Performance of Aggregation Pheromone System on Unimodal and Multimodal Problems

Shigeyoshi Tsutsui
Dept. of Management Information Sciences, Hannan University
5-4-33 Amamihigashi, Matsubara
Osaka 580-8502, Japan
tsutsui@hannan-u.ac.jp

Martin Pelikan
Dept. of Math and Computer Science, University of Missouri at St. Louis, 8001 Natural Bridge Rd., St. Louis, MO 63121
pelikan@cs.umsl.edu

Ashish Ghosh
Machine Intelligence Unit, Indian Statistical Institute
203 B. T. Road, Kolkata 700 108, India
ash@isical.ac.in

Abstract- This paper describes and analyzes the aggregation pheromone system (APS) algorithm, which extends ant colony optimization (ACO) to continuous domains. APS uses the collective behavior of individuals that communicate using aggregation of pheromones. Two variants of APS are considered: the existing *generational* APS and the proposed *steady-state* APS. Both variants of APS are tested on several common unimodal and multimodal problems and their performance on these problems is analyzed with different parameter settings. The results indicate that using a steady-state evolutionary model improves the performance of APS on both unimodal as well as multimodal problems and that the performance of APS is relatively robust with respect to its parameter settings.

1 Introduction

As a bio-inspired computational paradigm, ant colony optimization (ACO) has been applied successfully to a large number of computationally hard problems. ACO simulates the collective behavior of ants, which communicate using pheromone trails. However, ACO has been mainly applied to discrete optimization problems such as the traveling salesman problem (TSP) [1-4], quadratic assignment [5], scheduling [6-7], vehicle routing [8], and routing in telecommunication networks [9].

As discussed in [32], many optimization algorithms have been originally developed for combinatorial or discrete oprimization and only afterwards adapted also to the continuous case. This is true for ACO. Although a direct application of the pheromone-trail metaphor to solving continuous optimization problems is difficult, some attempts were also made to use them for tackling continuous optimization problems. Up to now, only a few ant-colony based approaches for continuous optimization have been proposed in the literature. The first method called Continuous ACO (CACO) was proposed in [26]. CACO combines ACO with a real-coded GA which is similar to BLX-α [33] and the ACO approach was largely devoted to local search. CACO was extended in [27, 28]. In [29], the behavior of an ant called *Pachycondyla Apicalis* was introduced as the base metaphore of the algorithm (API) to solve continuous problems. The Continuous Interacting Ant Colony (CIAC) in [30] also introduces an additional direct communication scheme among ants.

In contrast to the above studies, studies that have a pure pheromone based method were proposed independently in [31, 32, 10, 11]. In [31], the pheromone intensity is represented by a single normal probability distribution function. The center of the normal distribution is updated to the place which has the best functional value at each iteration. Variance value is updated according to the current ant distribution. In [32], a mixture of normal kernels was introduced so that it can solve multimodal functions. However, both of the above two approaches use marginal distribution models (although they can be extended to multivariate ones) and the pheromone update rules used are quite differnt from those of the original ACO methods [2, 3].

In [10, 11], the aggregation pheromone system (APS) was introduced. APS replaces pheromone trails with *aggregation pheromones* (see Section 2.2.1) and uses it as the base metaphore of the algorithm. The pheromone update rule is applied in a way similar to that of ACO. As a result, aggregation pheromon density is eventually represented by a mixture of multivariate normal distributions, and APS could solve even hard problems which have a tight linkage among parameters.

This paper gives an elaborate study of the existing APS [10, 11], introduces a corresponding *steady-state* version, and compares the two using a set of common unimodal and multimodal continuous problems.

The remainder of this paper is organized as follows. Section 2 introduces APS and its variants. Section 3 presents experimental results. Finally, Section 4 concludes the paper.

2 Aggregation Pheromone Systems

This section starts with a brief introduction to ACO. The remainder of the section provides a detailed description of APS and its variants discussed in this paper.

2.1 A Brief Overview of ACO

In [2] an ACO algorithm called the *Ant System* (AS) is proposed for solving TSP. Given a TSP problem instance, AS works as follows. Each edge (i, j) between two cities i and j is assigned an initial pheromone intensity $\tau_{ij}(0)$. Each iteration updates the pheromone intensities on all edges by simulating a population of ants.

Each ant is first placed in a random city and then it traverses all the cities, laying a pheromone trail along the chosen tour. In each step of the traversal, the next city is chosen using a probability that is a function of the distance to the city and the amount of pheromone present on the direct path (edge) to the city. The probability of taking an edge decreases with the length of the edge and increases with the amount of pheromone on the edge.

Let $\tau_{ij}(t)$ be the trail intensity on edge (i, j) between cities i and j at iteration t. When all ants complete their traversals, the trail intensity on each edge (i, j) is updated according to the following formula:

$$\tau_{ij}(t+1) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij} \tag{1}$$

where ρ $(0 \le \rho < 1)$ is an evaporation coefficient, $\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$, and $\Delta\tau_{ij}^{k}$ are the amount of pheromone that is inversely proportional to the total length of the tour followed by the k-th ant so that larger values of $\Delta\tau_{ij}^{k}$ are given to the edges that belong to shorter tours. This process is iterated until termination criteria are met.

2.2 The Basic Model of the Aggregation Pheromone System

2.2.1 Appregation Pheromone

Pheromones that cause clumping or clustering behavior in a species, which bring individuals into a closer proximity, are referred to as aggregation pheromones [12]. Many functions of aggregation behavior have been observed in nature, including foraging-site marking and mating [13], finding a shelter, and defense. For example, cockroaches produce a specific pheromone with their excrement when they find a safe shelter and the pheromone attracts other members of their species [14]. Other members that are attracted to that shelter provide positive response in the form of the same pheromone, making the place even more attractive for other individuals.

Thus, aggregation pheromones cause individuals to aggregate around a *good position* which increasingly produces more pheromones to attract individuals of the same species. Since pheromones evaporate, aggregation at positions no longer eliciting a positive response slow down and, eventually, it stops completely.

The difference between ACO and APS is in how the pheromones function in the search space [11]. In ACO, pheromone intensity is assigned to each edge between two cities. On the other hand, in APS, the aggregation pheromone density is defined by a density function in the search space X in R^n. Each iteration consists of two basic steps:

(1) Update the pheromone density.
(2) Generate new individuals using the current pheromone density.

The following two subsections describe these two steps in more detail.

2.2.2 Updating the pheromone intensity

APS initializes the density of the aggregation pheromone to a constant so that new individuals are initially sampled according to a uniform distribution over the entire search space. In each iteration of APS, m individuals are generated based on the density of the aggregation pheromone. Individuals are attracted more strongly to the positions where the pheromone density is higher, whereas they are attracted less strongly to the positions where the density is lower. More specifically, APS generates new individuals using probabilities proportional to the values of the aggregation density function. Individuals increase the pheromone intensity in their neighborhood to increase the probability of generating more points in their neighborhood. To encourage the sampling of candidate solutions of higher quality, individuals with higher fitness emit more pheromone than those with lower fitness.

More formally, let $\tau(t,x)$ be the density function of the aggregation pheromone in iteration t. Initially (t=0), the aggregation pheromone is distributed uniformly, that is, $\tau(0,x) = c$, where c is an arbitrary positive constant. The probability density function used to generate new candidate solutions at iteration t, denoted by $p_\tau(t,x)$, is defined as

$$p_\tau(t,x) = \frac{\tau(t,x)}{\int_X \tau(t,x)dx}. \tag{2}$$

Each individual emits aggregation pheromone in its neighborhood. The intensity of aggregation pheromone emitted by individual x is based on the following properties:

(1) The intensity of the pheromone emitted by x should depend on its fitness $f(x)$ – the higher the fitness of x, the higher the pheromone intensity emitted by x. There are many approaches that can be used to control the pheromone intensity. In order to maintain a reasonable selection pressure, we use ranking based on fitness to control pheromone intensity. First, individuals are ordered according to their fitness. Each individual is then assigned its rank; the best individual has a rank m, whereas the worst individual has a rank 1. Where m is the population size.
(2) The intensity emitted by x should decrease with the distance from it so that the pheromone intensity of the points that are closer to x is increased more than the pheromone intensity of the points that are further from x.
(3) In order to alleviate the effects of linear transformations of the search space, the intensity should consider the overall distribution of individuals in the population. To achieve this, the pheromone intensity emitted by x is chosen to be a Gaussian distribution with the covariance equal to the covariance of the entire population.

More formally, let us consider the individual $x_{t,r}$ with rank r at time t. The density of the pheromone intensity emitted by $x_{t,r}$ at point x is given by a scaled normal distribution

$$\Delta\tau'(t,r,x_{t,r},x) = \frac{C}{\sum_{k=1}^{m} k^{\alpha}} r^{\alpha} N(x; x_{t,r}, \beta^2\Sigma_t), \qquad (3)$$

and the total aggregation pheromone density emitted by the entire population of m individuals at iteration t is then given by

$$\Delta\tau(t,x) = \sum_{r=1}^{m} \Delta\tau'(t,r,x_{t,r},x). \qquad (4)$$

Here, $N(x; x_{t,r}, \beta^2\Sigma_t)$ is a multivariate normal distribution function, Σ_t is the covariance of the population of all individuals at time t, α (α >0) is a parameter to adjust the relative importance of rank, β (β>0) is the scaling factor, and C is the total pheromone intensity emitted by all individuals at time t, that is,

$$\int_X \Delta\tau(t,x)dx = C. \qquad (5)$$

In this study, we assume that C is a constant for all t and

$$\int_X \Delta\tau(0,x)dx = \int_X c dx = C \ . \qquad (6)$$

The sampling method in subsection 2.2.3 is based on the assumptions of Eqs. 5 and 6. The scaling factor β controls exploration by reducing the amount of variation as the population converges to the optimum. Other approaches to scaling can be used and one of the interesting directions of future research is to introduce adaptive scaling similarly as in evolutionary strategies with adaptive mutation strength [22, 23].

The total aggregation pheromone density is updated according to the following formula:

$$\tau(t+1,x) = \rho \cdot \tau(t,x) + \Delta\tau(t,x), \qquad (7)$$

where ρ $(0 \le \rho < 1)$ controls the evaporation rate. The higher the value of ρ, the smaller the effect of the pheromone density emitted in the current iteration on the pheromone intensity used in the next iteration.

After the pheromone updating is performed, new individuals are generated based on the new pheromone density and the next iteration of APS is performed. Since the pheromone density updates increase pheromone intensity near individuals with higher fitness, the pheromone density in promising regions of the search space is expected to increase over time, eventually converging to global optima.

2.2.3 Sampling New Individuals

In this subsection, we discuss how to sample new individuals from the aggregation pheromone density function $\tau(t+1,x)$ obtained in Eq. 7. As described in subsection 2.2.2, new individuals are sampled with probabilities proportional to the aggregation density $\tau(t+1,x)$. To perform the sampling, we need to obtain probability density function $p_\tau(t+1,x)$ from $\tau(t+1,x)$. The term $\tau(t+1,x)$ in Eq. 7 can be rewritten as

$$\tau(t+1,x) = \rho^{t+1}\tau(0,x) + \sum_{h=0}^{t} \rho^h \Delta\tau(t-h,x). \qquad (8)$$

From equations 2, 5, 6, and 8, $p_\tau(t+1,x)$ is obtained as

$$p_\tau(t+1,x) = \frac{\rho^{t+1}}{\sum_{k=0}^{t+1}\rho^k} \cdot \frac{\tau(0,x)}{C} + \sum_{h=0}^{t} \frac{\rho^h}{\sum_{k=0}^{t+1}\rho^k} \cdot \frac{\Delta\tau(t-h,x)}{C}. \qquad (9)$$

If a probability density function $f(x)$ can be written as a mixture of other probability density functions

$$f(x) = p_1 f_1(x) + p_2 f_2(x) + \cdots + p_S f_S(x), \qquad (10)$$

with $\sum_{i=1}^{S} p_i = 1$, then the sampling of each point according to the distribution defined by $f(x)$ proceeds as follows:

(1) Choose which mixture component $f_i(x)$ should be used where a mixture component $f_i(x)$ is chosen with probability p_i.
(2) Generate a random point according to the density function of the chosen component

Note that $p_\tau(t+1,x)$ is indeed a mixture distribution of t multivariate Gaussian distributions and one uniform distribution. Therefore, we can sample it using the standard sampling procedure for mixture distributions where the probability of the i-th component of the mixture is

$$p_i = \rho^i \Big/ \sum_{k=0}^{t+1}\rho^k \qquad (11)$$

and the i-th mixture component is given by

$$f_i(x) = \begin{cases} \Delta\tau(t-i,x)/C & \text{if } i \le t \\ \tau(0,x)/C & \text{otherwise} \end{cases} \qquad (12)$$

This sampling is called *cycle sampling*.

Sampling according to the last mixture component $f_{t+1}(x)$ is simple because $f_{t+1}(x)$ is a constant and it thus represents a uniform distribution over the entire search space. The remaining components $f_i(x)$, where $i \le t$, are mixture distributions of m components each:

$$\frac{\Delta\tau(t-i,x)}{C} = \sum_{r=1}^{m} \frac{r^\alpha}{\sum_{k=1}^{m} k^\alpha} N(x; x_{t-i,r}, \beta^2\Sigma_{t-i}), \qquad (13)$$

Sampling according to $f_i(x)$, where $i \le t$, can be done analogically, i.e., sampling procedure for mixture distributions where the probability of the r-th component $N(x; x_{t-i,r}, \beta^2\Sigma_{t-i})$ of Eq. 13 is

$$p_r = r^\alpha \Big/ \sum_{k=1}^{m} k^\alpha \qquad (14)$$

We call this sampling the *rank sampling*. Since each component is a normal distribution, it can be sampled using Cholesky decomposition [15].

To perform the sampling based on Eq. 9, using the above sampling method, we need a large amount of memory to store $x_{t-h,r}$ vector values and covariance matrix $\beta^2\Sigma_{t-h}$ when cycle t becomes large. In this case $\rho^h \to 0$ for large h since ρ<1. Thus, we can limit the maximum number of cycles to keep data up to a constant H. Then, Eq. 9 for $t \ge H$ can be represented as

$$p_\tau(t+1,x) = \sum_{h=0}^{H-1} \frac{\rho^h}{\sum_{k=0}^{H-1}\rho^k} \cdot \frac{\Delta\tau(t-h,x)}{C}. \tag{15}$$

The computational complexity of these three samplings to generate an individual is as flollows: First, the cycle sampling is simply performed with O(t) for $t < H$ or O(H) for $t \geq H$. Next, the rank sampling is also simply performed with O(m). Finally, Cholesky decomposition [15] is performed with O(n^3). Thus the computation time of the sampling is mainly occupied by Cholesky decomposition. However, since a function evaluation usually needs a much larger computational time compared to the evolutionary algorithm runtime in real-world parameter optimization, the number of function evaluations is the more critical issue. In Section 3, we mainly evaluate the APS with the number of function evaluations.

Although at each APS cycle, sampling is performed according to Eqs. 9 or 15 probabilistically, we introduce a perturbation, resulting from conflict among individuals, or environmental disturbances. Perturbation works to perform the same function as mutation in evolutionary algorithms. The perturbation rate is represented by P_{rate}.

2.3 APS Algorithm Description

The pheromone density is initialized uniformly to a constant. The initial pheromone density is used to generate the first population of m individuals; therefore, the initial population is generated randomly with a uniform distribution over all possible individuals. Next, the population is evaluated and the individuals are ranked according to their fitness. Then, the covariance matrix is computed for the current population and the pheromone density is updated based on the current population and individual rank.

We use two approaches for generating new individuals and incorporating them into the original population:

Generational approach (APS/G): Here m individuals are first generated based on the current pheromone density. Then, the best $e \times m$ individuals, where e<1, from the previous population are added to the newly generated solutions. The best m individuals are then selected from the combined population of $(e+1)\times m$ to create the next population of m individuals. The pseudo-code of APS/G is shown in Figure 1.

Steady-state approach (APS/S): Here only $e \times m$ individuals are generated according to the current pheromone density where e<1. The new $e \times m$ individuals replace the worst $e \times m$ individuals in the original population, forming the new population of m solutions. The pseudo-code of APS/S is shown in Figure 2. The difference between Figure 1 and 2 is only in the method to generate new population $P(t+1)$ from the previous population $P(t)$ and in Steps 6, 7, and 9 of the figures.

In both approaches, the value of e should be rather small, e.g. e = 0.1. The next section presents experimental results of both APS variants on a number of continuous optimization problems. The two variants are compared and the results are discussed in the context of other methods for solving similar problems.

```
1.  Set APS cycle $t \leftarrow 0$
2.  Set the initial pheromone density $\tau(0, x)$ uniformly and sample initial
    population $P(t)$ randomly
3.  Evaluate $P(t)$ and give the rank number $r$ for each individual ($m$ for
    the best individual, 1 for the worst individual)
4.  Compute the covariance matrix $\Sigma_t$ for $P(t)$
5.  Update the pheromone density $\tau(t+1, x)$ according to Eq. 7
6.  Get best $e \times m$ individuals of $P(t)$ and rename as $E(t)$ for next cycle.
7.  Sample $m$ new individuals $N(t)$ according to Eq. 9 for $t<H$ and
    according to Eq. 14 for $t \geq H$ The sampling consists of 4 steps:
    (1) Cycle sampling
    (2) Rank sampling
    (3) Normal distribution sampling with Cholesky decomposition
    (4) Perturbation with rate $P_{\text{rate}}$
8.  Evaluate $N(t)$
9.  The best $m$ individuals are then selected from the combined
    population of $\{N(t)+E(t)\}$ to create the next population of $m$
    individuals $P(t+1)$
10. $t \leftarrow t+1$
11. If the termination criteria is met, terminate the algorithm. Otherwise,
    go to 4
```

Figure 1: Pseudo-code of APS/G

```
1.  Set APS cycle $t \leftarrow 0$
2.  Set the initial pheromone density $\tau(0, x)$ uniformly and sample initial
    population $P(t)$ randomly
3.  Evaluate $P(t)$ and give the rank number I$r$ for each individual ($m$ for
    the best individual, 1 for the worst individual)
4.  Compute the covariance matrix $\Sigma_t$ for $P(t)$
5.  Update the pheromone density $\tau(t+1, x)$ according to Eq. 7
6.  Get best $(1-e)\times m$ individuals of $P(t)$ and rename as $E(t)$ for next
    cycle.
7.  Sample $e \times m$ $m$ new individuals $N(t)$ according to Eq. 9 for $t<H$ and
    according to Eq. 14 for $t \geq H$ The sampling consists of 4 steps:
    (1) Cycle sampling
    (2) Rank sampling
    (3) Normal distribution sampling with Cholesky decomposition
    (4) Perturbation with rate $P_{\text{rate}}$
8.  Evaluate $N(t)$
9.  Obtain the next population of $m$ individuals $P(t+1)$ by combining the
    populations $\{N(t)+E(t)\}$
10. $t \leftarrow t+1$
11. If the termination criteria is met, terminate the algorithm. Otherwise,
    go to 4
```

Figure 2: Pseudo-code of APS/S

3 Experiments

3.1 Test Problems

The two variants (generational and steady-state) of APS are tested on five test functions: the Ellipsoidal function ($F_{\text{Ellipsoidal}}$), the Ridge function (F_{Ridge}), the Rosenbrock function ($F_{\text{Rosenbrock}}$), the Rastrigin function ($F_{\text{Rastrigin}}$), and the Schaffer function (F_{Schaffer}). F_{Ridge} has weak linkage among variables. $F_{\text{Rosenbrock}}$ has strong linkage among variables. $F_{\text{Ellipsoidal}}$ has no linkage among variables.

$$F_{\text{Ellipsoidal}} = \sum_{i=1}^{n} i x_i^2, \quad (-3.12 \leq x_i < 7.12) \tag{16}$$

$$F_{\text{Ridge}} = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2, \; (-44 \leq x_i < 84) \tag{17}$$

$$F_{\text{Rosenbrock}} = \sum_{i=2}^{n} (100(x_1 - x_i^2)^2 + (x_i - 1)^2), \quad (-2.048 \leq x_i < 2.048) \tag{18}$$

$$F_{\text{Rastrigin}} = 10n + \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i)\right), \; (-3.12 \leq x_i < 7) \tag{19}$$

$$F_{\text{Schaffer}} = \sum_{i=1}^{n-1}\left(x_i^2 + x_{i+1}^2\right)^{0.25}\left[\sin^2\left(50\left(x_i^2 + x_{i+1}^2\right)^{0.1}\right)\right], \qquad (20)$$
$$(-20 \le x_i < 30)$$

3.2 Description of Experiments

There are five parameters for both versions of APS: m, e, α, β, ρ. In all experiments, the same population size of m=100 was used. To use similar selection pressure in all experiments, we set e=0.1 in all experiments. Perturbation was applied to each solution with P_{rate} = 0.0005 by adding a randomly generated number from a zero-mean, unit-normal distribution. Two types of experiments were performed:

1) *Experiments with fixed parameters.*

   Here the remaining parameters were set as follows. For the generational APS, the following parameters were used in experiments: $\beta = 0.7$, $\alpha = 4$, $\rho = 0.8$.

   For the steady-state APS, the following parameters were used for $F_{Ellipsoidal}$, $F_{\text{Rastrigin}}$, and $F_{Schaffer}$: $\beta = 0.7$, $\alpha = 6$, $\rho = 0.2$. For the remaining problems, the steady-state APS used $\beta = 1.0$.

2) *Experiments for analyzing the sensitivity of APS to the values of its parameters.*

   In addition to presenting the performance results with the above settings, we analyzed the sensitivity of APS to changing the parameters α, β, ρ. To make the analysis feasible, only the sensitivity of APS with respect to one parameter is analyzed at a time, while all the remaining parameters are kept constant.

We evaluated APS by measuring #OPT (the number of runs in which APS succeeded in finding the global optimum) and MNE (the mean number of function evaluations to find the global optimum in those runs where it did find the optimum). We assumed the solution to be successfully detected if the functional value is within a variation of $n10^{-6}$ from the actual optimum value.

Problem size n = 20 is used for all test functions. 20 runs were performed in each setting. For unimodal problems, each run continued until the global optimum is found or a maximum of 500,000 evaluations is reached; for multimodal problems, the runs are terminated after 2 million evaluations. For a comparison with a GA, the results obtained with the steady state and generational APS are also compared to the results obtained with SPX crossover [16]. To keep the similar condition with APS, for SPX mutation was applied to each solution with the probability of 0.0005 by adding a randomly generated number from a zero-mean, unit normal distribution.

3.3 Results

3.3.1 Results with fixed parameters

Table 1 shows the results of APS using the parameter values described in Section 3.2. The results are compared with the results of SPX, which represents another crossover operator in real-coded GAs [16].

The results clearly indicate that for all test problems, the average performance of the steady-state APS is better than that of the generational APS. Both variants outperform SPX in most cases, although on $F_{\text{Rastrigin}}$ and F_{schaffer}, the generational APS is outperformed by SPX. The steady-state APS provides the best performance for all test problems. The edge that the steady-state model has over the generational model was more prominent for the multimodal cases ($F_{\text{Rastrigin}}$ amd F_{Shaffer}). This is possibly due to the fact that in steady-state GAs, only a few parent individuals are deleted and thus some promising solutions are not lost.

Table 1 Results of APS with default values

Function	Name		$F_{\text{Ellipsoidal}}$	F_{Ridge}	$F_{\text{Rosenbrock}}$	$F_{\text{Rastrigin}}$	F_{Schaffer}
	Linkage		No	Medium	Strong	No	Weak
	Multimodal		No	No	No	Yes	Yes
Evolutionary Algorithm	APS/G	#OPT	20/20	20/20	20/20	17/20	8/20
		MNE	59955.0	71420.0	97640.0	485168.8	594572.5
		STD	1534.1	1726.8	5161.8	127989.4	484500.6
	APS/S	#OPT	20/20	20/20	20/20	20/20	20/20
		MNE	24933.0	54584.5	74412.0	240759.5	207143.5
		STD	3341.2	7273.5	36374.5	79902.3	40154.2
	SPX	#OPT	20/20	20/20	20/20	17/20	20/20
		MNE	113274.8	138175.5	255639.0	406927.9	453569.3
		STD	1697.1	1362.5	21419.4	42627.2	2090.6

3.3.2 Sensitivity of APS to the values of its parameters

In this section we analyze the performance of both variants of APS for varying parameters.

Results with variation of evaporation rate ρ:

Figures 3(a), 3(b) and 3(c) represent the variation of MNE and #OPT with ρ for the function $F_{\text{Ellipsoidal}}$ (unimodal, no epitasis) $F_{\text{Rosenbrok}}$ (unimodal, strong epitasis) and F_{Schaffer} (multimodal, weak epitasis). The first figure shows the variation with generational model and the second one for the steady state model.

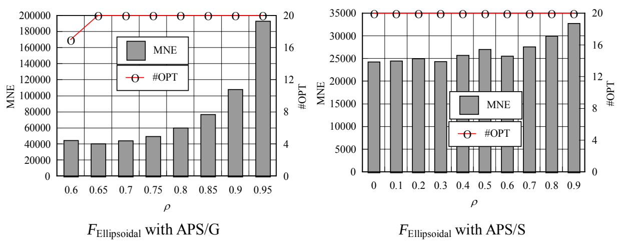

Figure 3(a): Variation of MNE & #OPT with ρ for $F_{\text{Ellipsoid}}$

Looking at Figure 3(a), one can easily infer that although #OPT is more or less the same for both the models, MNE, on the average, is much smaller for the steady state model (25000-33000) compared to the generational model (40000-200000). Also MNE does not vary much over ρ. From Figure 3(b) one can say that MNE varies a lot for the generational model (100000-400000) compared to the steady state model (50000-150000), also #OPT is less for the generational model for some values of ρ and #OPT is more for some other values of ρ. Thus for the strong epitasis case, #OPT is much sensitive on ρ for the generational model. For the multi-modal case (Figure

3(c)) MNE is quite low and #OPT is very high for almost all values of ρ for the steady state model. Thus we can say, from the results, that the steady state model is more robust with the variation of the evaporation coefficient ρ than APS/G. In steady state model, since all the parent individuals are not deleted at a certain point of time, it shows a similar effect of *no evaporation* and is thus not much sensitive on ρ.

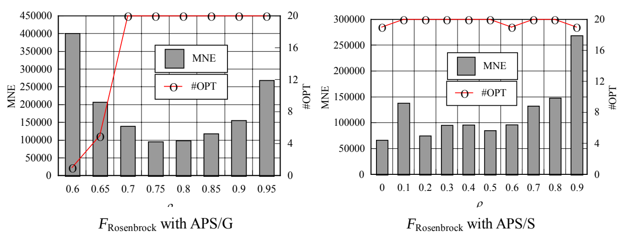

Figure 3(b): Variation of MNE & #OPT with ρ for $F_{\text{Rosenbrock}}$

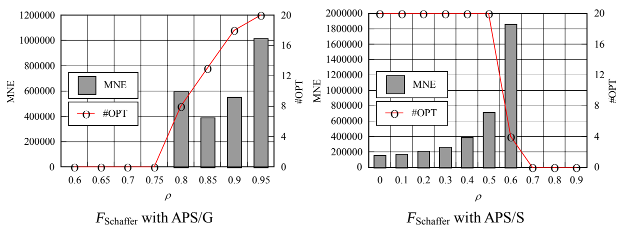

Figure 3(c): Variation of MNE & #OPT with ρ for F_{Schaffer}

Results with variation of α:

Here α is parameter to adjust the relative importance of rank. From Figure 4(a), we notice that although #OPT is the same for both the models, MNE, on the average, is smaller (varies from 22000 to 33000) for the steady state model compared to the generational model (whose minimum is 40000). Also MNE is more or less invariant over α.

From Figure 4(b) we notice that MNE is smaller for the steady state model (75000 to 115000) compared to the generational model (100000-200000), #OPT falls drastically for large value of α for the generational model; but it maintains 100% accuracy in the steady state case. Thus for the strong epitasis case, #OPT is not sensitive at all and MNE is less sensitive on α for the steady state model.

For the multi-modal case (Figure 4(c)) MNE is quite low (200000-300000) and #OPT is very high for all values of α for the steady state model; whereas MNE is quite high (600000-1000000) for the generational model and #OPT falls to a minimum of 0 very fast. Thus we can say from the results that the steady state model is much more robust with the variation of α.

Results with variation of scaling factor β:

Figure 5(a) shows that although #OPT is more or less the same for both the models (with a few exceptions), MNE, on the average, is smaller (varies from 22000 to 39000, with an exception) for the steady state model compared to the generational model (minimum is 42000). From Figure 5(b) it is hard to decide which model has an edge on the other. But for the multi-modal case (Figure 5(c)) MNE is quite low (200000-500000, with just one exception) and #OPT is very high for almost all values of β for the steady state model; whereas MNE is quite high (500000-1500000) for the generational model and #OPT is most always small for different values of β.

Thus we can say from the results that both the models are sensitive to variation of β, although the steady state model is less susceptible.

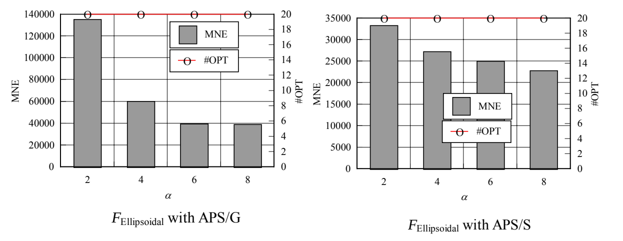

Figure 4(a): Variation of MNE & #OPT with α for $F_{\text{Ellipsoid}}$

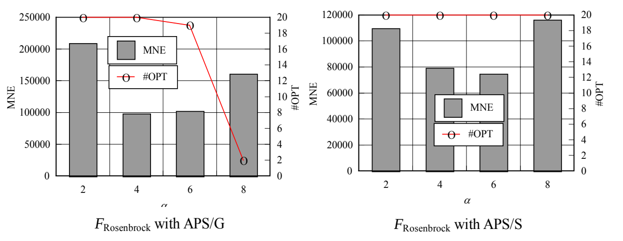

Figure 4(b): Variation of MNE & #OPT with α for $F_{\text{Rosenbrock}}$

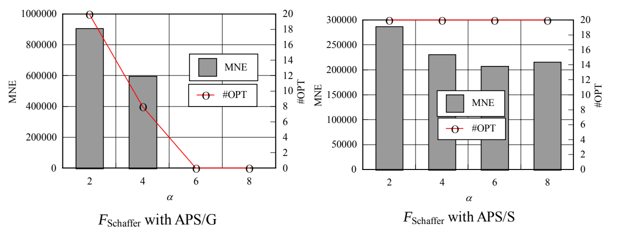

Figure 4(c): Variation of MNE & #OPT with α for F_{Schaffer}

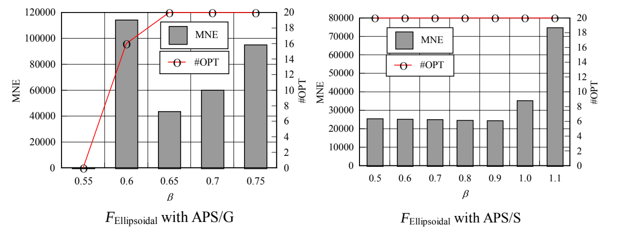

Figure 5(a): Variation of MNE & #OPT with β for $F_{\text{Ellipsoidal}}$

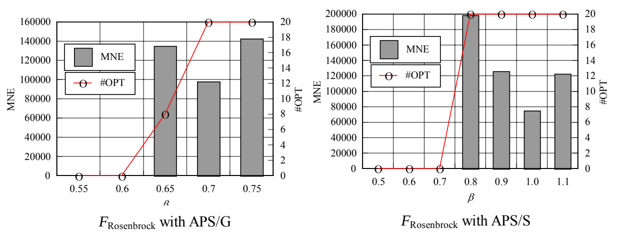

Figure 5(b): Variation of MNE & #OPT with β for $F_{\text{Rosenbrock}}$

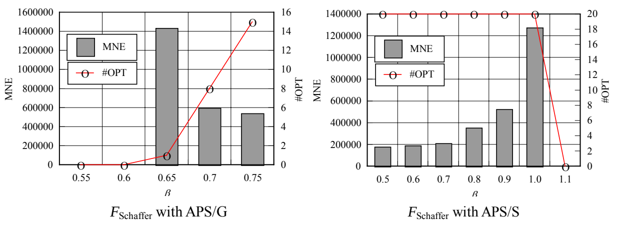

Figure 5(c): Variation of MNE & #OPT with β for F_{Schaffer}

4 Conclusions and future work

In this paper, we have described the *aggregation pheromone system (APS)*, which uses *aggregation pheromones* for solving real-valued optimization problems. We analyzed the performance of two variants of APS, the generational APS (APS/G) and the steady-state APS (APS/S), on several standard unimodal and multimodal problems. We also analyzed the sensitivity of APS to its parameters.

The results indicate that the steady state APS outperforms the generational APS as well as SPX. Furthermore, it is shown that the steady state APS is not very sensitive to the evaporation rate and the ranking factor, although it is still relatively sensitive to the shrinking factor. Nonetheless, the results obtained with the steady-state APS are promising and they indicate that APS represents an interesting area for future research in optimization of real-valued problems with evolutionary algorithms.

We do not claim that APS is the most efficient algorithm for solving the above test problems. However, it provides an alternative approach to using probabilistic model building and sampling in evolutionary algorithms [24, 25] for solving real-valued problems, which should be applicable to both unimodal and multimodal problems. There are many topics for future work in this area, many of which are possible only because APS uses an explicit probabilistic model to model and sample candidate solutions. One of the most interesting directions for future research is to use APS as the basis for building multivariate models of the population that can identify and exploit problem decomposition.

Although the results obtained with APS on standard test problems are promising, there are many important topics for future research in this area. First of all, it is important to identify classes of problems for which APS outperforms advanced evolutionary algorithms in the continuous domains, for example, the covariance matrix analysis evolution strategy (CMA-ES) [18]. Preliminary experiments show that for unimodal problems, APS is usually outperformed by CMA-ES, although on some multimodal problems, APS clearly outperforms CMA-ES. Since the model used in APS to sample new individuals can contain multiple attractors, it can be hypothesized that multimodal problems are indeed the most promising area of application for APS. Furthermore, this also indicates that in combination with niching, APS can be used to discover multiple optima of multimodal problems; this can often be beneficial in real-world applications.

Another interesting area for future research is to extend APS to deal with multiobjective problems in the continuous domain. For example, APS can be combined with the nondominated sorting selection GAs of NSGA-II [19].

Additionally, techniques can be designed to adaptively control APS parameters to maximize APS performance without having to manually tune its parameters. The most important candidates for adaptive parameter tuning are the evaporation rate, the shrinking factor, and the mutation strength.

Finally, although APS is capable of exploring multiple attractors simultaneously, APS is not capable of exploiting problem decomposition, which is an inherent feature of many difficult real-world problems. However, since APS uses a probabilistic model to model and sample candidate solutions, it is possible to incorporate machine learning methods to identify and exploit appropriate problem decomposition and deal with each sub-problem independently or quasi independently. APS might provide an interesting alternative to using standard mixtures of normal distributions for this purpose as is done in [20-21].

Acknowledgments

The authors greatfully acknowledge Dr. Marco Dorigo for his valuable comments on this paper. This research is partially supported by the Ministry of Education, Culture, Sports, Science and Technology of Japan under Grant-in-Aid for Scientific Research number 16500143. A part of this work was done when Ashish Ghosh visited Hannan University, Osaka with a fellowship from Japan Society for Promotion of Science. Pelikan was also partially supported by the Research Award and the Research Board at the University of Missouri.

5 Bibliography

[1] Bullnheimer, B., Hartl, R. F., and Strauss, C.: A New Rank Based Version of the Ant System: A Computational Study, *Central European Journal for Operations Research and Economics*, 7(1):25-38, 1999.

[2] Dorigo M., Maniezzo, V., and Colorni, A.: The Ant System: Optimization by a Colony of Cooperating Agents, *IEEE Trans. on SMC-Part B*, 26(1):29-41, 1996.

[3] Dorigo M. & L.M. Gambardella : Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE TEC*, 1(1):53-66, 1997.

[4] Stützle, T. and H. Hoos, H.: The MAX-MIN Ant System and local Search for Combinatorial Optimization Problems: Towards Adaptive Tools for Global Optimization, *Proc. of the 2nd Metaheuristics Int. Conf. (MIC-97)*, 1997.

[5] Maniezzo, V.: Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem, *CSR 98-1*, Scienze dell'Informazione, Universitá di Bologna, 1998.

[6] Costa, D. and Hertz, A.: Ants Can Colour Graphs. *Journal of the Operational Research Society*, 48:295-305, 1997.

[7] Forsyth P. and Wren, A.: An Ant System for Bus Driver Scheduling. Proc. of the 7th Int. Workshop on Computer-Aided Scheduling of Public Transport, 1997.

[8] Bullnheimer, B., Hartl, R. F., and Strauss, C.: Applying the Ant System to the Vehicle Routing Problem, *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Voss, S., et al (Eds.), Kluwer, 1999.

[9] Schoonderwoerd, R., Holland, O., Bruten, J., and L. Rothkrantz, L.: Ant-based Load Balancing in Telecommunications Networks. *Adaptive Behavior*, 5(2):169-207, 1997.

[10] Tsutsui, S. and Ghosh, A.: An Extension of Ant Colony Optimization for Function Optimization *Proceedings of the 5th Asia-Pacific Conference on Simulated Evolution And Learning (SEAL04)*, Pusan, Korea, 2004.

[11] Tsutsui, S.: Ant Colony Optimisation for Continuous Domains with Aggregation Pheromones Metaphor: *Proceedings of the The 5th International Conference on Recent Advances in Soft Computing (RASC-04)*, pp. 207-212, 2004.

[12] Lorenzo Figueiras, A.N. and Lazzari, C. R.: Aggregation behavior and interspecific responses in three species of Triatomine, *Memórias do Instituto Oswaldo Cruz* . 93(1):133-137, 1998.

[13] Bell, W. J.: Chemo-orientation in Walking Insects, *Chemical Ecology of Insects*, Bell, W. J. and Carde, R. T. Eds, 93-109, 1984.

[14] Sakuma, M. and Fukami, H.: Aggregation arrestant pheromone of the German cockroach, Blattella germanica (L.) (Dictyoptera: Blattellidae): isolation and structure elucidation of blasttellastanoside-A and -B, *Journal of Chem. Ecol*. 19:2521-2541, 1993.

[15] Schatzman, M., Taylor, J., and Schatzman, M.: *Numerical Analysis: A Mathematical Introduction*, Oxford Univ Press, 2002.

[16] Higuchi, T., Tsutsui, S., and Yamamura, M.:Theoretical analysis of simplex crossover for real-coded Genetic Algorithms, *Proc. of the PPSN VI*, 365-374, 2000.

[17] Ono, I. and Kobayashi, S.: A Real-Coded Genetic Algorithm for Function Optimization Using the Unimodal Normal Distribution Crossover, *Proc. of the 7th ICGA*, 246-253, 1997.

[18] Hansen, N. and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pp. 312-317, 1996.

[19] Deb, K., Agrawal, S., Pratap, A., Meyarivan, T. A Fast Elitist Non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. *Proceedings of the Parallel Problem Solving from Nature VI Conference* , 16-20 September. (Paris, France), pp. 849-858, 2000.

[20] Ahn, C.W., Ramakrishna, R.S., and Goldberg, D.E. Real-coded Bayesian Optimization Algorithm: Bringing the Strength of BOA into the continuous World. *Proc. of the Genetic and Evolutionary Computation Conference 2004 (GECCO-2004)*, 840-851, 2004.

[21] Ocenasek, J., Schwarz, J.: Estimation of Distribution Algorithm for mixed continuous-discrete optimization problems. *Proc. of the 2nd Euro-International Symposium on Computational Intelligence*, IOS Press, Kosice, Slovakia, 227-232, 2002.

[22]Schwefel, H. P.: Adaptive Mechanismen in der biologischen Evolution und ihr Einfluss auf die Evolutionsgeschwindigkeit, *Technical Report Re 215/3*, 1974.

[23] Rechenberg, I: *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, Frommann-Holzboog, 1973.

[24] Larranaga, P. and Lozano, J. A. editors: *Estimation of Distribution Algorithms*: A New Tool for Evolutionary Computation. Kluwer, Boston, MA, 2002.

[25] Pelikan, M., Goldberg, D. E., and Lobo, F.: A survey of optimization by building and using probabilistic models. *Computational Optimization and Applications*, 21(1):5-20, 2002.

[26] Bilchev, G. and Parmee,I. C.: The Ant Colony Metaphor for Searching Continuous Design Spaces. Proc. of the AISB Workshop on Evolutionary Computation, T. Fogarty (Eds.), Lecture Notes in Computer Science. 993, Springer Verlag, 24-39, 1995.

[27] Mathur, M. and Karle, S.B. and Priye, S., Jyaraman, V.K. and Kulkarni, B. D.: Ant Colony Approach to Continuous Function Optimization. *Ind. Eng. Chem. Res*, 39:3814-3822, 2000.

[28] Wodrich, M. and Bilchev G.: Cooperative distribution search: the ant't way, *Control Cybernetics*, 3:413-446, 1997.

[29] Monmarch'e, N. and Venturini, G. and Slimane, M.: On how *Pachycondyla apicalis* ants suggest a new search algorithm, *Future Generation Computer Systems*, 16(8):937-946, 2000.

[30] Dr'eo, J. and Siarry, P.: A New Ant Colony Algorithm Using the Heterarchical Concept aimed at optimization of multiminima continuous functions, *Proc. of the Third International Workshop on Ant Algorithms (ANTS 2002)*, Dorigo, M. and Di Caro, G. and Sampels, M. (Eds.). Lecture Notes in Computer Science 2463 Springer Verlag, 216-221, 2002.

[31] Seid H. Pourtakdoust and Hadi Nobahari (2004). An Extension of Ant Colony System to Continuous Optimization Problems. *Proc. of Fourth International Workshop on Ant Colony Optimization and Swarm Intelligence (ANTS 2004)*, Dorigo, M. and Birattari, M. and Blum, C. and Gambardella, L.M. and Mondada, F. and Stuetzle, T. (Eds.). Lecture Notes in Computer Science 3172 Springer Verlag. 294-301, 2004.

[32] Socha, K.: ACO for Continuous and Mixed-Variable Optimization, *Proc. of Fourth International Workshop on Ant Colony Optimization and Swarm Intelligence (ANTS 2004)*, Dorigo, M. and Birattari, M. and Blum, C. and Gambardella, L.M. and Mondada, F. and Stuetzle, T. (Eds.). Lecture Notes in Computer Science 3172 Springer Verlag, 25-36, 2004.

[33] Eshelman, L. J.: The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination, *Foundations of Genetic Algorithms*, Morgan Kaufmann, 265-283, 1991.