Journal of Universal Computer Science, vol. 14, no. 15 (2008), 2582-2596
submitted: 18/2/08, accepted: 30/6/08, appeared: 1/8/08 © J.UCS

Two Step Swarm Intelligence to Solve the Feature Selection Problem

Yudel Gómez, Rafael Bello, Amilkar Puris, María M. García
(Universidad Central de Las Villas, Santa Clara, Cuba
{ygomezd, rbellop, ayudier, mmgarcia}@uclv.edu.cu)

Ann Nowe
(Vrije Universiteit Brussel, Belgium
ann.nowe@vub.ac.be)

Abstract: In this paper we propose a new approach to Swarm Intelligence called Two-Step Swarm Intelligence. The basic idea is to split the heuristic search performed by agents into two stages. In the first step the agents build partial solutions which, are used as initial states in the second step. We have studied the performance of this new approach for the Feature Selection Problem by using Ant Colony Optimization and Particle Swarm Optimization. The feature selection is based on the reduct concept of the Rough Set Theory. Experimental results obtained show that Two-step approach improves the performance of ACO and PSO metaheuristics when calculating reducts in terms of computation time cost and the quality of reducts.

Keywords: Feature Selection Problem, Swarm Intelligence, Ant Colony Optimization, Particle Swarm Optimization, Rough Set Theory, Two-Step Swarm Intelligence
Categories: I.2.6, I.2.8, I.5.1

1 Introduction

Biologically or nature inspired systems, methods and technologies represent a very broad area covering many interesting topics, Swarm intelligence is an example of this. Swarm intelligence[Bonabeau, 99] can be defined as the collective intelligence that emerges from a group of simple entities; these agents enter into interactions, sense and change their environment locally. There are two popular swarm inspired methods in computational areas: Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO).

Ant Colony Optimization is inspired by the behaviors of ants and has many applications in discrete optimization problems. The approach relies on a metaheuristic which is used to guide other heuristics in order to obtain better solutions than those that are generated by local optimization methods. In ACO a colony of artificial ants cooperates to look for good solutions to discrete problems. Artificial ants are simple agents that incrementally build a solution by adding components to a partial solution under construction. This computational model was introduced by M. Dorigo. Information about this metaheuristic can be found in [Dorigo, 96a] [Dorigo, 99b] [Dorigo, 99c].

Particle Swarm Optimization is a population based stochastic optimization technique developed by Eberhart and Kennedy in 1995 [Kennedy, 95a] and [Kennedy, 95b], inspired by social behavior of bird flocking. This is a metaheuristic for the optimization of continuous functions; but, in 1997, Kennedy and Eberhart proposed a kind of discrete particle swarm optimization algorithm [Kennedy, 97].

The feature selection, the problem being studied here, can be seen as a discrete problem. The feature selection problem (FSP) can be viewed as a particular case of a more general subset selection problem in which the goal is to find a subset maximizing some adopted criterion. Feature selection methods search through the subsets of features and try to find the best subset among the competing 2^N-1 candidate subsets according to some evaluation measure, where N denotes the total number of features.

Feature selection is useful in different computational tasks, for instance, in machine learning processes. An appropriate representation space for learning by selecting relevant attributes to the problem domain is a crucial issue for learning systems [Inza, 00] [Stefanowski, 04] [Xing, 01]. Usually, not all features describing the examples are relevant to the classification process and some of them are irrelevant or redundant. Too many irrelevant features increase the complexity of learning process and decrease the accuracy of induced knowledge. Feature selection is useful to reduce the dimensionality problem; it results not only in improving the speed of data manipulation [Zhang, 02], but even in improving the classification rate by reducing the influence of noise [Dong, 03][Somol, 02].

Considering FSP as a search space problem, we can consider each state representing a subset of features. All feature selection methods contain two important components: an Evaluation function used to evaluate a candidate feature subset and a Search algorithm to search through the feature space. Search strategies are important because the feature selection process may be time consuming and an exhaustive search for the "optimal" subset is impractical for even moderate sized problems [Zhang, 02]. Examples of search strategies are heuristic search, probabilistic methods and hybrid algorithms.

Swarm intelligence techniques have been used as the search algorithm in the feature selector. Methods which combine ACO and Rough Set Theory (RST) to find reducts with promising results were proposed in [Bello, 05a] [Bello, 05a] [Bello, 05a] [Jensen, 03]. They are based on the reduct concept. A new algorithm to find minimal rough set reducts by using PSO was introduced in [Wang, 05] [Wang, 07]. In this case, a binary representation of the particles was used. The experimental results developed in that work showed that PSO is efficient for rough set-based feature selection. The application of this approach for rule learning was presented in [Wang, 06]. Other applications of PSO in machine learning appear in [Settles, 03] [Sousa, 03].

In this paper, a new feature selector method based on a new approach to Swarm intelligence is proposed. The feature selector looks for reducts, the search process is implemented by using ACO or PSO, and a measure based on rough sets is used to build the evaluation function. The basic idea is to split the heuristic search performed by agents (particles or ants) into two stages. In the first step the agents build partial solutions which are used as initial states in the second step. Some parameters are set up according to the stage. In the following, the basic elements of RST, PSO and ACO are presented. Next, the new approach is introduced. After that, the performance of it is studied.

2 About Rough Set Theory

Rough Sets Theory was proposed by Z. Pawlak [Pawlak82]. The rough set philosophy is founded on the assumption that some information is associated with every object of the universe of discourse [Polkowski, 02]. Rough set data analysis is one of the main techniques arising from RST, it provides a technique for gaining insights into the data properties [Xu, 02]. The rough set model has several advantages for data analysis. It is based on the original data only and does not need any external information; no assumptions about data are made; it is suitable for analyzing both quantitative and qualitative features, and the results of rough set model are easy to interpret [Tay, 02].

In RST a training set can be represented by a table where each row represents objects and each column represents an attribute. This table is called an Information System; formally, it is a pair S= (U, A), where U is a non-empty finite set of objects called the universe and A is a non-empty finite set of attributes. A Decision System is a pair DS=(U, $A\cup\{d\}$), where $d\notin A$ is the decision feature. The basic concepts of RST are the lower and upper approximations of a subset $X\subseteq U$ [Komorowski, 99]. These were originally introduced with reference to an indiscernibility relation IND(B), where objects *x* and *y* belong to IND(B) if and only if *x* and *y* are indiscernible from each other by features in B.

Let be $B\subseteq A$ and $X\subseteq U$. It can be proved that B defines an equivalence relation. The set X can be approximated using only the information contained in B by constructing the B-lower and B-upper approximations of X, denoted by B_*X and B^*X respectively, where $B_*X=\{x : [x]B \subseteq X \}$ and $B^*X=\{x : [x]B \cap X\neq\phi\}$, and [x]B denotes the equivalence class of x according to B-indiscernible relation. The objects in B_*X are guaranteed to be members of X, while the objects in B^*X are possibly members of X. If $B^*X- B_*X$ is not empty, then X is a rough set.

RST offers several measures about a Decision System. Among them is the quality of the approximation of classification (expression 1). It expresses the percentage of objects which can be correctly classified into the given classes Y= $\{Y_1, ..., Y_n\}$ employing only the set of features in B.

$$\gamma_B(Y) = \frac{\sum_{i=1}^{n} |B_* Y_i|}{|U|} \qquad (1)$$

An important issue in the RST is feature reduction based on the reduct concept. A reduct is a minimal set of features that preserves the partitioning of universe and hence the ability to perform classifications. The subset B is a reduct if IND(A)=IND(B); that is, $\gamma A(Y)=\gamma B(Y)$. The concept of reduct is one of the most important concepts RST.

However, the practical use is limited because of the computational complexity of calculating reducts. The problem of finding a globally minimal reduct for a given information system is NP-hard. For that reason, methods for calculating reducts have been developed using heuristic methods [Jensen, 03]. In this paper, ACO and PSO metaheuristics are used, and a new approach to implement these is introduced.

3 Swarm Intelligence

Swarm Intelligence includes several heuristics methods such as Particle Swarm Optimization and Ant Colony Optimization.

3.1 Particle Swarm Optimization

Particle swarm optimization uses a population of particles. Each particle is a potential solution. The system is initialized with a population of random solutions and searches for optima, according to some fitness function, by updating particles over generations; that is, particles ¨fly¨ through the N-dimensional problem search space by following the current better-performing particles.

Each particle remembers its own best position *Xpbest* (that is, where the function was fittest), and of all these, the globally best value *Xgbest* is determined. As showed in expression (2), the particles are attracted by *Xpbest* and *Xgbest*. At each iteration the particle velocity vectors V are modified according to (2). Factor c1 and c2 are empirically determined; they are used to establish a balance between exploration and convergence. The new position is calculated using expression (3).

$$V_i= w*V_i + c1*rand()*(Xpbest-X_i) + c2*rand()*(Xgbest-X_i) \qquad (2)$$

$$X_i(t+1)= X_i(t)+ V_i \qquad (3)$$

Where V_i, X_i, *Xpbest* and *Xgbest* are N-dimensional vectors. The parameter w is the inertia weight; a suitable selection of w provides a balance between global and local exploration.

In feature selection problem, we have an N-dimensional space, where N is the number of features. The optimal position is the shortest subset with highest classification quality. When PSO is used in the feature selection problem each particle is a N-dimensional binary vector, the value "1" means the corresponding feature is selected, while "0" means not selected.

The algorithm looks for minimal reducts R, that is, minimal subsets of features with equal value of the quality of the approximation of classification ($\gamma_R(Y)$) to all features ($\gamma_A(Y)$). In this case, the fitness function is the same used in [Wang, 07], see expression (4). This expression means the quality and the feature subset's length are taking into account in the feature selection task. The goodness of each position is evaluated by this fitness function. The criteria are to maximize fitness values.

$$Fitness = \alpha * \gamma_R(D) + \beta * \frac{N - |R|}{N} \qquad (4)$$

Taking into account the binary representation used in this proposal, it is necessary to redefine expression (3). The movement of the particle is realized by flip of bit value, and the velocity is no longer a change ratio of its position but a change probability of its position. We propose expression (5), based on the equations of the position and the velocity of the particle showed in [Kennedy, 97] [Yuan, 98], to calculate the value of dimension j in particle i.

$$X_{ij}(t+1)=\begin{cases}1 & if\ \ rand()\le \dfrac{1}{1+e^{1.5*N*Vij}}\\ 0 & othercase\end{cases} \qquad (5)$$

The value of the parameter inertia weight (w) is in the interval [1.4, 0.4], it is calculated by using a positive linear function changing according to the cycle iteration, see expression (6).

$$w=(w-0.4)*(ncMAX-k)/ncMAX+0.4 \qquad (6)$$

Where *ncMAX* denotes the number maximum of cycles or generations and k is the current cycle.

3.2 Ant Colony Optimization

Ant System (AS) is the first ACO algorithm; it will be described using the Travelling Salesman Problem (TSP). In TSP, we have a set of *N* fully connected cities $\{c_1, \ldots, c_n\}$ by arcs (i,j); each arc is assigned a weight d_{ij} which represents the distance between cities *i* and *j*, the goal is to find the shortest possible trip visiting each city once before returning to initial city. When ACO is used to solve this problem, pheromone trails (♦$_{ij}$) are associated to arcs which denote the desirability of visiting city *j* directly from city *i*. Also, the function ♒$_{ij}$= 1/d$_{ij}$ indicates the heuristic desirability of going from *i* to *j*, where d_{ij} is the distance between cities *i* and *j*. Initially, ants are randomly associated to cities. In the successive steps ant k applies a random proportional rule to decide which city to visit next according to expression (7):

$$p_{ij}^{k}=\frac{(\tau_{ij})^{\alpha}*(\eta_{ij})^{\beta}}{\sum_{l\in N_i^k}(\tau_{il})^{\alpha}*(\eta_{il})^{\beta}} \quad if \quad j\in N_i^k \quad (neighborhood \quad of \quad ant\ k) \qquad (7)$$

where α and ♌ are two parameters to point out the relative importance of the pheromone trail and the heuristic information respectively. After all ants have built their tours the values ♦$_{ij}$ are updated in two stages. First, ♦$_{ij}$ values are decreased by evaporation, ♦$_{ij}$=(1-❒)*♦$_{ij}$, using the parameter ❒, where 0<❒<1. This is to avoid unlimited accumulation of pheromone. Second, all ants increase de value of ♦$_{ij}$ on the arcs they have crossed in their tours, ♦$_{ij}$=♦$_{ij}$+Inc$_{ij}$, where Inc$_{ij}$ is the amount of pheromone deposited by all ants which included the arc (i,j) in their tour. Usually, the amount of pheromone deposited by ant *k* is equal to 1/C$_k$, where C_k is the length of the tour of ant *k*.

Some direct successor algorithms of Ant Systems are: Elitist AS, Rank-based AS and MAX-MIN AS.

A more different ACO algorithm is Ant Colony System (ACS). ACS uses the following pseudorandom proportional rule (8) to select the next city j from city i.

$$j = \begin{cases} \arg\max_{l \in N_i^k} \left\{ \tau_{ij} * (\eta_{il})^{\beta} \right\} & if \quad q \leq q_0 \\ random\ selection\ according\ to\ (7) & otherwise \end{cases} \tag{8}$$

where q is a random variable uniformly distributed in [0,1], q_0 which is a constant taken in the interval [0,1], controls the amount of exploration. In ACS, ants have a local pheromone trail update (♦$_{ij}$=(1-☒)* ♦$_{ij}$+ ☒*♦$_{ij}$(0)) applied after crossing an arc(i,j), where ♦$_{ij}$(0) represents the initial value for the pheromone, and a global pheromone trail update (♦$_{ij}$=(1-❒)* ♦$_{ij}$ + ❒*Inc$_{ij}$) executed only by the best-so-far ant.

When ACO is used to solve the feature selection problem the graph representation is lightly different to the TSP case. This problem can be modeled in the following way. Let A={a_1, a_2, a_{na}} be a set of features. You can view this set as a network in which nodes represent features, and all nodes are connected by bi-directional links. Pheromone values τ_i are associated to nodes a_i, τ_i represents the absolute contribution of the feature a_i to a reduct. In the first step, each ant k is assigned to one node, it can move to all nodes in the network (b_k = {a_i}, where b_k is the subset ant k has to build). Ants perform a forward selection in which each ant k expands its subset b_k step-by-step by adding new features; for performing it, each ant k looks for all features in the set A-b_k and selects the next feature among them to include in b_k according to the ACO rule. This rule is the pseudorandom proportional rule (expression 8) in the case of ACS. The quality of the approximation of classification measure of RST (expression (1)) is used like a heuristic function (η) in the ACO model (η(B)=γ_B(Y)).

The ACS algorithm showed the best performance when a comparative study about several ACO algorithms was developed to solve the feature selection problem as is stated in [Bello, 05b] and [Bello, 05c]. For this reason, we use the ACS algorithm to study the two stage approach.

4 Two-Step Swarm Intelligence

The Two-Step Swarm Intelligence proposed in this paper is based on the following idea: to divide the search process made by the agents (ant or particles) in two stages, so that in the first stage preliminary results are reached which are used to build the initial swarm for the second stage. In the case of FSP, this means that subsets of features generated in the first stage are potential reducts used as initial population of the second stage.

4.1 The Two-Step approach

The determination of the initial state in which the search process starts has been an interesting problem in heuristic search. It is well known that the initial state has an important effect in the search process. The aim is to be able to approach the initial state to the goal state as close as possible. Of course, it is necessary to consider an adequate balance between the computational cost of obtaining that initial state and the total cost; in other words, the sum of the cost of approaching the initial state towards the goal state plus the cost of finding the solution from that "improved" initial state should not be greater than the cost of looking for the solution from a random initial state.

Formally, the purpose is the following. Let E_i be an initial state randomly generated, or obtained by any other method without a significant computational cost, E_i* is an initial state generated by some method *M* that approaches it to the goal state, $C_M(E_i, E_i*)$ denotes the cost of obtaining E_i* from state E_i using the method *M*, and $CC_{HSA}(x)$ is the computational cost of finding a solution from state *x* using a Heuristic Search Algorithm (HSA). Then, the objective is that $C_M(E_i,E_i*) + CC_{HSA}(E_i*) < CC_{HSA}(E_i)$.

In the Two-step approach proposed here, the procedure to generate E_i* and the HSA are both the ACS or PSO algorithms, so the objective is to obtain $C_{ACS}(E_i,E_i*) + CC_{ACS}(E_i*) < CC_{ACS}(Ei)$ or $C_{PSO}(E_i,E_i*) + CC_{PSO}(E_i*) < CC_{PSO}(Ei)$. As ACO or PSO are used in both stages, the difference between the 2 stages is obtained by giving different values to some parameters of the model in each step. A ratio (*r*) for some parameters is introduced to establish the relative setting of the values of the parameters of the algorithm in both stages; it indicates which proportion of the complete search is given to the first stage. For example, if *r*=0.3 for the parameter number of cycles, means that the first step will cover 30% of the search process and the second step the remaining 70%.

The setting of the ratio *r* has a high influence on the overall performance of the algorithm. A high value of *r*, that is, about value 1, causes the state E_i* to be closer to the goal state, then the value of $C_{ACS}(E_i, E_i*)$ may increase and the value of $CC_{ACS}(E_i*)$ will decrease. But, in addition to this balance between the costs of $C_{ACS}(E_i,E_i*)$ and $CC_{ACS}(E_i*)$, we have the problematic about how much the space search is explored; while more greater is the rate *r*, the search in the second stage decreases for several reasons: (I) there are less ants working, (II) the amount of cycles decreases, and (III) although the quantity of possible initial states for the second stage must grow when *r* grows, that amount is already limited by the result of the previous stage.

Therefore, a key point is to study what value of rate *r* is the best in order to obtain the best balance between the searches in both stages. This value must allow:

- To diminish the value of $C_{HSA}(E_i,E_i*) + CC_{HSA}(E_i*)$.
- To allow an exploration of the search space that guarantees to find good solutions.

4.2 Two-Step ACO

In order to apply the two stage approach we need to decide how to set some parameters. The number of ants and the number of cycles per stage is calculated according to the ratio *r*; that is, let *m* be the quantity of ants to use and *NCmax* the number of cycles to perform, these values are divided for each stage in the following form: First stage {$m_1=m*r$ and $NCmax_1=NCmax*r$}, and second stage {$m_2=m*(1-r)$ and $NCmax_2=NCmax*(1-r)$}. Also, we must define what a partial solution is in the first stage. In this case, the reference value is the quality of the approximation of classification measure $\gamma_A(Y)$, where *A* is the set of all features, and ants have to build subset of features with a quality of the approximation of approximation equal to

r*γ_A(Y). In the second stage we are looking for a solution to the original problem, so the subsets have to obtain the value γ_A(Y).

In other words, in the first stage m_1 ants during $NCmax_1$ cycles will be looking for subset of features with a quality of the approximation of classification equal to r*γ_A(Y). In the second stage, m_2 ants will perform $NCmax_2$ cycles using the subsets resulting from the first stage as initial states in order to generate reducts.

The TS-ACS-RST-FS algorithm has the following structure:
Given parameters *alpha, beta, NCmax* (total number of cycles), *na, ratio, ro* and *epsilon*

```
S0: Calculate the quantity of ants (m) according to the number of features (na).
    Calculate the quality of the classification using all features ($\gamma_B(Y)$ for B=A).
S1: Stage 1
    Calculate value for parameters in the first stage:
        $NCmax_1$=ratio*NCmax
        $m_1$= ratio*m
        $\gamma_{B1}(Y)$=ratio* $\gamma_A(Y)$
    Apply ACS-RST-FS algorithm.
    Candidate subsets = subSetselected by the ACS-RST-FS algorithm.
S2: Stage 2
    Calculate value for parameters in the second stage:
        $NCmax_2$=NCmax-$NCmax_1$
        $m_2$= m-$m_1$
        $\gamma_{B2}(Y)$= $\gamma_A(Y)$
    Apply ACS-RST-FS algorithm, in each cycle assign a random subset from
    Candidate subsets as initial subset for each ant.
```

In the step S0, the calculation of the number of ants could be random, however we propose initialize this parameter in function of the number of features as proposed in [Bello, 05a]; the quality of the classification is calculated using all the features, this value indicates the quality that should reach a subset to be considered a reduct. According to the value fixed for the ratio, it is determined the value of the parameters $NCmax_1$, m_1 and γ_{B1}(Y). During the algorithm, in the first stage S1 the best solutions are selected, those constitute potential reducts and will be the initial states for the ants in the second phase. In the second stage S2 the ants resume the search staring from these subsets obtained in the stage 1, which have reached a quality of the classification greater than or equal to r*γA(Y). Throughout the second stage the algorithm ACS-RST-FS is executed completing the number of wanted cycles and each ant works until complete a reduct.

The value of the ratio (r) is an important parameter. Values near to 1 produce subsets near to the definition of reducts in the first stage.

4.3 Two-Step PSO

The Two-step PSO to feature selection problem (TS-PSO-RST-FS) algorithm is define in the following form. The parameters that establish the differences between stages are the following: *ratioQ* and *ratioC*. The first is a value used to define a quality level according to expression (9), it is used to select candidate feature subsets. The rule to select subsets in the first step is: All subsets *R* such that $\gamma_R(Y)$ ¤ *QualityLevel* are selected as a potential reduct. The last is a value used to calculate the quantity of cycles in each stage according to expression (10) and (11)

$$\text{QualityLevel}=\text{ratioQ}^{*}\,\gamma_A(Y),\quad \text{A is the set of all features} \tag{9}$$

$$\text{ncStep1}=\text{round}(\text{ratioC}^{*}\text{ncMAX}) \tag{10}$$

$$\text{ncStep2}=\text{ncMAX} - \text{ncStep1} \tag{11}$$

Where round(x) denotes the closest integer to x.

The TS-PSO-RST-FS algorithm introduces a step between the first and second stages in which the set of potential reducts are used to build two sets: *MostUsedFeatures* and *LessUsedFeatures*, they represent the sets of the most widely used features and the less widely used feature respectively. Both are N-dimensional binary vectors; the features included in these sets have value 1 in the vector. The *MostUsedFeatures* set includes all features belong to a number a potential reducts greater than a given percent, called *PerUsed*, of the reducts; for instance, if *PerUsed*=75%, this means only features which belong to at least to the 75% potential reducts are included in *MostUsedFeatures*. In the other hand, the *LessUsedFeatures* set contains all features belong to at most to *PerNotUsed* percent of potential reducts; for instance, if *PerNotUsed*=30%, this means only features belong to at most to the 30% potential reducts are included in *LessUsedFeatures*. Using *MostUsedFeatures* and *LessUsedFeatures* sets the last swarm of stage 1 is modified to yield the initial swarm of second stage in the following way. Each particle X_i in the last swarm is replaced by the vector *MostUsedFeatures* or is modified by using the vector *LessUsedFeatures* in a random way: *If rand()≤0.5 then Replace else Modify*. Modify means that all feature included in *LessUsedFeatures* receive value "0" in the particle.

Greater values of the inertia weight in the first stage and this processing step help to find good seeds to build the initial swarm for the second stage.

TS-PSO-RST-FS algorithm:

```
S1:
  Generate initial population of Swarm (Xi) and Velocity (Vi).
  Calculate Xpbest for each particle and Xgbest
  Execute PSO during ncStep1 cycles:
  Potential-reducts set =  All subset R such that  γR(Y)¤QualityLevel

Post processing Stage 1:
     Calculate MostUsedFeatures and LessUsedFeatures from potential-reducts set
     Initial swarm for second stage = last swarm of stage 1 modified using
     MostUsedFeatures and
```

```
    LessUsedFeatures according to the Rule If rand()≤0.5 then Replace else
    Modify.

S2:
  Calculate Xpbest for each particle and Xgbest
  Execute PSO during ncStep2 cycles:
  Reducts = All subset R such that $\gamma_R(Y) = \gamma_A(Y)$
```

In the step S1 the algorithm PSO is executed in traditional way during *ncStep1* cycles, all the subsets of resulting features with quality of the classification grater than or equal to *QualityLevel* will be selected. The most used features (*MostUsedFeatures*) and the less used features (*LessUsedFeatures*) are determined once finished this first stage using the selected subsets. The last swarm of the first stage is modified according to this information. For each particle it is generated an random number, if the generated value is smaller than 0.5 the particle is substituted by the vector *MostUsedFeatures*, in another case each component of the particle is modified receiving value zero those that correspond to features that are in *LessUsedFeatures*. This modified swarm is a good initial state for the second phase. In the S2 step the algorithm PSO is executed until completing the quantity of wanted cycles, in each cycle all the particles that represent subsets of features whose quality of the classification equal to the one obtained with all the features are selected. At the end of the process all the super subset should be eliminated.

5 Experimental results

We developed a comparative study of TS-ACS-RST-FS and TS-PSO-RST-FS algorithms respect to the ACS-RST-FS and PSO-RST-FS algorithms. The last one is the following:

PSO-RST-FS algorithm:

```
  Generate initial population of Swarm (Xi) and Velocity (Vi).
  Calculate Xpbest for each particle and Xgbest
  Execute PSO during ncMAX cycles
  Reducts ← All subset R such that $\gamma_R(Y) = \gamma_A(Y)$
```

Particle Swarm Optimization algorithms were executed by using the expressions (1), (2), (4)-(6), (9)-(11) with the following parameters: ncMAX=120, c1=c2=2, quantity of particles equals to 21 and α=0.54. In the case of the TS-PSO-RST-FS algorithm, ratioQ=0.75, ratioC=0.3, PerUsed=66% and PerNotUsed=30%.

In the Two-step approach the values of the ratios have important effects. A low value for ratioQ yields many low-quality potential reducts, consequently the MostUsedFeatures and LessUsedFeatures set include useless information about the features; therefore, the effect of using MostUsedFeatures and LessUsedFeatures to modify the swarm is poor. Otherwise, a value near to 1 produces subsets near to the definition of reducts in the first stage. In the case of ratioC, a low value allows to perform a greater quantity of cycles in the second stage from the modified swarm.

The two algorithms are tested and compared using six databases from UCI Repository [Blake, 98]. In order to use the classical rough set theory, the databases were discretized using the method proposed in [Tay, 02]. Each algorithm was executed six times using each database and the results were averaged over these 6 runs.

In Table 1, the algorithm performances were compared with respect to the resulting average length of the resulting reduct set (Columns (4) and (6)) and the quantity of times in which the algorithm found the minimal reducts (columns (5) and (7)). The length of a reduct is defined by the number of features in the reduct.

Dataset (1)	**Features**(2)	**Instances**(3)	**PSO-FS**(4)	**PSO-FS**(5)	**TS-PSO**(6)	**TS-PSO**(7)
Breast cancer	9	699	4.95	6	4.6	6
Heart	13	294	7.97	4	6.8	6
Exactly	13	1000	6	6	6	6
Credit	20	1000	12.4	4	10.3	5
Dermatolog y	34	358	15.3	3	12.6	5
Lung	56	32	15.6	3	12.8	5

Table 1: A comparison between algorithms PSO-RST-FS and TS-PSO-RST-FS

These results are very interesting because these shows the Two-step PSO approach obtains reducts shorter than the PSO-RST-FS algorithm. So, the certainty of finding minimal reducts increases by using the TS-PSO-RST-FS algorithm.

In the case of Ant Colony Optimization the value of the ratio has a similar effect. For the experiments, we have used the following values for parameter ratio {0.2, 0.3, 0.36, 0.5, 0.6, and 0.8}. The effect of the ratio was studied in three aspects, the quantity of reducts, the length of reducts and the computation time. Each algorithm was executed 10 times and the results were averaged over these 10 runs.

In Table 2, we present the results obtained using a decision system that contains 20 objects which are described by 16 discrete features (na=16) and belongs to two classes. The first class contains 9 objects and the second one contains 11 objects, the quality of the approximation of classification is equal to 1 ($\gamma_A(Y)=1$); so, this is a consistent decision system (each pair of indiscernible objects belongs to the same class). The quantity of cycles was 21 (NCmax=21). In columns number 6 and 7 the information is presented in the following form: averaged number of reducts/averaged length of reducts/time.

Algorithm	**NC max_1**	**NC max_2**	**m_1**	**m_2**	**beta=5 q_0=0.9**	**beta=1 q_0=0.3**
ACS			--	--	46.7/3.95/228s	123/4.19/274s
TS-ACS rate=0.2	4	17	3	13	32.7/4.2/82s	76.3/4.2/89.9s
TS-ACS rate=0.3	6	15	5	11	43.3/4.1/53s	71/4.2/64s
TS-ACS rate=0.36	8	13	6	10	38.7/3.9/39s	67.3/4.1/47s
TS-ACS rate=0.5	10	11	8	8	29.7/3.8/32s	43.3/4.1/44s
TS-ACS rate=0.6	13	8	10	6	20.33/3.8/41s	37/4.2/49s
TS-ACS rate=0.8	17	4	13	3	9/3.8/82s	10.67/4.2/97s

Table 2: A comparison of ACS-RST-FS and TS-ACS-RST-FS using different ratio values

You can see the ratio about to 0.3 yields the best results. This setting obtains a quantity of reduct near to ACS-RST-FS algorithm [Bello, 05c] but only in the 23% of the time. Similar results were obtained using other databases. This is an expected result, because the value ratio=0.3 offers a good balance between both stages; a higher numbers of ants and cycles in the second stages allows the algorithms to perform a larger exploration of the search space from of initial subsets having certain quality.

In Table 3, we present a similar study using Breast Cancer database from UCI Repository [Blake, 98], in this database the number of ants is 9 (na=9) and we used 9 ants (m=9). In columns number 4 and 5 the information is presented in the following form. In the first row, the results obtained by the ACS algorithm (average number of reducts/time) are presented; in the rest of the rows we present the results yielded for the TS-ACS-RST-FS algorithm (percent of average number of reducts/percent of time) respects to the results showed in the first row. For instance, the TS-ACS-FS algorithm with r=0.3, β=5 and q_0=0.9 obtained an average of 50.9 reducts (109% respects to 46.7 reducts) in only 71 seconds (31% respects to 228s). These results are very interesting because the Two-Step ACS approach enable us to obtain equal or greater quantity of reducts in lower time than ACS-RST-FS algorithm.

From tables 2 and 3, it is possible to see the time cost in the case of TS-ACS-RST-FS algorithm is very low. From this advantage, we propose a second idea: to increase the quantity of ants in order to yield a greater exploration of the search space. We developed experiments in which the number of ants was increased by the factors {1.33, 1.5, 1.8, 2.1}. In the last row of Table 3 the results with value 2.1 are showed. The relation is established respect the quantity of reducts and the time. For instance, when the number of ants is increased to 2.1*m (m=2.1*9=19), the algorithm TS-ACS-RST-FS obtains 124% number of reducts respect to the quantity of reducts obtained by ACS-RST-FS algorithm in only 67% of the time used by this last algorithm (for beta=5 and q_0=0.9). We used the rate=0.3 because this value shows the best results in experimental results.

Algorithm	**m_1**	**m_2**	**beta=5 q_0=0.9**	**beta=1 q_0=0.3**
ACS m=9	--	--	46.7/228s	123/274s
TS-ACS rate=0.2	2	7	60%/34%	70/49%
TS-ACS rate=0.3	3	6	109%/31%	73%/37%
TS-ACS rate=0.36	3	6	105%/25%	77%/31%
TS-ACS rate=0.5	4	5	100%/22%	73%/26%
TS-ACS rate=0.6	5	4	65%/13%	50%/20%
TS-ACS rate=0.8	7	2	33%/27%	31%/26%
TS-ACS rate=0.3 m=2.1*m=19	6	13	124%/67%	103%/83%

Table 3: A comparison between ACS-RST-FSP algorithm and several alternatives to TS-ACS-RST-FS algorithm (NCmax1=6, NCmax2=15)

6 Conclusion

We have presented an improvement of the Swarm Intelligence to solve the feature selection problem which consists on splitting the search process developed by agents into two stages. This model to build a feature selector uses Particle Swarm Optimization and Ant Colony Optimization to perform the heuristic search process and the quality of the approximation of classification measure of Rough set theory to build the evaluation function. In this approach some parameters (such as number of ants, quantity of cycles, etc.) receive different values in each stage according to a ratio parameter which indicates what proportion of the complete search corresponds to each stage.

We studied the performance using different ratio values in the feature selection problem. The experimental results showed the performance was increased strongly.

This new approach produces an important reduction of the computation time cost in the case of ACO, and increases the quality of the reducts.

Acknowledgements

The authors would like to thanks VLIR (Vlaamse InterUniversitaire Raad, Flemish Interuniversity Council, Belgium) for supporting this work under the IUC Program VLIR-UCLV.

References

[Bello, 05a] Bello, R., Nowé, A.: "Using ACO and Rough Set Theory to Feature Selection," *WSEAS Transactions on Information Science and Applications*, vol. 2, pp. 512-517, May 2005.

[Bello, 05b] Bello, R., Nowé, A.: "A Model based on Ant Colony System and Rough Set Theory to Feature Selection," in *Genetic and Evolutionary Computation Conference (GECCO05)*, pp. 275-276, Washington DC 2005.

[Bello, 05c] Bello, R., Nowé, A.: "Using Ant Colony System meta-heuristic and Rough Set Theory to Feature Selection," in *The 6th Metaheuristics International Conference (MIC2005)*, Vienna, Austria, 2005.

[Blake, 98] Blake C. L., Merz, C. J.: "UCI Repository of machine learning databases", 1998, *http://www.ics.uci.edu/~mlearn/MLRepository.html*.

[Bonabeau, 99] Bonabeau, E.: "Swarm Intelligence: From natural to Artificial systems," *Oxford University Press*, 1999.

[Dong, 03] Dong, M., Kothari, R.:"Feature subset selection using a new definition of classifiability," *Patter Recognition Letter*, vol. 24, pp. 1215-1225, 2003.

[Dorigo, 96] Dorigo, M.: "The Ant System: Optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man and Cybernetics-Part B*, vol. 26, pp. 1-13, 1996.

[Dorigo, 99b] Dorigo, M., Caro, G. D.: "Ant algorithms for Discrete optimization," *Artificial Life*, vol. 5, pp. 137-172, 1999.

[Dorigo, 04] Dorigo, M., Stutzle, T.: *Ant Colony Optimization*: MIT Press, 2004.

[Eberhart, 95b] Eberhart, R. C., Kennedy, J.: "A new optimizer using particle swarm theory," in *Sixth International Symposium on Micro Machine and Human Science*, pp. 39-43, 1995.

[Inza 00] Inza, I.: "Feature subset selection by bayesian networks based optimization," *Artificial intelligence*, vol. 123, pp. 157-184, 2000.

[Jensen, 03] Jensen R., Shen, Q.: "Finding Rough Set Reducts with Ant Colony Optimization," in *UK Workshop on Computational Intelligence*, 2003, pp. 15-22.

[Kennedy, 95a] Kennedy, J., Eberhart, R. C.: "Particle swarm optimization," in *IEEE International Conference on Neural Networks*, pp. 1942-1948, 1995

[Kennedy, 97] Kennedy, J., Eberhart, R. C.: "A discrete binary version of the particle swarm optimization algorithm," in *IEEE International Conference on Neural Networks*, Perth, Australia, 1997, pp. 4104-4108.

[Komorowski, 99] Komorowski, J., Pawlak, Z.: "Rough Set: A tutorial.," in *Rough Fuzzy Hybridization: A new trend in decision making*, pp. 3-98, 1999.

[Pawlak82] Pawlak, Z.:"Rough sets," *International Journal of Information & Computer Sciences,* vol. 11, pp. 341-356, 1982.

[Polkowski, 02] Polkowski, L.: "Rough sets: Mathematical foundations," *Physica-Verlag,* p. 574, 2002.

[Settles, 03] Settles, M., Rodebaugh, B., Soule, T.: "Comparison of genetic algorithm and particle swarm optimizer when evolving a recurrent neural network," in *Genetic and Evolutionary Computation Conference 2003 (GECCO 2003),* pp. 151-152, Chicago 2003.

[Somol, 02] Somol, P., Pudil, P.:"Feature selection toolbox," *Pattern Recognition,* vol. 35, pp. 2749-2759, 2002.

[Sousa, 03] Sousa, T., Silva, A., Neves, A.: "A particle swarm data miner," in *11th Protuguese Conference on Artificial Intelligence (EPIA 2003).* vol. 2902 Beja, Portugal.: Lecture Notes in Computer Science, pp. 43-53, 2003.

[Stefanowski, 04] Stefanowski, J.: "An experimental evaluation of improving rule based classifiers with two approaches that change representations of learning examples," *Engineering Applications of Artificial Intelligence,* vol. 17, pp. 439-445, 2004.

[Tay, 02] Tay, F. E., Shen, L.: "Economic and financial prediction using rough set model " *European Journal of Operational Research,* pp. 641-659, 2002.

[Tay, 02] Tay, F. E., Shen, L.: "A modified Chi2 algorithm for discretization," *IEEE Transactions on knowledge and data engineering,* vol. 14, May/June 2002

[Wang, 05] Wang X.: "Finding Minimal Rough Set Reducts with Particle Swarm Optimization," *Lectures Notes on Computer Science* vol. 3641, pp. 451-460, 2005.

[Wang, 06] Wang X.: "Rough set feature selection and rule induction for prediction of malignancy degree in brain glioma," *Computer Methods and Programs in Biomedicine,* vol. 83, pp. 147-156, 2006.

[Wang, 07] Wang X.: "Feature selection based on rough sets and particle swarm optimization," in *Pattern Recognition Letters*, pp. 459-471 vol. 28, 2007.

[Xing, 01] Xing, H., L. X. 14, "Feature space theory -a mathematical foundation for data mining.," in *Knowledge-based systems* vol. 14, pp. 253-257, 2001.

[Xu, 02] Xu, Z. B.: "Inclusion degree: a perspective on measures for rough set data analysis.," in *Information Sciences*, pp. 227-236, 2002.

[Yuan, 98] Yuan, S., Chu, F. L.: "Fault diagnostics based on particle swarm optimization and support vector machines," *Mechanical Systems and Signal Processing* vol. 21, pp. 1787-1798, 2007.

[Zhang, 02] Zhang, H., Sun G.: "Feature selection using tabu search method," *Pattern Recognition,* vol. 35, pp. 710-711, 2002.