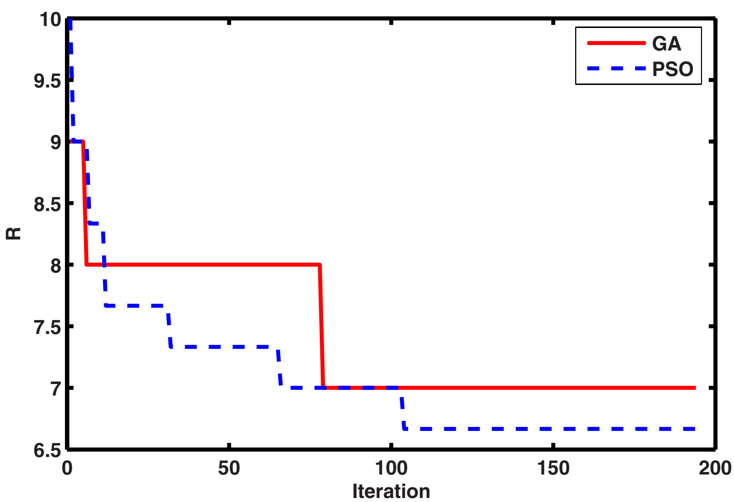

Fig. 2. Performance of rough set reduction for lymphography dataset

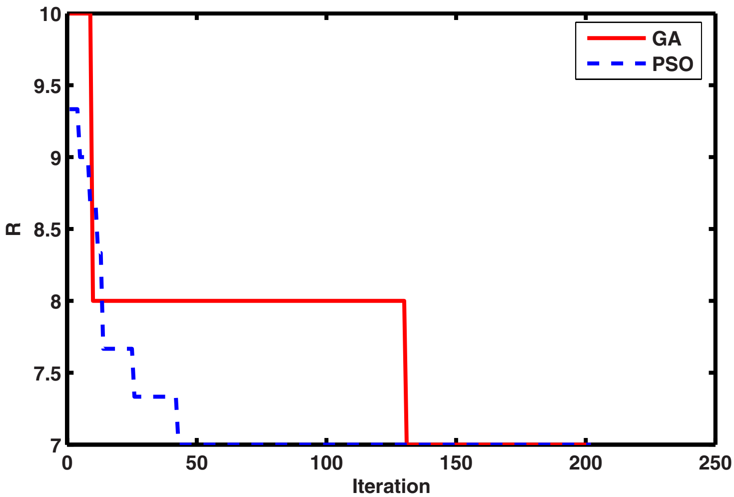

Fig. 3. Performance of rough set reduction for mofn-3-7-10 dataset

Table 1. Datasets used in the experiments

Dataset	*Size*	*ConditionAttributes*	*Class*	*GA*	*PSO*
lung-cancer	32	56	3	10	8
zoo	101	16	7	5	5
corral	128	6	2	4	4
lymphography	148	18	7	6	3
hayes-roth	160	4	3	3	3
shuttle-landing-control	253	6	2	6	6
monks	432	6	2	3	3
xd6-test	512	9	2	9	9
balance-scale	625	4	3	4	4
breast-cancer-wisconsin	683	9	2	4	4
mofn-3-7-10	1024	10	2	7	7
parity5+5	1024	10	2	5	5

5 Conclusions

In this paper, we investigated the problem of finding optimal reducts using a particle swarm optimization approach. The proposed approach discovered the best feature combinations in an efficient way to observe the change of positive region as the particles proceed throughout the search space. We evaluated the performance of the proposed PSO algorithm with Genetic Algorithm (GA). The results indicates that PSO usually required shorter time to obtain better results than GA, specially for large scale problems, although its stability need to be improved in further research. The proposed algorithm could be an ideal approach for solving the reduction problem when other algorithms failed to give a better solution.

Acknowledgments

This work is supported by NSFC (60573087), MOST (2005CB321904), and MOE (KP0302). *Ajith Abraham* is supported by the Centre for Quantifiable Quality of Service in Communication Systems, Centre of Excellence, appointed by The Research Council of Norway, and funded by the Research Council, NTNU and UNINETT.

References

1. Pawlak, Z.: Rough Sets. International Journal of Computer and Information Sciences 11 (1982) 341–356
2. Pawlak, Z.: Rough Sets: Present State and The Future. Foundations of Computing and Decision Sciences 18 (1993) 157–166
3. Pawlak, Z.: Rough Sets and Intelligent Data Analysis. Information Sciences 147 (2002) 1–12
4. Kusiak, A.: Rough Set Theory: A Data Mining Tool for Semiconductor Manufacturing. IEEE Transactions on Electronics Packaging Manufacturing 24 (2001) 44–50
5. Shang, C., Shen, Q.: Rough Feature Selection for Neural NetworkBased Image Classifiction. International Journal of Image and Graphics 2 (2002) 541–555
6. Francis E.H. Tay, Shen, L.: Economic And Financial Prediction Using Rough Sets Model. European Journal of Operational Research 141 (2002) 641–659
7. Świniarski, R. W., Skowron, A.: Rough Set Methods in Feature Selection and Recognition. Pattern Recognition Letters 24 (2003) 833–849
8. Beaubouef, T., Ladner, R., Petry, F.: Rough Set Spatial Data Modeling for Data Mining. International Journal of Intelligent Systems, 19 (2004) 567–584
9. Shen, L., Francis E. H. Tay: Applying Rough Sets to Market Timing Decisions. Decision Support Systems 37 (2004) 583–597
10. Gupta, K. M., Moore ,P. G., Aha, D. W., Pal, S. K.: Rough Set Feature Selection Methods for Case-Based Categorization of Text Documents. Lecture Notes in Computer Science 3776 (2005) 792–798
11. Boussouf, M.: A Hybrid Approach to Feature Selection. Lecture Notes in Artificial Intelligence 1510 (1998) 231–238
12. Skowron, A., Rauszer, C.: The Discernibility Matrices and Functions in Information Systems. In: Świniarski, R. W.(ed.), Handbook of Applications and Advances of the Rough Set Theory, Kluwer Academic Publishers (1992) 331–362
13. Zhang, J., Wang, J., Li, D., He, H., Sun, J.: A New Heuristic Reduct Algorithm Base on Rough Sets Theory. Lecture Notes in Artificial Intelligence 2762 (2003) 247–253
14. Hu, K., Diao, l., Shi, C.: A Heuristic Optimal Reduct Algorithm. Lecture Notes in Computer Science, 1983 (2000) 139–144
15. Zhong, N., Dong, J.: Using Rough Sets with Heuristics for Feature Selection. Journal of Intelligent Information Systems 16 (2001) 199–214

16. Banerjee, M., Mitra, S., Anand, A.: Feature Selection Using Rough Sets. Studies in Computational Intelligence, Springer 16 (2006) 3–20
17. Kennedy, J., Eberhart, R.: Swarm Intelligence, Morgan Kaufmann Publishers, San Francisco, CA (2001)
18. Clerc, M., Kennedy, J.: The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space. IEEE Transactions on Evolutionary Computation 6 (2002) 58–73
19. Clerc, M.: Particle Swarm Optimization. ISTE Publishing Company, London (2006)
20. Parsopoulos, K. E., Vrahatis, M. N.: Recent Approaches to Global Optimization Problems Through Particle Swarm Optimization. Natural Computing 1 (2002) 235–306
21. Abraham, A., Guo, H., Liu, H.: Swarm Intelligence: Foundations, Perspectives and Applications. In: Nedjah, N., Mourelle, L. (eds.), Swarm Intelligent Systems, Studies in Computational Intelligence, Chapter 1, Springer (2006) 3–25
22. Salman, A., Ahmad, I., Al-Madani, S.: Particle Swarm Optimization for Task Assignment Problem. Microprocessors and Microsystems 26 (2002) 363–371
23. Sousa, T., Silva A., Neves A.: Particle Swarm Based Data Mining Algorithms for Classification Tasks. Parallel Computing 30 (2004) 767–783
24. Liu, B., Wang, L., Jin, Y., Tang, F., Huang D.: Improved Particle Swarm Optimization Combined With Chaos. Chaos, Solitons and Fractals 25 (2005) 1261–1271
25. Schute, J. F., Groenwold, A. A.: A Study of Global Optimization Using Particle Swarms. Journal of Global Optimization 31 (2005) 93–108
26. Liu, H., Abraham, A., Choi, O., Moon, S. H.: Variable Neighborhood Particle Swarm Optimization for Multi-Objective Flexible Job-Shop Scheduling Problems. Lecture Notes in Computer Science 4247 (2006) 197–204
27. Wang, G.: Rough Reduction in Algebra View and Information View. International Journal of Intelligent Systems 18 (2003) 679–688
28. Xu, Z., Liu, Z., Yang, B., Song, W.: A Quick Attibute Reduction Algorithm with Complexity of Max$(O(|C||U|), O(|C|^2|U/C|))$. Chinese Journal of Computers 29 (2006) 391–399
29. Cantú-Paz, E.: Efficient and Accurate Parallel Genetic Algorithms, Kluwer Academic publishers, Netherland (2000)
30. Abraham A., Evolutionary Computation, Handbook for Measurement Systems Design, Peter Sydenham and Richard Thorn (Eds.), John Wiley and Sons Ltd., London, ISBN 0-470-02143-8, (2005) pp. 920-931