

These four steps will be repeated until the items are packed. This method was tested on the literature instances, and we obtained encouraging results.

CIE00459 Identification of Run Length Distribution in Residual Control Chart While Monitoring Autoregressive and Moving Average Processes

Seyyed Mohamad Taghi Fatemi Ghomi, Yaser Samimi
Department of Industrial Engineering, Amirkabir University of Technology, Iran

Introducing a definite mathematical relationship to calculate average run length (ARL) in residual control chart for autoregressive process of order p , is the main idea of this paper. In addition, a procedure is proposed to compute ARL in residual control chart for monitoring moving average process of order q . Also, tables of ARL values especially prepared for AR(1) and MA(1) processes are presented. Since 1988 which Special Cause Chart (SCC) was first introduced by Alwan and Roberts, though several researchers have investigated the probability distribution of residuals in AR(1) process, definition of a mathematical formula for ARL in autoregressive processes which is of great interest to compare different monitoring procedures for autocorrelated processes, has been seldom addressed. Invoking mathematical software providing quick solutions for systems of mathematical equations allowed us to implement the proposed approach as a computer program.

CIE00462 Multi-Component Preventive Maintenance Optimization Based on Availability

Mohammad Doostparat, Farhad Kolahan
Department of Mechanical, Ferdowsi University of Mashhad, Iran

In this paper the problem of preventive maintenance (PM) planning for a system with deteriorating components has been addressed. The problem involves a multi-component system with resource constraint and minimum availability requirements. The cost function is weighted summation of repair costs, system downtime cost and random failure cost. Maintenance and repair activities are divided into three actions; namely simple service (inspection), repair and replacement. During the planning horizon, inspections are performed on the regular basis. In each inspection period, one of the three PM activities is carried for each component. The objective is to maintain certain level of availability with minimal total cost. Since the problem is complicated in nature, Simulated Annealing (SA) algorithm is employed as the solution procedure. Computational results show that this algorithm has good performance in solving PM scheduling problems.

CIE00462_2 Evaluating the Discretization of Search Space in Continuous Problems for Ant Colony Optimization

Mahdi Abachizadeh, Farhad Kolahan
Department of Mechanical Engineering, Ferdowsi University of Mashhad, Iran

In this paper, after a beginning on the concept of ant algorithms, a brief survey of the ant-based methods proposed for optimization of problems with continuous design spaces is presented. As a common approach in continuous domains, discretizing the search space is the model presented to be appended to the original ant colony system (ACS) algorithm. Evaluating this method and comparing it to the standard simulated annealing shows that it is robust enough not to fall in local minima. However, when higher resolution is required, the algorithm fails to capture the global optimums and the computational costs rapidly increase. Therefore, it can be safely proposed for the problems in which a trade-off between time, solution accuracy and algorithm intricacy is needed.

CIE00462_3 A Genetic Algorithm Approach For Prediction Of Process Parameters In Submerged Arc Welding

Farhad Kolahan¹, Ahmad Tavakkoli², Mir Masood Bagheri¹

¹*Department of Mechanical Engineering, Ferdowsi University of Mashhad, Iran*

²*Department of Management, Ferdowsi University of Mashhad, Iran*

Among different welding techniques, Submerged Arc Welding (SAW) is one of the most widely used processes employed in metal forming industries. In this paper, a Genetic Algorithm approach is proposed to optimally determine SAW process parameters for any desired weld bead geometry. A five-level factorial technique is employed to relate the important process-control variables (welding voltage, wire feed rate, welding speed and nozzle-to-plate distance) to the bead-quality features (penetration, reinforcement, bead width, total volume of the weld bead and dilution). The adequacy of the proposed approach is verified with ANOVA. Then, the developed models embedded to a GA algorithm to determine the best SAW process parameters for any target values of weld bead geometries. Computational results show that GA method can be used effectively for solving complicated and highly non linear equations in prediction and optimization of welding process parameters.

EVALUATING THE DISCRETIZATION OF SEARCH SPACE IN CONTINUOUS PROBLEMS FOR ANT COLONY OPTIMIZATION

Mahdi Abachizadeh

Department of Mechanical Engineering
Ferdowsi University of Mashhad
Iran
m_abachizadeh@yahoo.com

Farhad Kolahan

Department of Mechanical Engineering
Ferdowsi University of Mashhad
Iran
kolahan@um.ac.ir

ABSTRACT

In this paper, after a beginning on the concept of ant algorithms, a brief survey of the ant-based methods proposed for optimization of problems with continuous design spaces is presented. As a common approach in continuous domains, discretizing the search space is the model presented to be appended to the original ant colony system (ACS) algorithm. Evaluating this method and comparing it to the standard simulated annealing shows that it is robust enough not to fall in local minima. However, when higher resolution is required, the algorithm fails to capture the global optimums and the computational costs rapidly increase. Therefore, it can be safely proposed for the problems in which a trade-off between time, solution accuracy and algorithm intricacy is needed.

KEYWORDS

Ant colony optimization, continuous optimization, discretization, simulated annealing, test functions.

NOMENCLATURE

F :	objective function (general concept)
$f(x)$:	objective function in test functions
i :	city at which the ant has just arrived
j :	candidate city the ant will visit
L_{gb} :	length of the globally best tour
m :	number of ants
n :	number of cities or variables
p :	destination city in a city pair

q :	randomly generated number in domain $[0,1]$
q_0 :	control parameter
r :	departure city in a city pair
S :	decision result based on state transition rule
s :	proportional probability
α :	pheromone decay parameter
β :	intensification factor of heuristic function
$\Delta\tau$:	added amount of pheromone
η :	heuristic function
ρ :	pheromone evaporation factor
τ :	pheromone function
τ_0 :	initial amount of pheromone

1. INTRODUCTION

Heuristic algorithms are methods which have been developed to find good but not necessarily optimal solutions in a reasonable amount of time. On the other side, combinatorial optimization problem such as scheduling, sequencing and time tabling are naturally NP-hard problems; i.e. it is not possible to find polynomial time algorithms to solve them optimally. Therefore during the last two decades, heuristic algorithms could improve the solutions obtained by exact mathematically-based methods.

Regarding the continuous domains, the application of heuristic methods has been always a challenging concept. The two available techniques are either partitioning the domain into a finite set of components or working with continuous variables. Enhanced versions of heuristics have been proposed

to optimize continuous problems. Examples include the Continuous Genetic Algorithm (CGA) (Chelouah, R., and Siarry, P., 2000), Enhanced Simulated Annealing (ESA) (Siarry, P., et al., 1997) or Enhanced Continuous Tabu Search (ECTS) (Chelouah, R., and Siarry, P., 1999). Four main versions are also suggested based on the Ant Colony Optimization (ACO) which will be comprehensively introduced in section 4. Against the discrete nature of their original versions, these algorithms employ continuous operators.

Generally, it is believed that the approach of discretization of continuous search spaces is only reliable when the initial range is not wide and the required resolution is not high. Here it is tried to examine this concept for the ACO algorithm. Also the qualitative impressions of being “wide” or “high” are evaluated by applying the proposed algorithm on some standard test functions. A standard SA code is also used for the same problems to provide the comparison of robustness and convergence quality. In addition, the obtained results are judged against the enhanced ACO methods to illustrate the advantages and drawbacks of discretization.

2. INTRODUCTION

Ant colony optimization is a nature-inspired method that was introduced in the early 1990's by Marco Dorigo for the solution of combinatorial optimization problems. This algorithm is based on the foraging behavior of real ants. Real ants are biologically blind; however, they are capable of finding the shortest path from a food source to their nest without using any visual cues. This amazing capability is founded on a simple fact. Ants secrete pheromone, a chemical substance, with a constant rate on the paths they march on and also are able to sense the intensity of pre-deposited pheromone in the environment. Based on their instinct, they prefer to follow the paths with higher amount of pheromone but this tendency is not deterministic. Therefore in general, if more ants march on a certain path, more pheromone will be accumulated and the path will be even more desirable. This behavior is shown in Figure 1.

The behavior is very simple and not even fully deterministic but effective enough as it is cooperative. All this behavior is simulated with a little difference in the ACO algorithms among which the Ant Colony System (ACS) is of our interest in this paper (Dorigo, M., and Gambardella, L.M., 1997) (It is preferred to call it with the general name

of ACO to the end of paper). The artificial ants employed in ACO are not fully blind, i.e. they have general information about the search space, have a memory of the length of the path they have explored and also live in an environment where time is discrete so the decisions are made in a step-by-step procedure. To understand the ACO algorithm, knowing the application of it on the TSP problem is essential as the algorithm have three distinct operators first defined and best described based on this problem.

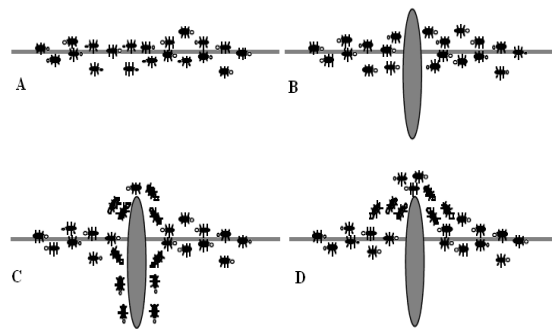


Figure 1 The behavior of ants when encountering an obstacle on their path

TSP problem is in fact a group of problems being one of the most distinguished challenges in the history of applied mathematics, making it a reliable benchmark for optimization methods. In this problem, there are n cities where finding the shortest tour including all cities being visited for just one time and ending in the first city is desired. The ACO algorithm employs m ants which are spread randomly on the cities. The ants start building their tour individually and come to the end of iterations altogether. Three basic rules called “state transition rule”, “global updating rule”, and “local updating rule” build the foundation of this algorithm.

2.1. State transition rule

Unlike other heuristic methods such as tabu search, genetic algorithm and simulated annealing where the coded solution candidate is built altogether and then evaluated, the ants construct the solution in a step-by-step procedure in the ACO. It means each ant should decide where to go for its next step by selecting among all unvisited candidate elements. The mechanism used in the ACO is a combination of directed greedy behavior and Rolette wheel known as state transition rule. The ant arrived at the city i chooses the next city among unvisited cities according to the following mechanism:

$$S = \begin{cases} \operatorname{argmax} \{ [\tau(i, j)] \cdot [\eta(i, j)]^\beta \} & q \leq q_0 \\ s & \text{otherwise} \end{cases} \quad (1)$$

$$s = \begin{cases} \frac{[\tau(i, j)] \cdot [\eta(i, j)]^\beta}{\sum_{u \in \text{allowed } u} [\tau(i, u)] \cdot [\eta(i, u)]^\beta} & \text{if } j \in \text{allowed} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where $\tau(i, j)$ is the amount of pheromone related to the path between cities i and j , and $\eta(i, j)$ is the heuristic function defined here as the inverse of distance between these two cities. The heuristic function is an operator in ACO which is defined according to the nature of the involved problem as it can be distance like in TSP problem or any other concepts such as cost or time. It has the rule of guiding and accelerating the convergence but is not vital to the concept of ACO. As can be seen, the total decision term is a combination of both pheromone and heuristic functions with the latter having a power of β .

The state transition rule consists of two sub-rules, while q and q_0 determine which one to be used. The constant parameter q_0 demonstrates the relative importance of sub-rules; however, q is a randomly generated number, uniformly distributed in domain $[0,1]$.

If there comes $q \leq q_0$ which is the case of exploitation, the city with the largest combination of pheromone and heuristic is chosen. Otherwise, the algorithm does not decide deterministically but only gives chances to the elements in proportion to their values as it is in a Rolette wheel; which means the city with the largest calculated term is not necessarily chosen. Thus, exploration of candidates with smaller function values is made feasible. In general, the ants act in a parallel manner. Their first elements of solution are assigned randomly and then to the end of constructing the solution, state transition rule is repeated.

2.2. Global updating rule

In ACO, the globally best ant which is the ant that has constructed the best solution from the beginning of the trial is allowed to deposit pheromone on its trail, even though no better tour is found in several consequent iterations. This rule which acts as positive feedback makes the search for the real best solution more directed. The rule is given by:

$$\tau(r, p) = (1 - \alpha) \cdot \tau(r, p) + \alpha \cdot \Delta\tau(r, p) \quad (3)$$

$$\Delta\tau(r, p) = \frac{1}{F} = (L_{gb})^{-1} \quad (4)$$

where α is a coefficient that acts to decrease the amount of pheromone inspired by evaporation in nature. The added amount of pheromone is obtained by inverting the objective function F which is the globally best tour length L_{gb} obtained from the beginning of the optimization.

2.3. Local updating rule

To avoid premature convergence and just like the natural phenomenon happening in nature due to evaporation, a local pheromone trail updating is performed on the value of pheromone related to the pair of cities just chosen by state transition rule:

$$\tau(r, p) = (1 - \rho) \cdot \tau(r, p) + \rho \cdot \Delta\tau(r, p) \quad (5)$$

$$\Delta\tau(r, p) = \tau_0 \quad (6)$$

It should be noted that the decay parameter ρ is chosen from domain $[0,1]$. Also in general, τ_0 which is the initial amount of pheromone, is calculated as the inverse of a rough estimate of the objective function multiplied by n , the problem dimension.

3. ACO FOR DISCRETE PROBLEMS

The TSP problem, on which the ACO was described, belongs to a group of problems in which the final solution is a string that includes all the elements of search space and therefore the design variables are not defined or even exist in the familiar way as is known in classical optimization. In other words, the algorithm is employed to put the search space elements in the proper *arrangement*.

In another class of problems, it is requested to *select* a proper value for design variable(s) from a finite set of available choices. In this type of problems, the solution construction mechanism faces no major difference in comparison to the described algorithm except for the definition of path and pheromone function. In analogy to the TSP problem, the allowable numerical values of discrete variables can be modeled as the zones of cities which no movement between the zones are permitted and only an ant can go from a zone to another zone in another city.

The graph appearing from the number of design variables and their corresponding allowable values builds the search space where the path and its subsequent pheromone function may be associated either with the set of its edges or the vertices of the graph.

4. ACO FOR CONTINUOUS DOMAIN

As briefly mentioned in the introduction, there are four important ant-related algorithms proposed in the literature. They include Continuous ACO (CACO) suggested in 1995 (Bilchev, G., and Parmee, I.C.), API (after *Pachycondyla APIcalis*) presented in 2000 (Monmarche, N., et al.), Continuous Interacting Ant Colony (CIAC) proposed in 2004 (Dreo, J., and Siarry, P.) and finally ACO_R introduced by Socha, K., and Dorigo, M. (2006).

One of the first attempts to apply an ant-related algorithm to the continuous optimization problems was CACO by Bilchev and Parmee. In CACO, the ants start from a point, called a nest, situated somewhere in the search space. The good solutions found are stored as a set of vectors, which originate in the nest. The ants at each iteration of the algorithm choose probabilistically one of the vectors. They then continue the search from the end-point of the chosen vector by making some random moves from there. The vectors are updated with the best results found.

There are important differences to the original ACO. They introduce the notion of nest, which does not exist in the ACO metaheuristic. Also, CACO does not perform an incremental construction of solutions, which is one of the main characteristics of the ACO metaheuristic.

Another ant-related approach to continuous optimization is the API algorithm by Monmarche et al. API does not claim to be based on the ACO metaheuristic. The ants perform their search independently, but starting from the same nest (the nest is moved periodically). The ants use only tandem running that is a type of recruitment strategy. This algorithm allows tackling both discrete and continuous optimization problems.

The third ant-based approach to continuous optimization is CIAC by Dreo and Siarry. CIAC uses two types of communication between ants: stigmergic information (spots of pheromone deposited in the search space) and direct communication between ants. The ants move through the search space being attracted by pheromone laid in

spots, and guided by some direct communication between individuals. Although also CIAC claims to draw its original inspiration from ACO, the differences are many as there is direct communication between ants and no incremental construction of solutions.

The final and best known of ACO-based methods is ACO_R. In this algorithm, the discrete probability distribution is replaced by a continuous one that is a probability distribution function.

The Gaussian functions as the most popular distribution is the one employed in this method. Since a single Gaussian function is not able to describe a situation where two disjoint areas of the search space are promising (as it only has one maximum), Gaussian kernel is the method proposed (Figure 2).

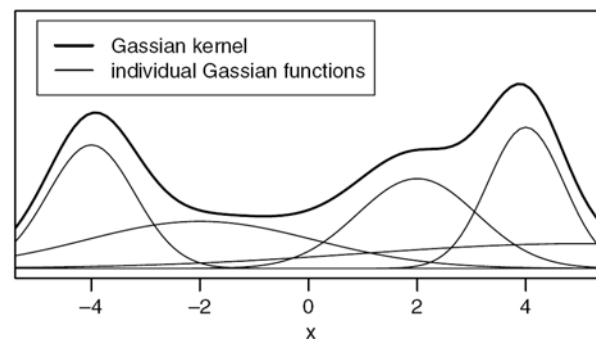


Figure 2 Example of five Gaussian functions and their superposition (Socha, K., & Dorigo, M., 2006)

As it can be concluded, all the proposed ant-based algorithms for tackling continuous problems have some enhanced operators which normally deal with continuous variables. Even the ACO_R that is the most straightforward way of extending ACO to continuous domains is basically intricate due to its additional actions and consequently is harder to implement.

Based on these explanations, it was tried to examine the idea of discretizing the search space. Thus, no modification to the concept of ACO described in section 3 is needed and consequently the algorithm is expected to be easier to employ. The numerical results and discussions followed by the implementation of this algorithm, hereafter called ACOD, are studied in the next section.

5. BENCHMARKING THE ACOD

As mentioned in the previous sections, the method that is subjected to examination here makes use of no enhanced operators. Therefore, it is decided to

compare it with another standard metaheuristic algorithm. The simulated annealing as a qualified method can be a reliable counterpart to the ACOD.

In continuous optimization, there has been a number of benchmarking functions for which the optimal solution is known *a priori*. These functions are generally multimimima and consequently challenging for optimization. A comprehensive description of the employed test functions can be found in [9].

In all fields of optimization, it is significantly important to select a proper criterion of comparison when two or more methods are engaged. In case of combinatorial optimization, usually each algorithm is given the same amount of CPU time and the results obtained within that time are compared. This makes the comparison of different algorithms complicated, as the CPU time depends on some issues such as the programming language, the compiler, the skills of the programmer, and finally on the machine used for running the experiments. Thus, in case of combinatorial optimization it is strongly recommended to re-implement all the algorithms used in the comparison in order to make it fair. This still does not guarantee an entirely fair comparison, as it is difficult to ensure that the same amount of effort is put into optimization of the code of all the implemented algorithms.

In contrast, the great majority of the papers on continuous optimization algorithms use the number of function evaluations needed to achieve a certain solution quality or the obtained algorithm performance after a predefined number of function evaluations. Such an approach gives several key advantages. It solves the problem of the algorithms being implemented using different programming languages; it is insensitive to the code-optimization skills of the programmer (or to the compiler used); and it allows comparing the results obtained on different machines. The drawback of this approach is that it does not take into consideration the time-complexity of the algorithms subjected to comparison. However, in view of the other numerous disadvantages of using the CPU time as a criterion, it is an acceptable methodology, and is adopted it in this paper.

Tables 1 and 2 display the settings used for ACOD and SA during the optimization procedure, respectively. The results obtained are also presented in Table 3. It is notable that the results are averaged after 5000 function evaluations and 10 independent runs.

Before bringing the results, it is noteworthy mentioning that defining the heuristic function usually brings a dimension of intricacy to the applying of ACO on most problems. That is because defining a measurable and meaningful concept during the solution, between any two elements of search space and meanwhile having a clear relevance to the objective function is generally very complex. Therefore, as done in this paper and also other reviewed methods, and without damaging the overall effectiveness of ACO, this function is neglected.

Table 1 ACOD parameter settings

m	α	ρ	τ_0	q_0
10	0.1	0.1	1e-6	0.5

Table 2 SA parameter settings

Cooling factor	Initial temperature
0.995	1000

Table 3 ACOD and SA results for three test functions

Function	Easom	Goldstein-Price	Ackley's path 10
Search domain	[-100,100]	[-2,2]	[-32,32]
Known optimum (x_1)	π	0	0
Known optimum (x_2)	π	-1	0
Known optimum $f(x)$	-1	3	0
x_1 by SA	3.2288	0.0188	0.3655
x_2 by SA	3.2917	-1.0244	0.1509
$f(x)$ by SA	-0.9427	3.4201	2.4061
x_1 by ACOD	3.1511	0.0027	0.0070
x_2 by ACOD	3.1471	-1.0006	0.0348
$f(x)$ by ACOD	-0.9999	3.0029	0.1604
Error by SA	0.0573	0.4291	2.4061
Error by ACOD	0.0001	0.0029	0.1604

Results clearly show that ACOD is the winner against the SA for all the three test functions. Both algorithms have run away from local minimums; however, ACOD could converge much closer to the global optimums rather than the SA.

6. ACOD AGAINST OTHER ANT-BASED METHOD

The specifically-enhanced ant-based algorithms for optimization of problems with continuous domain were described in section 5. They are definitely the first choices when an ant-based algorithm is to be selected for continuous problems though a question is still not answered:

Should the method of discretization be totally disregarded?

There is certainly no absolute conclusion. Reporting from the literature, Table 4 gives the averaged performance of these methods for some test functions after 100 independent runs. The numbers are the relative mean number of function evaluations to achieve the global optimum with the absolute and relative error of 10^{-4} . The numbers in parentheses are the actual values.

Table 4 Comparison of ACOD and other ant-based methods in finding the global optimum

Test function	Rosenbrock $-R_2$	Goldstein-Price
ACO _R	1.0 (820)	1.0 (384)
CACO	8.3	14
API	12	N/A
ACOD	-	19

For example for the R_2 function, the worst performance belongs to CIAC that has been able to find the global optimum in all its tryouts after the average number of 11480 function evaluations (not mentioned in Table 3). The case for ACOD is that after 20000 function evaluations, it has reached the solution resolution of 8×10^{-4} and there is even no guarantee for achieving the global optimum. That is due to the fact the feedback embedded in the ACOD fails to improve the solution and it is just the unbiased randomness of ACO that may result in progression.

The case for the Goldstein-Price function is different. The ACOD has been always able to capture the global optimum. Having a look on Table 4, its performance is comparable to CACO and much better than CIAC [8]. This can be interpreted due to the limited search domain. In addition, the variation in the amount of function is relatively small.

Based on the results presented in sections 5 and 6, the following conclusions are presented on the topic of discretization in continuous problem:

I. The size of the search domain is critically important. For a predefined resolution, by extending the search space, the computational costs drastically increase and the performance is no longer guaranteed.

II. The required resolution directly affects the performance of ACOD. For relatively smaller resolutions, the performance is promising and skipping the local minimums is generally achieved. When higher resolution is requested, the ant algorithms need to be enhanced with additional operators working with continuous variables. Insisting on discretization necessitates larger memory for keeping and evaluating the newly defined points.

III. Time is always a key point in evaluation of algorithms. When the continuous domain is discretized, the pheromone function should be defined for all the inserted points. Consequently, the next-step evaluation-selection procedure is inevitably repeated for all of them. In contrary, the ACO_R method for instance is equipped with a limited number of continuous distribution functions which cover all the search space. These algebraic functions can be easily calculated for any point of the search space. Therefore, the search for a point is replaced by a search for a continuous region. This way the algorithm needs much less CPU time.

IV. For real applications with continuous domains where the global optimum and smoothness of the objective function is not known, ACOD can be expected to converge to a relatively acceptable estimation of the global optimum. This estimation might be sufficient in some cases or at least direct other enhanced algorithms. However when achieving a solution with high resolution in a wide range is requested, employing other experienced methods is inevitable.

7. CONCLUSION

In the beginning, a standard version of Ant Colony Optimization (ACO) called Ant Colony System (ACS) has been described on the traveling salesperson problem. Afterward, the modifications needed to apply the same algorithm on the class of discrete problems with the selection (and not sequencing) have been introduced. Based on this foundation, the concept of applying ACO as a

basically discrete method on continuous domains has been explained. The enhanced ant-based algorithms for tackling continuous problems including CACO, API, CIAC and ACO_R were briefly reviewed. The ACOD method which is based on discretization of search domain has been later proposed and examined with the help of some test functions. Comparing the results with those obtained by applying the SA method clearly proves that ACOD has achieved reasonable outcomes and also outperformed SA in all cases. To challenge the proposed ACOD, cases with higher resolution, wider range and more intense variation in function values were employed. The results confirmed that this method fails to compete with other enhanced methods and therefore stating that it should be carefully employed.

REFERENCES

- Bilchev, G., Parmee, I. C., (1995), "The Ant Colony Metaphor for Searching Continuous Design Spaces" Proceedings of the AISB Workshop on Evolutionary Computation, Sheffield, pp. 25-39.
- Chelouah, R., Siarry, P., (1999), "Enhanced Continuous Tabu search", in *Meta-Heuristics Advances and Trends in Local Search Paradigms for Optimization*, ed. by Voss, S., Martello, S., Osman, I. H., Roucairol, C., Kluwer Academic Publishers, Boston, MA, pp. 49-61 (chapter 4).
- Chelouah, R., Siarry, P., (2000), "A Continuous Genetic Algorithm Designed for the Global Optimization of Multimodal Functions", *Journal of Heuristics*, Vol. 1, No. 6, pp. 191-213.
- Dorigo, M., Gambardella, L. M., (1997), "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem", *IEEE Transaction on Evolutionary Computation*, Vol. 1, No. 1, pp. 53-66.
- Dreo, J., Siarry, P., (2004), "Continuous Interacting Ant Colony Algorithm Based on Dense Hierarchy", *Future Generation Computer Systems*, Vol. 20, No. 5, pp. 841-856.
- Elbeltaghi, E., Hegazy, T., Grierson, D., (2005), "Comparison Among Five Evolutionary-based Optimization Algorithms", *Advanced Engineering Informatics*, Vol. 19, No. 1, pp. 43-53.
- Monmarche, N., Venturini, G., Slimane, M., (2000), "On How *Pachycondyla Apicalis* Ants Suggest a New Search Algorithm", *Future Generation Computer Systems*, Vol. 16, pp. 937-946.
- Siarry, P., Berthiau, G., Durbin, F., Haussy, J., (1997), "Enhanced Simulated Annealing for Globally Minimizing Functions of Many Continuous Variables" *ACM Transaction On Mathematical Software*, Vol. 23, No. 2, pp. 209-228.
- Socha, K., Dorigo, M., "Ant Colony Optimization for Continuous Domains" *European Journal of Operational Research*, In Press.