2016 International Conference on
Unmanned Aircraft Systems (ICUAS)
June 7-10, 2016. Arlington, VA USA

Multi Colony Ant Optimization for UAV Path Planning with Obstacle Avoidance

Ugur Cekmez[1], Mustafa Ozsiginan[2] and Ozgur Koray Sahingoz[3]

***Abstract*— In recent years, the availability of low-cost and autonomous unmanned aerial vehicles *(UAVs)* results in the use of them for different types of military and commercial applications. The crucial part of the autonomous *UAVs* is their online or offline path planning algorithms. In the literature, there are many types of solutions, which use evolutionary and/or swarm intelligence approaches. Ant colony optimization is one of the mostly used algorithms, which has been applied to solve different type of path planning problems. Mainly, most of these studies have focused on a single colony ant colony optimization *(ACO)*, which can find better solutions in fewer computation times. However, it is able to converge to a sub-optimal solution in the planning process. One approach to avoid the premature convergence is the use of Multi-Colony *ACO*, in which a number of ant colonies try to find an optimal solution cooperatively by exchanging their valuable information with each other. In this paper, it is aimed to implement an obstacle avoidance *UAV* path planning by using Multi-Colony *ACO* algorithm. We experimentally investigate the use of Multi-Colony *ACO* approach results from an effective path planning for *UAVs* with a comparison to a single colony *ACO* approach.**

I. INTRODUCTION

In the last few decades, Unmanned Aerial Vehicles *(UAVs)* have been widely used in the military and civilian context with a different type of missions such as surveillance, reconnaissance, targeting, seeding *(in farms)*, weather forecast, command and control of troops, etc. The path planning process is an important area of interest in the usage of *UAVs* with these scenarios. It allows the *UAV* to autonomously compute an optimal or near-optimal flight path between the initial point to the end by checking some specific control points or fulfill some mission specific constraints *(path length, obstacle avoidance, fuel consumption, etc.)*. In the use of *UAVs*, generally path planning process try to arrange a new path *(generally in 3D space)* for checking a specific area or specific control points as depicted in Figure 1. In recent years, we have witnessed a rapid growth in popularity and development in the solution of the *UAV* Path Planning problem especially with the help of evolutionary and/or swarm intelligence algorithms, such as Genetic Algorithms, Ant Colony Optimization, Particle Swarm Optimization, Simulated Annealing, etc [1], [2].

Swarm intelligence seeks an optimal solution by inspiration in the behavior of swarms of animals *(such as bees, ants, bats, glowworm, etc.)* to reach a more sophisticated intelligent behavior. Ant Colony Optimization *(ACO)* is one of the mostly used swarm intelligence algorithm, which takes its inspiration from the foraging behavior of real ant colonies to solve combinatorial optimization problems in the virtual environment. They are based on a colony of artificial ants, which work cooperatively to reach an optimal/near optimal solution of the problem. Due to real ants properties, an ant can communicate with others in the swarm through artificial pheromone trails.

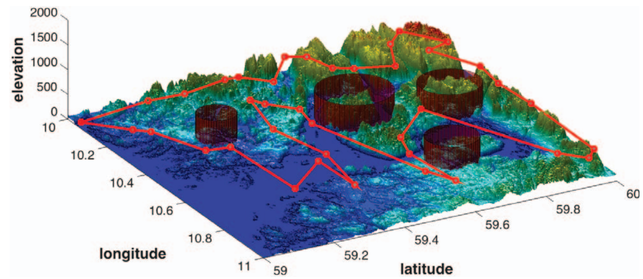

Fig. 1. A UAV Path Planning for 40 control points and 4 radar areas

Although, *ACO* is very efficient swarm optimization algorithm, and it has been tested on a large number of academic and real-world problems, there are some faults about it to converge slowly to the solution and prone a stagnation due to the search to a certain extent, possibility of finding same solutions, which is not conducive to find better solutions [19].

To avoid premature convergence and stagnation problems of classical *ACO*; a Multi-Colony *ACO* was proposed [13], [18]. Apart from *ACO*, this algorithm consists of several different ant colonies by using a separate pheromone table to maximize the search area explored, and it can be used in different types of discrete optimization problems. In this optimization, each colony produces various results in their iterative solution steps. After each iteration *(or after a number of iterations)*, each colony may exchange its valuable information such as the best solution of the colony with its neighbor colony/colonies. After this information exchange, each colony updates its pheromone matrix with the incoming pheromone information.

In this paper, we proposed a *UAV* path planning algorithm to construct a feasible path by using a Multi-Colony *ACO* algorithms to reach an optimal or near optimal solution quickly. We also compared the proposed algorithms with the classical *ACO*. The experimental results demonstrate that the proposed algorithms produce more feasible solutions in *UAV* path planning tasks than the classical *ACO* algorithm.

The remainder of the paper is organized as follows. In Sec-

[1]Computer Engineering Department, Yildiz Technical University, Istanbul, Turkey `ucekmez@yildiz.edu.tr`
[2]Aeronautics and Space Technologies Institute, Turkish Air Force Academy, Istanbul, Turkey `mustafaozsiginan@gmail.com`
[3]Computer Engineering Department, Turkish Air Force Academy, Istanbul, Turkey `sahingoz@hho.edu.tr`

978-1-4673-9333-1/16/$31.00 ©2016 IEEE

tion II, the related works about the research area are detailed. Section III analyzes the background knowledge about the research by emphasizing the importance of the Multi *ACO* algorithms. The proposed algorithms are detailed and the experimental results are given in Section IV, and Section V respectively. Finally, Section VI draws a conclusion.

II. RELATED WORK

In recent years, researchers have been focusing on autonomous robots such as droids, vehicles and *UAVs* for reducing human casualties and other missions such as surveillance, reconnaissance, targeting, etc. Especially, development of more intelligent *UAVs* are important to perform missions such as surveillance, future combat, monitoring nuclear and biological hazards for protecting human life.

While the *UAV* is flying in air for a given mission, it has to ensure and detect any obstacles, such as mountains, buildings, radars and other air crafts. In literature, there are many different evolutionary, heuristics and hybrid methods were discussed. In this study, a Multi-Colony ACO algorithm is developed to generate safe trajectory as a path where the path is set up from an initial position to a final destination by avoiding any obstacle encountered.

In the aim of generating paths for *UAVs*, there are two different types of trajectories handled online and offline. In online trajectory scenarios, the *UAV* does not have its prior knowledge about the environment and it has to get required data from its sensors, operators and/or commanders in order to complete the mission. On the other hand, in offline trajectory scenarios, the *UAV* knows about its environment and plans its movement before the mission has started. In the study [5], authors present an evolutionary algorithm based on online/offline path planning for UAV navigation to calculate a curved path with desired characteristics in a *3-D* rough terrain environment. The proposed method shows that it is capable for producing optimum path in a small number of iterations. In addition, Cekmez et al. [6] propose an algorithm for getting feasible solution for offline *UAV* route planning. According to that study, a genetic algorithm based parallel approach is designed and implemented to improve the performance for large scale problem domains. The experimental results show promising results when comparing the solutions with the classical serial *CPU* implementations.

Zhan et al. propose a Fitness-Scaling Chaotic Artificial Bee Colony *(FSCABC)* algorithm for path planning of unmanned combat aerial vehicles in their study [7]. The proposed approach is more efficient than classical *GA, ACO, CABC* algorithms. Their experimental results show that *FSCABC* is more powerful, it uses less iterations and it needs less *CPU* time and resource comparing to elite genetic algorithm with migration, particle swarm optimization and chaotic artificial bee colony. Moreover, such bio-inspired algorithms take longer when the problem domain gets bigger. Because of this, [2] tests and proposes a newly algorithm for calculating this time consuming *UAV* path planning problem. The proposed algorithm in that study transforms traditional *ACO* algorithm to a massively parallel *ACO* using *GPU* architecture. According to the experimental results, Parallel *ACO* has more performance in speed and solution quality than the corresponding traditional *ACO* for *UAV* path planning.

Ant Colony Optimization *(ACO)* was inspired from foraging behavior of the real ants working together to find the shortest path between anthill and food source. Dorigo introduced the first implementation of *ACO* algorithm in early *1990s* [4], [8]. Using *ACO* for finding best solution, initially proposed by Dorigo et al. that was named as Ant System to apply for solving Traveling Salesman Problem (TSP) [9] which is one of the mostly known combinatorial optimization problem. In the ongoing years, same studies continue to solve *TSP* with *ACO* [10] with some modifications. One of the most important modification is the usage of local search techniques such as 2-opt, 3-opt, etc.

Although *ACO* is very suitable for lots of combinatorial and polynomial problems, research showed that *ACO* algorithm can find near optimal solution at early stage of search process. However, all ants quickly converged to a single solution. Therefore it is no longer able to find the global optimum solution [11]. This problem is called as search stagnation problem. In literature, there are many studies in which researchers proposed alternative solutions for overcoming this type stagnation problem.

One of the proposed solution is named as Multi-Colony ACO. Sim et al. proposed this approach for load balancing in circuit-switched networks [12]. According to Multi-Colony ACO, each colonies lay their own pheromones and are cooperatively working to solve optimization problems. Information exchange in Multi-Colony ACO algorithm is proposed by Middendorf et al. [13]. According to their study, multiple colonies have to exchange information after completing a certain number of iterations. The result of their study shows a decrease in run time of the algorithm with increasing interval between information exchanges.

In study [14], authors also investigated performance comparison of Multi-Colony *ACO* algorithms with traditional single colony *ACO* algorithms on different dynamic environment. They used two types of colonies those were homogeneous where the colonies have the same behavior and heterogeneous where the colonies have different behaviors. According to their experimental results, Multi-Colony *ACO* algorithms have promising performance than traditional *ACO* in dynamic environments. The reason for better performance of Multi-Colony ACO algorithms is that they imply on many engineering area especially non-deterministic NP-Complete problem like path planning. In study [15], authors investigated the performance of traditional *ACO*, *GA* and proposed algorithm of Multi-Colony ACO for multi unmanned combat aerial vehicles *(UCAVs)* cooperative path planning problem. They constructed and presented the mathematical formulation of path planning with constrains. Then, they simulated *GA*, traditional *ACO* and proposed Multi-Colony ACO to find a path that has an optimization criterion. Experimental results of their Multi-Colony ACO implementation has better performance than *ACO* and *GA*.

Hao et al. [16] examined the path planning problem for mobile robots to avoid from an obstacle by using the *MAK-LINK* graph theory and Multi-Colony ACO approach. They compared the Multi-Colony ACO system against traditional *ACO*. As their experimental results suggest, Multi-Colony ACO is very efficiently applied to obstacle avoidance robot path planning and the solution quality can be improved as the colonies exchange information and it is better to exchange the local best solution only with the neighbors in a directed ring.

III. BACKGROUND

A. Traveling Salesman Problem

The aim in Traveling Salesman Problem (TSP) is simply to route a salesman through a number of cities where the locations are known. The criterion here is to construct this route by passing every given city once and reaching the starting point by ensuring that the route is the shortest path. In *TSP*, there are $(N-1)!/2$ single solutions where *N* is the number of cities. At least one of these solutions is the best one with respect to its total distance. The optimum solution of a *TSP* can be found by a brute-force search algorithm but having a large scale problem may fail the search within the given time constraint and the best solution found in a predetermined time interval might not be a feasible one. In addition, the searching increases exponentially as the problem domain gets bigger, which makes the current hardware impossible to solve it by a full search method. Considering such limitations, using the optimization algorithms focused on the problems such as *TSP* can be seen as a more effective way to study on large scale domains.

B. Swarm Intelligence for UAV Path Planning

Since it is very resource and time consuming to solve *NP-complete* problems in conventional full domain search techniques, it is widely accepted that these kind of problems can be approximated with the help of evolutionary algorithms. Although evolutionary algorithms do not guarantee to find the optimal solution, they can be applied to problems in order to calculate a feasible result in reasonable time. In general, there are heuristics categorized in evolutionary algorithms based on biology, chemistry, swarm intelligence, etc. These heuristics can also be combined to yield new approaches to the problems. In this study, Ant Colony Optimization, a swarm intelligence based evolutionary algorithm is designed and implemented to solve *UAV* path planning problem.

C. Ant Colony Optimization

Ant Colony Optimization is a technique specially focused on solving *NP-complete* problems in theory of computation by using its heuristic operators. It is a swarm intelligence based approach inspired by the collective behavior of animals in the wild. As the animals find better ways for food and collaborate in order to ensure the continuity of their swarm, the optimization algorithm try to find and adapt new ways of sharing knowledge of individuals to new generations. In general, individuals in the swarm exhibit homogeneous behavior to find solutions in a distributed structure. Each individual drives towards the target of their swarm and shares its information in different ways. As the generations progress, new individuals are likely to converge to better solutions than the previous ones. In this context, Dorigo et al first used ants and ant colonies for solving *TSP* [3]. The reason of choosing a problem like *TSP* is because that *TSP* is a well-known and comprehensible problem for benchmarking between the optimization algorithms as well as its is well suited for the *ACO*.

In the *ACO* approach, there are a number of ants representing distributed agents. Each ant is intended to search for the shortest path in the given cities. As the *TSP* requires, each ant has to visit all the cities exactly once and return to their starting point. Each ant is deployed to a random city and required to perform a movement to another city by applying the random proportional rule given in Equation 1. In this equation, i is the current city that ant k is deployed and j is one of the next candidate cities that is unvisited yet. p_{ij}^k represents the probability of ant's tendency to visit j when it is in i. τ_{ij} is the amount of pheromone between these cities and n_{ij} is another representation of $\frac{1}{d_{ij}}$ where d_{ij} is the distance from i to j. α and β are the custom parameters which effects the importance of pheromones and distances while selecting the next city. Note that the pheromone table is first initialized with a small floating number.

$$p_{ij}^k = \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{il}]^\alpha \cdot [\eta_{il}]^\beta} \tag{1}$$

After the tours are constructed separately by each ant, pheromone table, the ants use to communicate, is updated. This is an important step because all the next generations are fed by the pheromone table and it helps the ants to head for better selections. Different approaches according to the problem can be applied when updating pheromones. Each ant is allowed to update the pheromone table as much as $\frac{Q}{C_k}$ where Q is considered as ant's ability to affect corresponding indexes in the pheromone table and C_k is the fitness value of the path. The update rule is shown in Equation 2.

$$\Delta\tau_{ij}^k = \begin{cases} \frac{Q}{C^k} & edge(i,j) \in T^k \\ 0 & otherwise \end{cases} \tag{2}$$

The update rule is applied to each index in the pheromone table through the cities in the path the ant constructed. Combined form of the rule is demonstrated in Equation 3.

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^k \tag{3}$$

After updating the pheromones, the process continues over iterations until the termination criterion has met. At the beginning of each iteration, the pheromones on the roads are evaporated. Evaporation is a technique in order to prevent the stagnation of each generations. The evaporation technique is shown in Equation 4.

$$\tau_{ij} \leftarrow (1-\rho) \cdot \tau_{ij} \tag{4}$$

A general overview of the *ACO* is depicted in Algorithm 1.

Algorithm 1 Basic Workflow of single ACO

```
procedure ACO(pheromones,distances)
    Set parameters
    Initialize pheromone trails
    while termination condition not met do
        EvaporatePheromones
        ConstructPath
        UpdateTrails
    end while
end procedure
```

The tour construction of an ant is described in Algorithm 2.

Algorithm 2 Tour construction of an ant

```
procedure CONSTRUCTPATH(ant,variables)
    path[0] ← pick a random city
    for j = 1 to n − 1 do
        for k = 1 to n − 1 do
            prob[k] ← CalculateProb(path, k)
        end for
        path[j] ← SelectCity(prob)
    end for
    fitness ← CalculateFitness(path)
end procedure
```

D. Multi-Colony Ant Colony

In the structural concept of *ACO*, there is a possibility of converging to a sub-optimal solution and the ants in the colony may be stuck in the same route because of pheromone values between the control points. This stagnation probability can be reduced to a certain extent by fine tuning the parameters such as number of ants, pheromone update and evaporation rate, alpha and beta, etc. However, fine tuning does not guarantee to find better solution and the parameters should be changed according to many criteria depending on the problem set. In this context, new optimization techniques arise in the area.

One of the improvements for *ACO* is to transform it to a Multi-Colony ACO where *N ACO* work for the same problem set distributively and share their local colony knowledge among the other colonies in specific intervals. In the Multi-Colony ACO, there are more than one colonies cooperating in order to better explore the problem space. Each colony has its own pheromone table and there is a synchronization point where distributed colonies share their accumulated knowledge with the other colonies. Knowledge sharing is as simple as making their private pheromone values available among others. In such a scenario, having many different colonies at hand makes it possible to keep them from converging quickly. It also yields a variety of results and diversity between the pheromone values so the next generations will have much more options to select. Considering that the generations of a single colony gain their knowledge from only their ancestors, having multiple colonies would make it more enhanced.

There are different approaches when sharing local knowledge of separate colonies. One of them is to exchange the information of the best ant among all the colonies. In this case, the global best ant of each *N* iteration updates the pheromone tables of all the colonies. Another approach is to route all the locally best ants to update the next colony's pheromone table in a circular structure in order to increase the diversity and make all the colonies influenced by their neighbors. Considering these scenarios, there also exists some hybrid methods combining these two and add other features when sharing pheromone values [18]. The basic knowledge sharing mechanism is depicted in Figure 2.

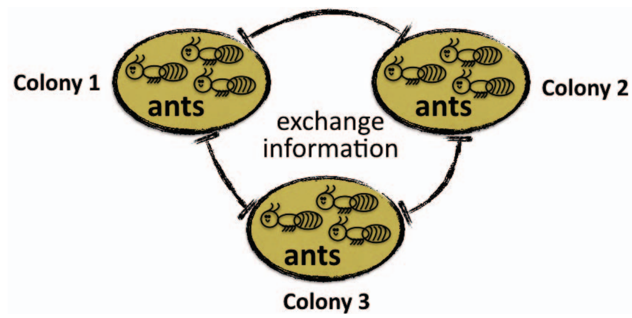

Fig. 2. Basic principle of sharing knowledge between colonies.

IV. MULTI COLONY APPROACH FOR UAV PATH PLANNING

In the proposed algorithm of this study, there are *N* colonies working on the same problem set. Each has its own pheromone table and is required to share its table after given time interval. When each colony completes one iteration, the ants are sorted according to their fitness values. The first half of the ants in each colony directly updates their pheromone tables. In order to keep the diversity, a *2-opt* local optimization is applied to the second half of the colonies. After the local optimization, at each tenth iteration, the colonies share their pheromone tables as depicted in Figure 3. The best 10 ants of each colony update both their own pheromone tables and of the colony next to them.

As the constructing path process is exactly same with a single ant, the crucial point here is to provide a better pheromone table for the ants. The better pheromone table means that the values should allow ants to be able to select more points to visit.

The proposed algorithm consists of the following main features:

- The optimization algorithm maintains several colonies
- All of the colonies have the same number of ants (equal to the number of cities in the problem),
- All of the colonies run for the same number of iterations
- All of the colonies share the same heuristic function
- All of the colonies have the same parameter values (α, β and ρ)

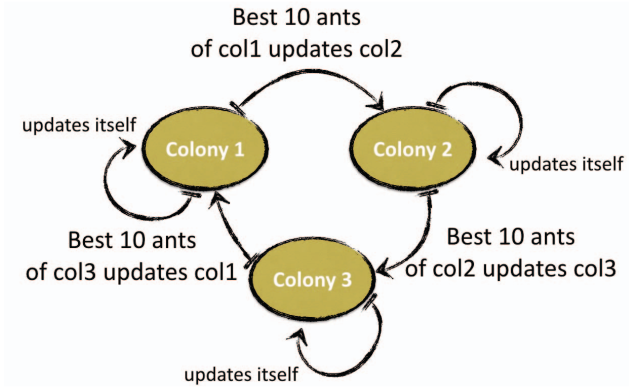

Fig. 3. Knowledge Sharing in three colonies Multi-Colony ACO

- Each colony has its own trail, in an attempt to maximize the search area explored
- At each iteration, a *2-opt* local optimization technique is applied to the worst 50% solutions of each colony
- At every *10th* iteration, the colonies share their knowledge by their local best ants. Best *10* ants of each colony update the pheromone table of the next colony and of theirs *10* times

The general overview of the Multi-Colony ACO is described in Algorithm 3.

Algorithm 3 Basic Workflow of Multi-Colony ACO

```
procedure MACO(pheromones,distances)
    Set parameters
    Initialize pheromone trails for each colony
    while termination condition not met do
        EvaporatePheromones
        for colony = 0 to len(colonies) do
            ConstructPath(colony_c)
        end for
        SortAntsForEachColony
        ApplyLocalOptimizationFor(Worst 50%)
        UpdateTrails
        SharePheromonesWithNeighbors
    end while
end procedure
```

V. EXPERIMENTAL RESULTS

In the experimental results, since the *UAV* path planning problem is similar to the *TSP* with respect to its limitations, the proposed approach firstly focused on solving the *TSP* in an efficient way, then transforming it into what needed for the real concept. For this aim, a virtual simulation environment is created with the parameters shown in Table I. It is assumed that the UAVs in the simulation environment are the high maneuver capability quadcopters and the results are obtained from an offline path planning scenario. In this context, some of the *TSP* libraries from *TSPLIB* [20] are used as a base problem set. Then a number of obstacles such as radars applied to the appropriate locations of the problem sets. These radars represent a forbidden area that the *UAVs* are not allowed to fly over. Since this kind of obstacle changes the best path of the problems, there is no any fixed fitness values of the used problems. That's why only the multiple colony results are compared with each other when having one, two, three and four colonies. There are N ants at each colony where N is equal to the number of control points in the problem set. Each problem is solved in 500 iterations and at every *10th* iteration, the knowledge sharing mechanism, namely pheromone sharing function, is run.

TABLE I
PARAMETER SETTINGS USED IN THIS STUDY

Parameters	Values
# of visiting points	52, 76, 100, 225, 439
# of colonies	1,2,3,4
# of ants per colony	# of points
Pheromone sharing rate	at every 10 iteration
α	2.0
β	5.0
ρ	0.02
Initial Pheromone	$\frac{1}{C_{NN}}$
Iteration	500
Q	1.618
Q_{best}	$2 * 1.618$

The proposed algorithm is mainly written in *C++* language. All the tests run for this experiment is automatized by a *Python* script and the corresponding figures as well as the output paths are produced with a *Python* library.

Figure 4 shows a sample output for the experimental results for a *100*-point problem set. The big red dots are the obstacles that the *UAVs* are not allowed to fly over, instead the *UAVs* try find other ways to reach their given control points.

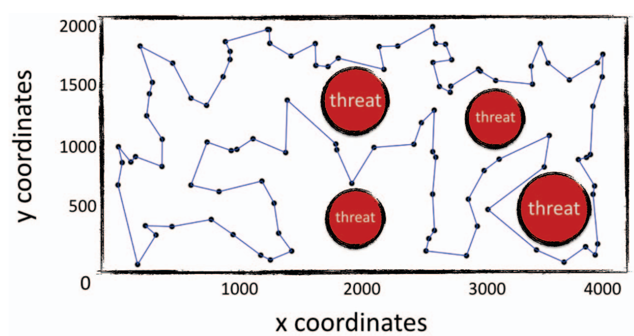

Fig. 4. Sample path for the 100-point problem including obstacles

As depicted in Figure 5, one colony gets stuck after some iterations and is not recovered till the end. The reason for this is the single pheromone table. After some generations, many ants choose the same path and the other probabilities evaporates, which makes them less likely to be chosen by the ants. Comparing a single colony solution to the other alternatives, it is clearly seen that having more than one colony makes it more optimized and it increases the chance to select different path for the ants. This is the main reason of why the multiple colonies continue to optimize their paths over the iterations.

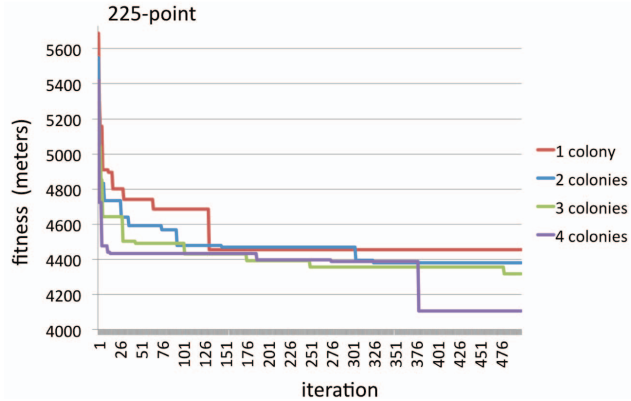

Fig. 5. Iteration results of 225-point problem for 1,2,3 and 4 colonies

In the experiments, it is seen that not getting stuck in one path is one of the yields of having multiple colonies. The other aspect is that multiple colonies have the chance to find better solutions comparing to a single colony option. As in shown in Figure 6, different problem sets gives variety of changes in the best fitness when comparing to the number of colonies in the solution. In the figure, the first bar that is equal to 100% represents the single colony result of the problem as a reference scale. The ones next to it are the increasing colony numbers. As the colony numbers get bigger, the best-so-far fitness also gets better with some degree. For the small-sized problems it seems that single colony is likely to find the optimal result but when the problem size get bigger, having multiple colonies shows its advantages. However, Figure 6 shows an unsteady distribution on the fitness change because of the disordered distribution of the cities in different problem sets.

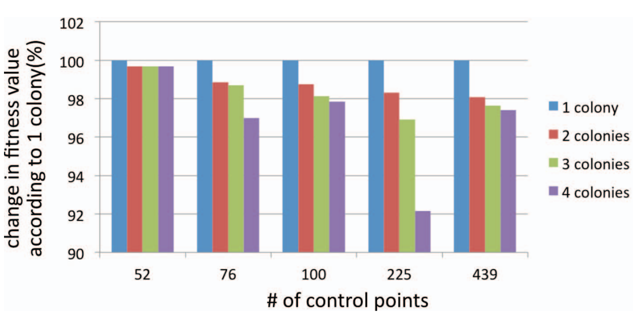

Fig. 6. Best solutions of the problem domains with different number of colonies

VI. CONCLUSION AND FUTURE WORK

In conclusion, this paper mainly discusses how to plan a feasible *UAV* path for checking a number of control points in a large area. The controlled area may have a radar network contains several radars with different ranges. The experimental results showed that the proposed model have a very good performance in comparison with the classical Ant Colony Optimization. In the future work, it is aimed to make additional experiments with a higher number of colonies. Also, performance is still a great problem if we increase the number of control points. Therefore, this algorithm should be converted to a parallel mode. At the same time, the whole area can be controlled by using a number of *UAVs*. This can be accomplished by developing a Multi-UAV path planning version of the proposed algorithm.

REFERENCES

[1] U. Cekmez, M. Ozsiginan, & O.K. Sahingoz. "Adapting the GA approach to solve traveling salesman problems on cuda architecture". In Computational Intelligence and Informatics (CINTI), 2013 IEEE 14th International Symposium on, pp. 423-428. IEEE, 2013 November

[2] U. Cekmez, M. Ozsiginan, & O.K. Sahingoz. "A UAV path planning with parallel ACO algorithm on CUDA platform". In Unmanned Aircraft Systems (ICUAS), 2014 International Conference on, pp. 347-354. IEEE, 2014.

[3] M. Dorigo, M. Birattari, & T. Stutzle, "Ant colony optimization", Computational Intelligence Magazine, IEEE, 1(4), 28-39, 2006.

[4] A. Colorni, M. Dorigo, & V. Maniezzo, "An Investigation of Some Properties of an Ant Algorithm", Proceedings of the Parallel Problem Solving from Nature Conference (PPSN 92), Brussels, Belgium, R.Mnner and B.Manderick (Eds.), Elsevier Publishing, 509-520, 1992.

[5] I.K. Nikolos, K.P. Valavanis, N.C. Tsourveloudis, & A.N. Kostaras, "Evolutionary algorithm based offline/online path planner for UAV navigation", Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 33(6), 898-912, 2003.

[6] U. Cekmez, M. Ozsiginan, M. Aydin, & O.K. Sahingoz, "UAV Path Planning with Parallel Genetic Algorithms on CUDA Architecture", WCE, IAENG, 2014.

[7] Y. Zhang, L. Wu, & S. Wang, "UCAV path planning based on FSCABC", Inf-an Int Interdiscip J, 14(3), 687-692, 2011.

[8] A. Colorni, M. Dorigo, & V. Maniezzo " Distributed Optimization by Ant Colonies", Proceedings of the First European Conference on Artificial Life, Paris, France, F.Varela and P.Bourgine (Eds.), Elsevier Publishing, 134-142, 1992.

[9] M. Dorigo, V. Maniezzo, & A. Colorni, "The Ant System: Optimization by a Colony of Cooperating Agents", IEEE Transactions on Systems, Man, and Cybernetics-Part B, 26(1):29-41, 1996.

[10] T. Sttzle, & M. Dorigo, "ACO Algorithms for the Traveling Salesman Problem", M. Makela, P. Neittaanmaki, J. Periaux, editors, Evolutionary Algorithms in Engineering and Computer Science, Wiley, 1999.

[11] C. Blum, & M. Dorigo, "Search bias in ant colony optimization: On the role of competition balanced systems", IEEE Trans. on Evolutionary Computation, Vol. 9, No. 2, pp. 159-174, 2005.

[12] KM. Sim, & WH. Sun, "Multiple ant colony optimization for load balancing", In: Intelligent Data Engineering and Automated Learning. Springer Berlin Heidelberg, p. 467-471, 2003.

[13] M. Middendorf, F. Reischle, & H. Schmeck, "Multi colony ant algorithms", Journal of Heuristics, 8.3: 305-320, 2002.

[14] M. Mavrovouniotis, S. Yang, & X. Yao, "Multi-colony ant algorithms for the dynamic travelling salesman problem", In: Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), IEEE Symposium on. IEEE, p. 9-16, 2014.

[15] F. Su, Y. Li, H. Peng, & L. Shen,"Multi-UCAV cooperative path planning using improved coevolutionary multi-ant-colony algorithm", In: Emerging Intelligent Computing Technology and Applications. Springer Berlin Heidelberg, p. 834-845, 2009.

[16] Y. Hao, S. Zhifeng , & Y. Zhao. "Path Planning for Aircraft Based on MAKLINK Graph Theory and Multi Colony Ant Algorithm", Computational Sciences and Optimization. CSO 2009. International Joint Conference on. Vol. 2. IEEE, 2009.

[17] A. Dussutour, S.C. Nicolis, G. Shephard, M. Beekman, D.J.T. Sumpter. "The role of multiple pheromones in food recruitment by ants", The Journal of Experimental Biology 212(4), 23372348, 2009.

[18] M. Middendorf, F. Reischle, & H. Schmeck. "Information exchange in multi colony ant algorithms", In Parallel and distributed processing pp. 645-652. Springer Berlin Heidelberg, 2000.

[19] H. Yueshun, D. Ping. "A Study of a New Multi-ant Colony Optimization Algorithm", Advances in Information Technology and Industry Applications, Volume 136, Lecture Notes in Electrical Engineering pp 155-161, 2012.

[20] G. Reinelt, "TSPLIBA traveling salesman problem library", *ORSA journal on computing*, 3(4), 376-384, 1991.