Ant Colony Optimization Approaches for the Dynamic Load-Balanced Clustering Problem in Ad Hoc Networks

Chin K. Ho, *Member, IEEE,* & **Hong T. Ewe,** *Sr. Member, IEEE*
Faculty of Information Technology
Multimedia University,
Persiaran Multimedia, 63100 Cyberjaya, Malaysia.
ckho@mmu.edu.my, htewe@mmu.edu.my

Abstract - This paper presents three ant colony optimization (ACO) approaches for a difficult graph theoretic problem formulated from the task of computing load-balanced clusters in ad hoc networks. These three approaches contain novel strategies for adapting the search process to the new problem structure whenever an environment change occurs. An environment change occurs when nodes in the network move. Dynamic changes in problem structure pose a great challenge for ACO algorithms because the pheromone information is rendered inaccurate and inconsistent. Hence, all three strategies to enable ACO to work in a dynamic setting have a common objective, that is, to adapt the pheromone information to closely reflect the new problem structure. The first approach is the population-based ACO algorithm (P-ACO) that incorporates a novel solution repair procedure. The second approach, which we call PAdapt, works by adapting three major algorithm parameters following an innovative strategy. The third approach, which we term GreedyAnts, uses a greedy solution construction strategy to bias the pheromone information towards the new problem structure. Empirical results show that GreedyAnts is very competitive with P-ACO, while PAdapt is less impressive. The GreedyAnts approach is advantageous over P-ACO because it does not require a solution repair heuristic that incurs additional processing.

Keywords **-** Ant colony optimization, search and optimization, dominating set problem.

I. INTRODUCTION

The ant colony optimization (ACO) metaheuristic [1] is a stochastic search and optimization technique that has been successfully used to address a wide array of combinatorial optimization problems. Some examples include the traveling salesman problem [2], quadratic assignment problem [3], job-shop scheduling [4], vehicle routing [5], sequential ordering [6], graph coloring [7], and shortest common super-sequence [8]. The majority of results reported in the literature were derived from the applications of ACO on static problems. A static problem is one in which the structural definition of the problem and the problem related data remain unchanged while the problem is being solved.

ACO can be regarded as a collaborative, multi-agent approach to problem solving. It maintains a colony (i.e. population) of artificial ants. ACO is iterative. In every iteration, each ant constructs a complete solution to the problem. The entire search experience of the colony is encoded in a global data structure commonly known as the pheromone trails. The pheromone associated to each solution component indicates how it has contributed in the construction of good solutions in the past. Each ant depends on two key factors in selecting a solution component. The first is of course the pheromone associated to that component. The second factor is a heuristic (normally greedy) measure giving the desirability of that component.

Recently, there has been increasing interests on designing ACO algorithms for dynamic problems. These are problems whose structure changes while the problem is being solved. This change in problem structure is termed as an 'environment change'. This is challenging to an ACO algorithm because an environment change would render certain portions of the pheromone trails inaccurate with respect to the new problem structure. In recognition of this, many research efforts [9, 10, 11, 12] for designing ACO algorithms for dynamic environments has focused on devising strategies for adapting the pheromone information so that ants are not misled by the outdated search experience.

The aim of this paper is to describe three ACO approaches, each of which incorporates a specific strategy for responding to environment changes. These three approaches are assessed on a variant of the dominating set problem [13], which is formulated from the load-balanced clustering problem in ad hoc networks.

II. PROBLEM DESCRIPTION AND FORMULATION

An ad hoc network [14] is modeled as an undirected graph $G = (V, E)$, where V is the set of vertices representing the nodes of the network and E is the set of edges. Let $\| ij \|$ be the Euclidean distance between nodes i and j. An edge $e_{ij} \in E$ exists between i and j if they can receive each

other's transmission, i.e. if $\| ij \| \leq R$, where R is the transmission range of the nodes. Nodes are organized into clusters to enhance manageability. Each cluster is coordinated by a special node called the cluster head (abbreviated as CH). Cluster members are most commonly selected based on their distance from the candidate CH.

The problem of load-balancing the clusters arises due to the limitation of resources available to CHs. CHs can efficiently support up to a certain number of nodes. It is desirable to have each CH support about the same number of nodes. A measure of the load of a CH is the number of ordinary nodes it has in its cluster [17, 18]. In actual ad hoc networks, the work load of a CH is directly proportional to its cluster size. The task of constructing load-balanced clusters can be formulated as follows: find a set of dominating nodes V' (the CHs) such that the nodes in $V - V'$ can be distributed as evenly as possible among the members of V' subject to the constraint that the number of dominated nodes per CH is at most *max_load*. The dominated node is said to be *covered* by its dominating node. The quality of the clustering can be measured by the variance of the load of the CHs, and is given by the following:

$$LBF = \frac{\sum_i (x_i - \mu)^2}{n_c} \qquad (1)$$

where n_c is the number of clusters, x_i is the load of CH i, and $\mu = (N - n_c)/n_c$. A smaller LBF signifies better load distribution.

III. THE ACO APPROACHES FOR DYNAMIC ENVIRONMENTS – RELATED WORK

Early work in this area saw researchers propose techniques that directly modify the pheromone information to reflect the new problem structure. One of such effort was reported in [11]. It proposed three pheromone modification methods for addressing the dynamic traveling salesman problem (TSP). The first two, known as the η-strategy and τ-strategy perform local pheromone modifications. Both strategies rely on the knowledge of where the change in the problem structure occurred. The third strategy, known as the Restart-strategy performs global pheromone modification by reinitializing all pheromone values by the same computed quantity. The global strategy produced very good results. However, the authors commented that the success of the global strategy may be due to the very minute environment change – only one city was inserted or deleted.

The population-based ACO (P-ACO) [9] which was originally proposed for static problems has been demonstrated to be promising in addressing dynamic problems. It maintains a population of solutions that determines the updates to the pheromone intensities. Initially, the population is empty. The best solution from each cycle is added into the population until the maximum population size is reached. After this has occurred, a strategy is used to replace a population member with a new solution from the latest cycle. When the population member is removed, the pheromone contributed by it is also subtracted from the pheromone matrix. This gives P-ACO the ability to undo previous pheromone updates. In a subsequent work [10], the authors showed how the original P-ACO can be adapted to dynamic problems. Whenever a change in the problem structure occurs, each solution in the population is repaired using a problem-specific heuristic. Changes to the population are then effected back to the pheromone matrix.

A solution deconstruction strategy [12] has also been proposed for making ACO adapt to environment changes in the context of constraint satisfaction problems. When a change in environment occurs, the deconstruction process is performed on the best solution from the current colony. This process is iterative. In each iteration, a solution component from the best solution is selected based on the nature of the event that occurred, and this component is removed from the solution. If this action results in worse violations of constraints, the solution component will be reinserted, and another different component will be tried. This process continues until zero constraint violation is achieved. In the worst case, all solution components making up the best solution will be processed.

IV. THE PROPOSED APPROACHES FOR HANDLING DYNAMIC ENVIRONEMTS

An ACO metaheuristic for the static version of this problem has been recently published in [15]. For the sake of completeness, a brief description of this metaheuristic is presented here.

Each ant begins with an empty solution, and incrementally constructs a complete solution by selecting CHs. After the selection of a CH, cluster members will be assigned to the CH. This process continues until every node is either a CH or already has been assigned to a CH. The selections of CHs are probabilistic, with the selection probability of each node determined by the product of its pheromone value and the visibility measure. The visibility measure is a function formulated by first assigning a weight to each node. The weight of a node tells the number of uncovered nodes it can cover if it is selected as a CH. The visibility measure will give a higher preference value to nodes whose weight is closer to the median of 1, 2, … *max_load*. The objective function is the LBF measure given in eq. (1).

The dynamic version of the problem arises as a result of node mobility. The composition of clusters can change under two scenarios. Firstly, a cluster member may move beyond the transmission range of its current CH. Secondly, a CH may move, resulting in at least one of its current cluster members to be out of its transmission range.

In the ensuing three subsections, we explain how three strategies for handling dynamic environments are incorporated into this ACO metaheuristic.

A. Approach 1: Population Based ACO (P-ACO)

Our contribution here is the design of the repair heuristic for the intended problem. This repair heuristic takes the following form:

For each node v that leaves its current cluster C_i, do

i. Find a neighbor cluster C_j with the smallest $load(C_j)$, i.e. the smallest number of cluster members.
ii. If $load(C_j) < max\_load$, then assign v to C_j. Stop and process the next node that moved. Otherwise, go to iii.
iii. If $load(C_j) \geq max\_load$, break C_j and form two new clusters C_s and C_t. This is done by first selecting a new CH each for C_s and C_t. Assign node v and the remaining nodes of C_j to C_s and C_t such that the cardinality of the new clusters are as close as possible.

For the P-ACO implementation, we use the age-based population update policy. This policy replaces the oldest solution in the population, and has been proven to be effective in [10].

B. Approach 2: PAdapt – Responding by Parameter Adaptation

The aim of this approach is to modify the outdated pheromone information by firstly reducing the influence of pheromone (which by convention is controlled via a parameter called α in solution construction, and at the same time increasing the influence of the visibility measure (controlled by parameter β). Secondly, the PAdapt approach also introduces a higher pheromone evaporation rate to allow ants to 'forget' knowledge from the outdated search experience. An environment change will trigger p% of the ants to construct solutions using the new parameter values (given in Table 1, column titled 'Adapted values'). For this work, p is set to 50%. This modified behavior will persists for a predetermined number of cycles (we use parameter *max*_cycle to refer to this value), which we set to five after conducting parameter tuning. This implies that there exist two groups of ants within the five cycles following an environment change. The iteration-best ant from each group is allowed to update the pheromone.

Parameter	Normal values	Adapted values
α	1	0.1
β	3	5
ρ	0.1	0.4

Table 1. Adapted values for three main parameters: α, β and ρ.

C. Approach 3: GreedyAnts – A Greedy Approach

The key idea behind the GreedyAnts is to have p% (p is set to 50% as in PAdapt) of the ants disregard the pheromone information, and deterministically select the node with the highest visibility value as the next CH. We call this group of ants greedy ants. The remaining 50% of the ants will behave in the usual way. Similar to PAdapt, both the iteration-best greedy and ordinary ant will update the pheromone information. The greedy behavior will persist for five iterations (*max*_cycle = 5) following an environment change, after which the greedy ants will go back to their normal behavior.

V. COMPUTATIONAL SETUP

The performance of the proposed ACO approaches is evaluated on 12 problem instances representing four ad hoc networks. Each network is generated by randomly placing N nodes in an area of size M x M meters. From each network, four problem instances are obtained by varying the transmission range. The characteristics of the problem instances are given in Table 2. Each approach is run 50 times for each problem instance. Altogether, there are 600 runs for each approach. Each run takes a maximum of 200 cycles. Under normal operations, the three major parameters are given in Table 1 (see the 'Normal Values' column). All approaches uses a colony size of 30 ants. For P-ACO, the size of the solution population is set to 7 based on our parameter tuning effort.

For each run, node mobility takes place at the 50^{th}, 100^{th} and 150^{th} cycle. This gives a window of 50 cycles for recovery. In each mobility event (i.e. environment change), 5% out of the total number of nodes will move. The severity of environment change here is larger than those employed in previous efforts [9, 10, 11, 12]. We label the 51^{st} to 100^{th} cycle as Interval1, the 101^{th} to 150^{th} cycle as Interval2, and the 151^{st} to 200^{th} cycle as Interval3. Each approach will attempt to recover in these intervals.

Network Size	**Area** (m x m)	**Transmission Range** (m)
400 nodes	3000 x 3000	210, 220, 230
350 nodes	2500 x 2500	200, 210, 220
300 nodes	2000 x 2000	180, 190, 200
250 nodes	1500 x 1500	130, 140, 150

Table 2. Characteristics of the problem instances. There are four problem instances for each network created from different transmission ranges.

A. Performance Measures

The following measures are used to gauge the performance of each approach:

- *Reactivity* [16]. This is defined by eq. (2). It measures the number of cycles required to recover from any solution quality degradation. m is the cycle in which an environment change occurred, ε is a performance threshold parameter set to 0.01, m' is the earliest cycle after an environment change in which the inequality in eq. (2) can be satisfied. If this happens, we say that recovery has been achieved. If there is any failure to recover, the reactivity is set to 50, i.e. the recovery window size.

$$Reactivity(m) = \left\{ m' - m \mid m' \in \mathrm{N}, \frac{f_{m'}}{\min f_{[m-5,m-1]}} \le (1+\varepsilon) \right\} \quad (2)$$

where N is the set of natural numbers, $f_{m'}$ is the cost of the best solution at cycle m', and min $f_{[m\text{-}5,\ m\text{-}1]}$ is the best solution cost found in the five cycles preceding the environment change.

- *LBF at Recovery*. This is the best solution for cycle m'. Should an approach fail to recover within the permitted window, the value for this measure will be set to the average iteration-best LBF across the recovery window.

- *Best LBF*. This gives the best solution within each interval. This measure is useful in two ways. Firstly, it can be compared against that of *LBF at Recovery* to determine if solution quality improves beyond the point of recovery. Secondly, it gives an indication of the best performance if an approach failed to recover in an interval.

In order to compare results among different problem instances, a reference approach called Control is used. Control is simply the original ACO approach that contains no strategy to respond to environment changes. The performance of an approach X with respect to a performance measure P, denoted $P(X)$, is computed as the gain over Control:

$$Gain_P(X, Control) = \begin{cases} \dfrac{P(Control) - P(X)}{P(Control)} \text{ x } 100 & \text{if } P(X) \le P(Control) \\ -\dfrac{P(X) - P(Control)}{P(X)} \text{ x } 100 & otherwise \end{cases} \quad (3)$$

Smaller values for each performance measure indicate better performance. Eq. (3) computes, as percentage, the ratio of the measured performance difference to the larger of the two values. A positive gain means X performs better than Control.

B. Parameter Tuning

The following parameters are tuned: α, β, ρ, p, and *max_cycle*. The problem instance with the smallest transmission range for each of the four network sizes is selected for use in tuning. Tuning is performed with respect to two performance measures: *Reactivity* and *LBF at Recovery*. Each approach is run 20 times on a tuning instance, and the two performance measures are averaged over these 20 runs. For each tuning instance, each combination of parameter values is ranked according to *Reactivity* (Rank A) and *LBF at Recovery* (Rank B). The best rank is one. The best parameter setting for each strategy is determined by computing the overall rank, $Rank(\boldsymbol{c_i})$, as follows:

$$Rank(c_i) = \left[\sum_{k=1}^{size} RankA(c_i, k) + \sum_{k=1}^{size} RankB(c_i, k) \right] \cdot \frac{1}{size} \quad (4)$$

where $size$ is the number of tuning instances, $\boldsymbol{c_i}$ is the i^{th} combination of parameter values. We select the $\boldsymbol{c_i}$ that gives the smallest value for Eq. 4.

VI. RESULTS AND DISCUSSION

The results for each performance measure are presented for the three intervals using boxplots. Fig. 1, Fig. 2 and Fig. 3 shows the boxplots for the Reactivity, *LBF at Recovery* and *Best LBF* measures respectively. The plot for each approach is constructed using the gains computed in the 600 runs.

We first analyze the performance of the proposed approaches with respect to the *Recovery* measure. The corresponding boxplot is Fig. 1. During Interval1, both P-ACO and GreedyAnts performed slightly better than Control, with medians close to 10%. During Interval2, there was significant increase in the ability of both P-ACO and GreedyAnts to recover. In this interval, GreedyAnts

emerged the better approach with median of 50% compared to 40% for P-ACO. Even though the performance of P-ACO and GreedyAnts decreased in Interval3, the medians were maintained at above 20%. The

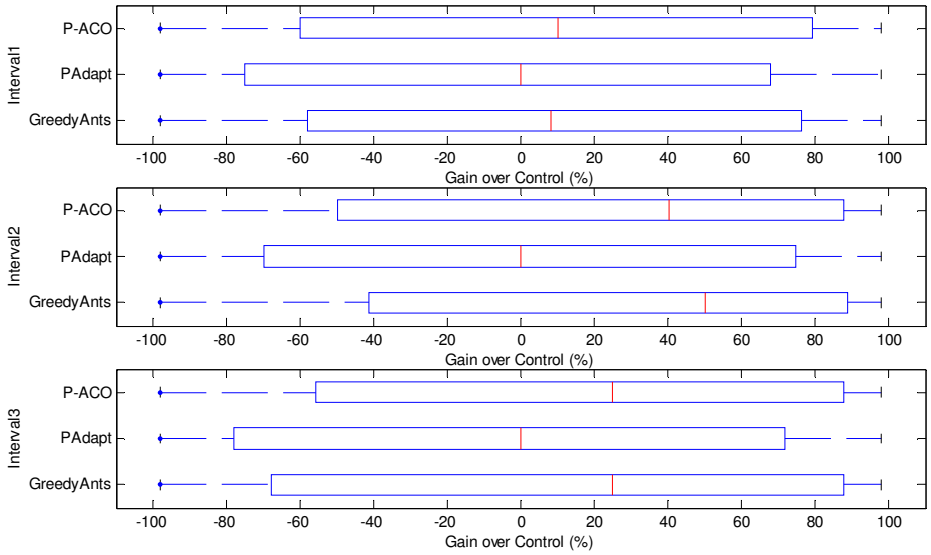

Fig. 1. Gains for P-ACO, PAdapt and GreedyAnts over Control for performance measure *Reactivity.*

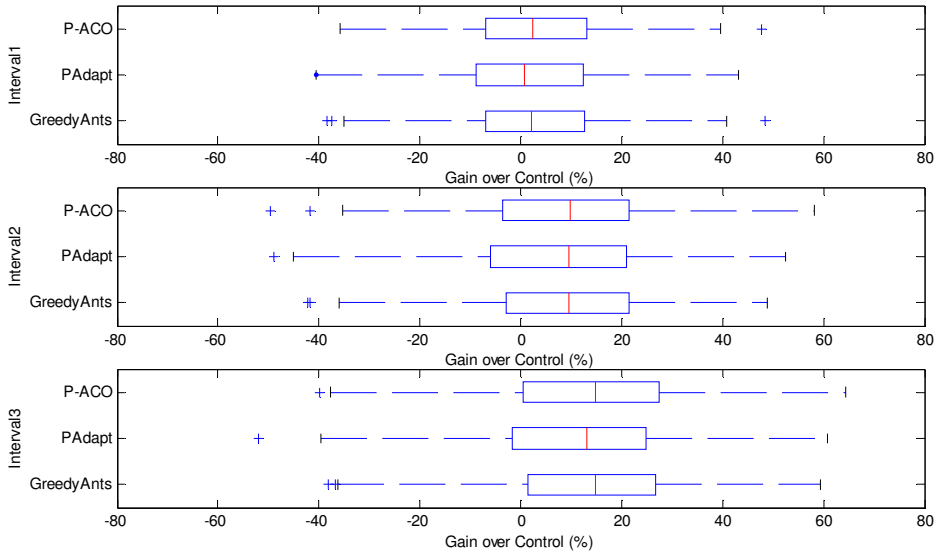

Fig. 2. Gains for P-ACO, PAdapt and GreedyAnts over Control for performance measure *LBF at Recovery.*

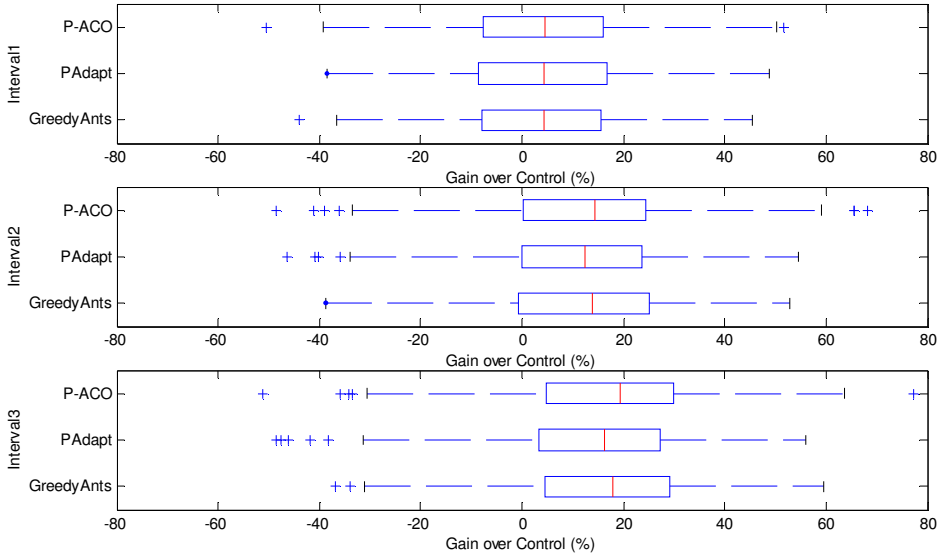

Fig. 3. Gains for P-ACO, PAdapt and GreedyAnts over Control for performance measure *Best LBF*.

PAdapt approach, however, did not manage to outperform Control as its medians are all zero for the three intervals.

For the second performance measure, i.e. *LBF at Recovery*, P-ACO, PAdapt and GreedyAnts were superior to Control in all three intervals. Fig. 2 shows that the performance for three approaches has consistently increased from Interval to Interval3. The better LBF produced at the point of recovery can be attributed to the pheromone adaptation strategies incorporated in the three approaches.

For the *Best LBF* measure (see Fig. 3), a similar performance trend can be observed. There are gradual improvements for all three approaches from Interval1 to Interval3. This result is important because it shows that the solution quality attained at recovery is maintained even after the recovery strategies is no longer in force after the first five cycles following the environment change. The best LBF recorded in each interval has, on average, made improvements over the LBF recorded at recovery. The average improvements were 11.97%, 12.22% and 11.96% for P-ACO, PAdapt and GreedyAnts respectively.

The empirical results did not indicate if any approach is generally more preferable with respect to network size. However, a frequently occurring trend is that GreedyAnts tend to show better reactivity for Interval2 and Interval3. This observation provides a hint that GreedyAnts may be the better approach to handle higher mobility rate.

The parameter tuning process revealed that p = 50% is the best setting for both PAdapt and GreedyAnts. We have also investigated the effects of selecting a higher (p = 70%) or lower (p = 30%) value. We measure, across all problem instances, the improvement rate as the difference between the number of runs that gave better and worse results for the *Reactivity* and *LBF at Recovery* measures. This difference is expressed as percentage over the total number of runs, and is formulated in Eq. 5 with respect to a performance measure Y.

$$Improvement_Y = \frac{R^+ - R^-}{R^{total}} \times 100 \quad (5)$$

where R^+ is the number of runs that saw improvement, R^- is the number of runs that saw degradation, and R^{total} is the total number of runs. Table 3 shows the results of this study. For PAdapt, reducing the value of p to 30% produced degradation for both performance measures. Increasing p to 70% showed a small improvement for the Reactivity measure, but a large 11% degradation for the LBF at Recovery measure. Now, for the GreedyAnts approach, changing the value of p to both 30% or 70% produced inferior results. A smaller value for p causes degradation of a larger magnitude. This trend is similar to that observed for PAdapt with reduced p value. Using a smaller value for p means that a majority of the ants are oblivious to the environment changes, therefore still placing the same emphasis on the outdated pheromone.

p	PAdapt		GreedyAnts	
	Reactivity (%)	LBF at Recovery (%)	Reactivity (%)	LBF at Recovery (%)
30%	-12	-7	-9	-6
70%	+ 4	-11	-4	-3

Table 3. Net improvements for p = 30% and 70%. Positive entries mean there are more runs that showed improvement than degradation.

Overall, the parameter adaptation strategy was the least effective. Both P-ACO and GreedyAnts were consistently superior to PAdapt for all three performance measures. Even though P-ACO is competitive with GreedyAnts, the GreedyAnts approach has a distinct advantage. It depends only on the updated visibility measure for adapting the pheromone information. This introduces no additional processing requirements. On the other hand, for P-ACO, we first need to design a good heuristic to perform solution repair. The application of the repair heuristic uses additional processing time, which is directly proportional to the population size and severity of the environment change.

VII. CONCLUSIONS

We have presented three ACO approaches for dynamic environments. PAdapt and GreedyAnts are original contributions originating from our research work. P-ACO is a framework that has been proposed earlier, and we have developed a repair heuristic for the load-balanced clustering problem. All three approaches are inspired by one common objective, that is, to adapt the partially outdated pheromone information so that it is consistent with the new problem structure.

Empirical results reveal that GreedyAnts and P-ACO yielded similar performance, while PAdapt was more inferior. Interestingly, the response from each approach was less impressive for the first environment change. However, improvements were evident in subsequent environment changes. In terms strategies to respond to environment changes, three conclusions can be asserted. Firstly, repairing previous good solutions, and then using them to influences future pheromone deposits – i.e. the P-ACO approach - is verified to be effective. Secondly, results for GreedyAnts showed that it is viable to use groups of ants with different behavior to collectively solve a common problem. Thirdly, simply adjusting the major ACO parameter values is insufficient for the algorithm to respond to environment changes.

REFERENCES

[1] M. Dorigo, V. Maniezzo, and A. Colorni, "Positive Feedback as a Search Strategy," Technical Report 91-016, Politecnico di Milano, Italy (1991).

[2] M. Dorigo, and L.M. Gambardella, "Ant Colony System: Optimization by a colony of cooperating agents," *IEEE Trans. Systems, Man, and Cybernetics – Part B*, Vol. 26, No. 1, 29-41, 1996.

[3] V. Maniezzo, and A. Colorni, "The ant system applied to the quadratic assignment problem," *IEEE Trans. Knowledge and Data Engineering*, Vol. 11, No. 5, 769-778, 1999.

[4] A. Colorni, M. Dorigo, V. Maniezzo, and M. Trubian, "Ant system for job-shop scheduling," *Belgian Journal of Operations Research, Statistics and Computer Science (JORBEL)*, Vol. 34, 39-53, 1994.

[5] L.M. Gambardella, E. Taillard, and G. Agazzi, "Ant colonies for vehicle routing problems," In Corne, D., Dorigo, M. and Glover, F., editors, *New Ideas in Optimization*, McGraw-Hill, 1999.

[6] L.M. Gambardella, and M. Dorigo, "HAS-SOP: An hybrid ant system for the sequential ordering problem," Technical Report 11-97, IDSIA, Lugano, CH, 1997.

[7] D. Costa, and A,.Hertz, "Ants can color graphs," *Journal of the Operations Research Society*, Vol. 48, 295-305, 1997.

[8] R. Michel, and M. Middendorf, "An island model based ant system with lookahead for the shortest supersequence problem," In: Eiben, A.E., Back, T., Schoenauer, M., and Schwefel, H.P. (eds.): *Proceedings of PPSN-V, Fifth International Conference on Parallel Problem Solving from Nature*, 692-701, 1998.

[9] M. Guntsch, and M. Middendorf, "A Population Based Approach for ACO," *Proceedings of EvoWorkshops, Lecture Notes in Computer Science*, Vol. 2279. Springer-Verlag, Berlin Heidelberg New York, 72-81, 2002.

[10] M. Guntsch, and M. Middendorf, "Applying Population Based ACO to Dynamic Optimization Problems," In: Dorigo, M., Di Caro, G., Sampels, M. (eds.): *Proceedings of Ant Algorithms, Third International Workshop, ANTS 2002, Lecture Notes in Computer Science*, Vol. 2463. Springer-Verlag, Berlin Heidelberg New York, 111-122, 2002.

[11] M. Guntsch, and M. Middendorf, "Pheromone Modification Strategies for Ant Algorithms Applied to Dynamic TSP," In: Boers, E.J.W. et al. (eds.): *Proceedings of EvoWorkshops, Lecture Notes in Computer Science*, Vol. 2037. Springer-Verlag, Berlin Heidelberg New York, 213-222, 2001.

[12] M. Randall, "Generalizing Ant Colony Optimization for Dynamic Optimization Problems," Technical Report 05-01, Faculty of Information Technology, Bond University, Australia, 2005.

[13] G. Chartrand, and L. Lesniak, *Graphs and Digraphs*. Chapman & Hall/CRC, Boca Raton, 2005.

[14] C.K. Toh, *Ad Hoc Mobile Wireless Networks: Protocols and Systems*. Prentice Hall, 2001.

[15] C.K. Ho, and H.T. Ewe, "A Hybrid Ant Colony Optimization Approach for Creating Load-Balanced Clusters," *In Proceedings of the IEEE Congress on Evolutionary Computation*, Edinburgh, Scotland, 2010-2017, 2005.

[16] K. Weicker, "Performance Measures for Dynamic Environments," *Lecture Notes in Computer Science*, Vol. 2439. Springer-Verlag, Berlin Heidelberg, 64-73, 2002.

[17] M. Charterjee, S.K. Das, and D. Turgut, "WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc Networks," *Cluster Computing*, Vol 5, 193-204, 2002.

[18] D. Turgut, S.K. Das, R. Elmasri, and B. Turgut, "Optimizing Clustering Algorithm in Mobile Ad Hoc Networks using Genetic Algorithmic Approach," *In Proceedings of IEEE Globecomm*, 62-66, 2002.