DACS3:Embedding Individual Ant Behavior in Ant Colony System

Zulaiha Ali Othman, Helmi Md Rais, and Abdul Razak Hamdan

***Abstract*—**Ants are fascinating creatures that demonstrate the ability to find food and bring it back to their nest. Their ability as a colony, to find paths to food sources has inspired the development of algorithms known as Ant Colony Systems (ACS). The principle of cooperation forms the backbone of such algorithms, commonly used to find solutions to problems such as the Traveling Salesman Problem (TSP). Ants communicate to each other through chemical substances called pheromones. Modeling individual ants' ability to manipulate this substance can help an ACS find the best solution. This paper introduces a Dynamic Ant Colony System with three-level updates (DACS3) that enhance an existing ACS. Experiments were conducted to observe single ant behavior in a colony of Malaysian House Red Ants. Such behavior was incorporated into the DACS3 algorithm. We benchmark the performance of DACS3 versus DACS on TSP instances ranging from 14 to 100 cities. The result shows that the DACS3 algorithm can achieve shorter distance in most cases and also performs considerably faster than DACS.

***Keywords*—**Dynamic Ant Colony System (DACS), Traveling Salesmen Problem (TSP), Optimization, Swarm Intelligent.

I. INTRODUCTION

TODAY'S business environment is increasingly complex and dynamic, with substantial flexibility required in operations. This is especially true of the logistics and transportation industry, where the need to deliver on time and to fulfill changes in customer requests makes it important to find the shortest paths for a delivery route. People, as we know, have a reduced ability to see the overall problem, particularly when the problem is relatively complex in term of its size and constraints. However, this inability can be overcome with the help of certain advanced optimization methods, which can aid humans in expediting the process.

The Ant Colony System (ACS) is the most successful algorithm used in combinatorial optimization problems such as the Traveling Salesman Problem (TSP), Vehicle Routing Problem (VRP), Job-Shop Scheduling Problem (JSP), and Quadratic Assignment Problem (QAP). The algorithm is inspired by the foraging behavior of a colony of ants, which, communicate through chemical substances called pheromones, acting as a memory preservation mechanism and providing guidance for ants in searching for shortest paths.

Zulaiha Ali Othman is with the Centre of Artificial Intelligent, Faculty of Information Science and Technology, National University of Malaysia (phone: 603-89216754; fax: 603-89216184; e-mail: zao@ftsm.ukm.my).

Helmi Md Rais is with the Department of System Science and Management, Faculty of Information Science and Technology, National University of Malaysia (e-mail: mubin7677@yahoo.com).

Abdul Razak Hamdan is with the Centre of Artificial Intelligent, Faculty of Information Science and Technology, National University of Malaysia (e-mail: arh@ftsm.ukm.my).

The principle of cooperation is the backbone of these algorithms. However, observing the behavior of a single ant can add value to the principle. Manipulating pheromone substances is a simple addition that can help to find the best solution. Therefore, many versions of ACS algorithms have been produced to find the shortest path by using the principle of cooperation among the ants. This study looks at individual ants' behavior in trying to reconnect paths previously laid by the colony when an obstacle is placed on such paths. Such blockages add another level of pheromone updates, which could contribute to faster optimum solutions.

This paper concentrates on an improvement of an ACS applied to the TSP domain, as first presented by Marco Dorigo [1]. The problem is to find the shortest tour of all the cities, where all cities are connected to each other, and visiting each city only once. This paper is divided into several sections. Section 2 discusses the previous experiments that generated the ACS concepts, along with the current observations of a single ant that inspired improvements in the ACS. Even though various versions of ACS have been invented which produce the shortest distance benchmark, the algorithm does not yet perform well for all datasets. Section 3 reviews the pass researches on ant colony algorithm. Section 4 presents the DACS3 model and its algorithm. Section 5 explains the setup for experimental comparison of the algorithms, and section 6 presents and discusses the results of DACS3 versus DACS.

II. ANT BEHAVIORS

Nature is a good source of solutions to problems faced by humans. Ants provide a good example for the case of transporting goods or finding shortest paths. Ants are social insects which cooperate through group communication, laying down chemical substances called pheromones to mark locations that have already been visited. The pheromones also serve as a reference for the return route back to their nest. These pheromones are then used by other ants as an indicator of the best path between the nest and food sources. The amount of pheromone laid determines whether the path is desirable to be taken by others; higher pheromone levels indicate more desirable routes.

Research on social insects began in the early nineteenth century, when the South African scientist Eugene Marais focused his attention on the behavior of termites, which he refers to as White Ants in his published work. His work was then picked up by Konrad Lorenz in his studies of imprinting and instinctive behavior of termites. Although both scientists are considered to be pioneers in the field of Ethology, the scientific studies of animal behavior, they were unaware of the actual mechanics of termite communication [22]. The answer to this question was discovered in the 1940s and 1950s when a French Entomologist named Pierre Paul Grasse investigated how termites communicate. He discovered that social insects react to significant stimuli, a signal which activates genetically encoded reactions called "stigmergy", which was a type of indirect communication that "workers stimulated by the performance they achieved" [9]. The characteristics of stigmergy were also found in J. L. Deneubourg et al's single and double bridge experiments on Argentine Ants, where they studied the pheromones laying and following behavior of ants [10].

Margo Dorigo et al. applied the results of these experiments to artificial ants, basing his work on ethologist studies that found ants established shortest paths based on pheromone trails. He took it one step further to study the random movement of ants, which he referred to as autocatalytic behavior, where ants move at random, detect an existing trail, decide to follow, and then re-enforce by laying down its own pheromones. Autocatalytic behavior is a process of positive and negative feedback, or pheromone reinforcement and evaporation, that causes very rapid convergence [1][8]. Figure 1 shows the experimental setup of Marco Dorigo, where an obstacle was placed in the path of multiple ants.

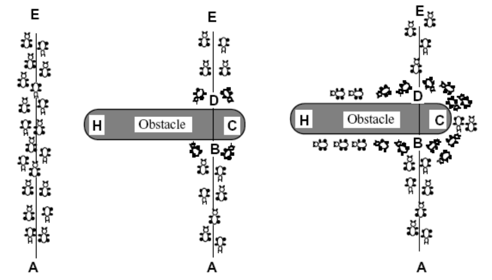

Fig. 1 Obstacle Experiment

The principle of cooperation among swarm insects is that communication among individuals contributes to the survival of the group as a whole. We have conducted several experiments to various colonies of "Malaysian House Red Ants" by using the experimental method of Marco Dorigo. An object or obstacle is placed on a single ant's normal path to find out how it behaves [23]. The experiment is conducted at a time when there are not many ants in the colony, normally late at night. Figure 2 shows ant behavior in constructing its paths. An ant travels along its normal path, Point A → Point B, following the strategic rules set out by the pheromones. When we put an obstacle in its path, the ant begins to search for an alternative route. It begins by examining both edges of the obstacle several times. Once it chooses a shorter edge to continue along, it begins to set up the path by visiting the shorter edge point d from point c, then tries to find a point e which returns to the existing trail. Once it does so, reinforces the newly constructed path by revisiting (c,d,e) several times.

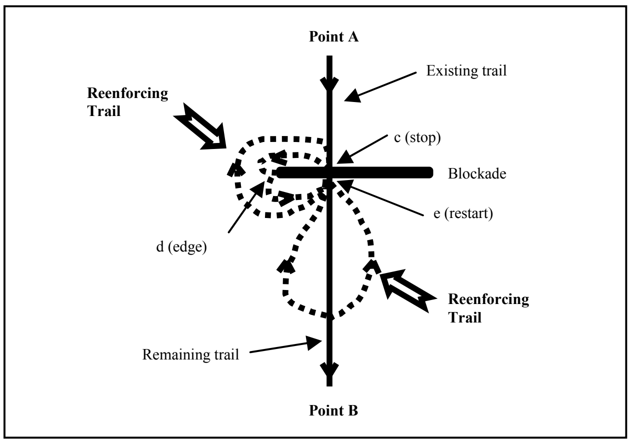

Fig. 2 Single Ant Obstacle Experiment

Once the new alternative path has received enough reinforcement, the ant then continues its journey using the path that existed before the obstruction. However, after some distance, it returns to the point c, where it restarts its journey and continues to reinforce the remaining path several times. From our observation, we concluded that to construct paths or a tour, there are three events involved: one event for path construction and two events for trail reinforcement.

III. Ant Colony Optimization

Ant System (AS) was first introduced and applied to TSP by Marco Dorigo et. al. [1], [2]. Initially, ants were placed on *n* cities and it moves from city *r* to city *s* using probabilistic formula called random-proportional rules [1] using Euclidean distance. After all ants have completed a tour, the pheromone level on all edges would be updated using local pheromone updating rule [2].

Later, Luca Maria Gambardella and his colleague have modified the AS algorithm and introduced ACS where it provides more balance and guidance in searching in three different ways. Firstly, the state transition rules (pseudo-random proportional action choice rules), provides a direct way to balance between an exploration of new edges and exploitation of a priori. Secondly, only edges that belong to the best ant tour being allowed to do pheromone updates through global pheromone updating rule and finally, local pheromone updating rule is applied while ants are trying to construct a solution [2], [6], [7]. Generally the basic idea was that the ants modified the pheromone level using local pheromone updating rule shown in (1) on the visited edges while constructing a solution after a series of choice selection on edges through decision making rule. After all ants have constructed a tour, it will then performed a second update using global pheromone updating rule [3] following edges that belong to only one best solution which produces the shortest tour from the beginning of the trial. L_{mn} is a tour length produced by nearest neighbour heuristic and n is the total number of cities.

$$\tau(r,s) \leftarrow (1-p).\tau(r,s) + p.\Delta\tau_0 \tag{1}$$

$$\text{where} \quad \Delta\tau_0 = (1/L_{mn}.n)$$

Stutzle and Hoos considers development done on ACS when developing MMAS but it is a direct improvement from AS [14]. MMAS is different in three ways [3], [21]. The first improvement is only one ant would update the pheromone which is the model of ACS but it could choose whether to update on solution of the current iteration or following the global best solution. Secondly, the pheromone strength was to be bounded to upper and lower limit [t_{max}, t_{min}] in order to avoid search stagnation. Lastly, the initial value for pheromone strength was initializing to t_{max} which was intended to provide a higher search exploration of solution at the beginning of the algorithm runs. The basic idea was that by allowing ants to update the pheromone level considering the solution on iteration to iteration basis (preferred choice) would guarantee more pheromone activities which constitutes an improve of searching performance. Nonetheless, the initialization of the pheromone level to the highest would encourage more exploration activities but it was limited to its boundaries where it would ensure that pheromone information is limited to its trails and not allows it to be too intensified or completely lost. When MMAS is close to convergence, one mechanism called pheromone trail smoothing (PTS) was used that helps increases pheromone trails proportionally to the maximum pheromone trails limit. The advantage of the mechanism is that, the information gathered during the run is not completely lost but merely weakened. This mechanism is interesting to be used when a long run is allowed.

Yi and Gong [3] have also proposed an algorithm which is a direct improvement of AS but considering improvement made to ACS and MMAS by introducing dynamic decay parameter to avoid the pheromone level growing too high and reaches local optima. With the theory that, pheromone evaporate quickly when it is intensified and less quickly when they are faint, the dynamic decay parameter was applied in both pheromone updates such as local pheromone updating rule and global pheromone updating rule [3]. They are also trying to accelerate the solution computation by allowing the best and worst tour done by ants to do pheromone update.

Beside the basic concepts, there are several strategies that have been adopted by ACO algorithms in order to find better solution quality and/or to achieve better performance such as candidate list, don't look bit and tour improvement heuristics.

IV. Dynamic ant Colony System 3 Level Updates (DACS3)

DACS3 considers the basic concepts introduced in ACS and DACS. However, we apply individual ant behavior as presented in Fig. 2, so DACS3 differs from previous systems in three ways. First, capturing all knowledge from the group and updating the pheromone level once the knowledge becomes available would expedite the process and increase the chances of finding a better solution. Second, a dynamic penalty on worst tours would open up chances for ants to navigate, limiting intentions and providing caution in an ant's decision to move. Finally, we get better search guidance by concentrating only on the best tours from all group performances, subdividing the group into two sections and then applying the global pheromone updating rule. Fig. 3 shows the workflow of the DACS3 model.

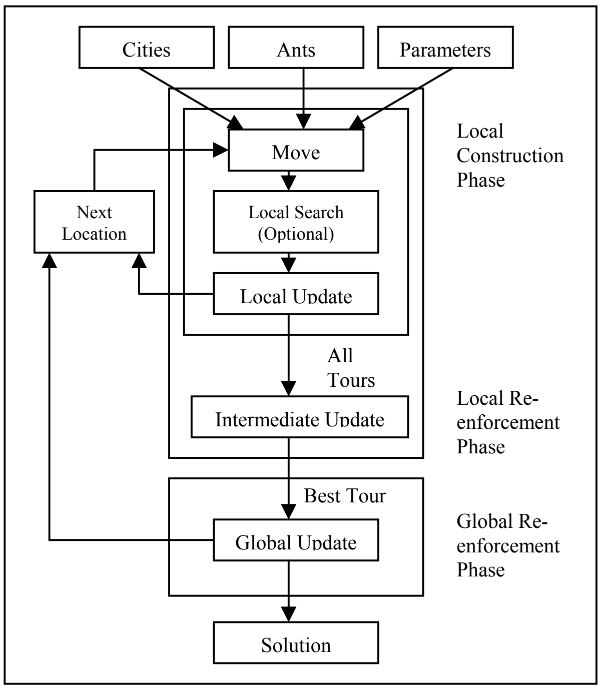

Fig. 3 DACS3 Diagram

In local construction phase, all cities shall be considered as a starting city r to start a tour computation. However, city r would not be considered as a visited city at the beginning of each tour construction. Thus, it constitutes a random start. Every ant would have to make a complete tour and the decision to choose which city s to move would be provided by state transition rules as use in [2]. Every time an ant visits a city s, it will modify the pheromone level by using local pheromone updating rule (1). The reason why we used ACS version of local pheromone updating rule because we believed that when an ant moves to a new location or ventures into the unknown territories, it would provides a constant pheromone deposit and evaporates at static state.

Once all ants in the group completed a tour, the available knowledge of every member of the group will then be used to modify the pheromone level using the intermediate pheromone updating rule (2) in local re-enforcement phase. The updates are necessary before all ants in the group been given a new task to complete. In this phase, the dynamic decay parameter will be used because it helps to alleviate an early stagnation or helps reducing the possibility of pheromone growing too high.

$$\tau(r,s) \leftarrow (1-[p.\tau(r,s)]).\tau(r,s)+\Delta C \qquad (2)$$

where

$$\Delta C = \begin{cases} [p.\Delta\tau(r,s)] & \text{if (r,s)} \in \text{ Local Group Best Tour} \\ -[p.\tau(r,s)].\Delta\tau(r,s) & \text{if (r,s)} \in \text{ Local Group Worst Tour} \\ 0 & \text{if (r,s)} \in \text{ Others} \end{cases}$$

and

$$\Delta\tau(r,s) = \begin{cases} (L_{grb})^{-1} \\ (L_{grw})^{-1} \end{cases}$$

All current completed tours in the group will be compared to the group best tour in the current iteration and the group worst tour from the beginning of trial. If no match found, the edges would experience normal dynamic evaporation. This method will appreciate every effort the ants make to produce the best tour but depreciate the worst tour from group performances. Dynamic penalty $[p.\tau(r,s)]$ is used to caution all ants of the bad paths when later it tries to make a decision to move on the next tour. L_{grb} is the total distance of the best tour in the current iteration and L_{grw} is the total distance of the worst tour of the group from the beginning of the trial.

The pheromone level once again will be modified using global pheromone updating rule (3) which is subdivided into two categories, best of the best and worst of the best. Only the best tour from the group performance will be considered for the pheromone update. If no match found, the edges would experience a normal dynamic evaporation. This method will provide better searching guidance in the effort to search for better solution.

$$\tau(r,s) \leftarrow (1-[p.\tau(r,s)]).\tau(r,s)+\Delta C \qquad (3)$$

where

$$\Delta C = \begin{cases} [p.\Delta\tau(r,s)] & \text{if (r,s)} \in \text{ Global Best Tour} \\ -[p.\tau(r,s)].\Delta\tau(r,s) & \text{if (r,s)} \in \text{ Global Worst Tour} \\ 0 & \text{if (r,s)} \in \text{ Others} \end{cases}$$

and

$$\Delta\tau(r,s) = \begin{cases} (L_{gb})^{-1} \\ (L_{gw})^{-1} \end{cases}$$

L_{gb} is the total distance of the globally best tour (best of the best) and L_{gw} is the total distance of globally worst tour (worst of the best) from the beginning of the trial. Fig. 4 shows how the DACS3 algorithm works. In this algorithm, we do not apply any additional strategic techniques or tour improvement heuristics.

```
GlobalBestTour = ∞;
GlobalWorstTour = 0;
LocalGroupWorstTour = 0;
Initialize pheromone level for all cities =$\tau_0$;
Generate initial solution using Nearest Neighbor (NN) heuristic;
CPU timer starts;
/* Trial begins */
Do
    /* Iteration begins */
    If i <= n
        LocalGroupBestTour = ∞;
        For k = 1 to m
            Start city = i;
            Do
                Select the next city j;
                /*Perform Local Pheromone Update*/
                Update trail level $\tau_{ij}$;                     (Rule (1))
            While (until all cities visited)
        EndFor
        /*Perform Intermediate Pheromone Update*/
        For k = 1 to m
            Compute tour distance;
            If (tour distance < LocalGroupBestTour)
                LocalGroupBest = current solution;
            Else if (tour distance > LocalGroupWorstTour)
                LocalGroupWorst = current solution;
            Else
                /*Pheromone updates for others*/
                Update trail level $\tau_{ij}$;                     (Rule (2))
            EndIf
            /*Pheromone updates for LocalGroupBest
            & LocalGroupWorst*/
            Update trail level $\tau_{ij}$ for LocalGroupBest;       (Rule (2))
            Update trail level $\tau_{ij}$ for LocalGroupWorst;      (Rule (2))
        EndFor
    EndIf
    /*Perform Global Pheromone Update*/
    Compute tour distance of LocalGroupBest
    If (tour distance < GlobalBestTour)
        GlobalBest = LocalGroupBest;
    Else if (tour distance > GlobalWorstTour)
        GlobalWorst = LocalGroupBest;
    Else
        /*Pheromone updates for others*/
        Update trail level $\tau_{ij}$;                             (Rule (3))
    EndIf
    /*Pheromone updates for GlobalBest & GlobalWorst*/
    Update trail level $\tau_{ij}$ for GlobalBest;                  (Rule (3))
    Update trail level $\tau_{ij}$ for GlobalWorst;                 (Rule (3))
While (until all termination statements satisfied)
```

Fig. 4 DACS3 Algorithm

V. The Experimental Setup

The algorithm was tested using several datasets taken from TSPLIB. The algorithm was developed using C language. Testing was performed on a machine with an Intel Core Duo 1.86GHz processor with 1 gigabyte of physical memory. The testing used a normal round up method ($\geq 0.5 = 1$) to calculate integer distances. This provides a more meaningful value than the bankers' roundup method. Several ant population sizes were tested in order to determine the best number of ants for the data. There are two types of datasets being tested: Euclidean distance (Oliver30, Berlin52, KroC100) and GEO distance (Burma14).

The experiments sought to determine which algorithms could reach optimal distance; if all tested algorithms were able to find it, then performance speed would be the second measurement. For comparison, the first column is the best distance from the beginning of the trail, as compared to the benchmark distance. The second column shows the number of iteration required to come up with the best distance. The third column is the best time obtained from 15 trials. The fourth column is an average time from 15 trials. Distance was measured by the integer distance (the roundup distance from each moves) and the real distance (in the bracket). The value is measured in Euclidean and GEO distances. Real distance was used as a measurement in calculating distances for Euclidean datasets, while integer distance was used as the basis of the distance calculation for GEO datasets. Table I shows the best parameter settings for the DACS3 experiment, which is a similar setup from previous work on the ACS and DACS algorithms [1] [2] [3] [6] [7].

TABLE I
DACS3 PARAMETERS SETTING

Parameters	Value
Ants Population Size (m)	10
q_0	0.9
β	2
p	0.1
Max Iterations	10000

VI. Result and Discussion

Table IIA and IIB shows the experimental result of DACS3 compared to DACS. DACS3 has reached the optimal benchmark distance for most of the case studies accept for KroC100, but the difference is very small (0.6%). DACS only reach the optimal benchmark distance for Oliver30 and Burma14 data. DACS produces 4.4% and 2.1% longer of shortest distance for Berlin52 and KroC100 datasets respectively. In term of performance time to get solution for those algorithms which achieved the same optimal distance, DACS3 has performed 90% and 75% faster compare to DACS for Oliver30 and Burma14.

TABLE IIA
DACS3 RESULT COMPARISON TO BENCHMARK

TSP Problem	Bench-mark Distance	**DACS3 (β=2, q_0=0.9, p=0.1) – m=10** Best Distance	Best Iteration	Best Time (Sec)	Average Time (Sec)
Oliver30 (30-city problem)	420 (423.74)	420 (423.74)	168	7.000	**7.043**
Berlin52 (52-city problem)	7542 (N/A)	7542 (7544.37)	6447	1278.407	1278.590
KroC100 (100-city problem)	20749 (N/A)	20880 (20881.61)	7954	10854.407	10856.385
Burma14 (14-city problem)	3323 (N/A)	3323 (3330.61)	94	0.953	**1.003**

TABLE IIB
DACS RESULT COMPARISON TO BENCHMARK

TSP Problem	Bench mark Distance	**DACS (β=2, q_0=0.9, p=0.1) – m=10** Best Distance	Best Iteration	Best Time (Sec)	Average Time (Sec)
Oliver30 (30-city problem)	420 (423.74)	1826	71.484	71.582	420 (423.74)
Berlin52 (52-city problem)	7881 (7880.78)	3661	694.594	694.835	7881 (7880.78)
KroC100 (100-city problem)	21190 (21191.37)	1720	2240.984	2242.229	21190 (21191.37)
Burma14 (14-city problem)	3323 (3330.61)	527	4.766	4.823	3323 (3330.61)

The result shows that DACS3 has produced shorter distance for all case studies but perform a bit longer time to get solution for bigger data. However, DACS3 outperformed DACS algorithm in all cases.

Fig 5(a)-(d) shows the log graphs of shortest distance versus iterations. The comparison is done between DACS3 and DACS for datasets Burma14, Oliver30, Berlin52 and KroC100. DACS3 has obtained good searching performance when small size problem (Burma14 and Oliver30) involved. This is shown by the fact that DACS3 has outperforms other algorithms throughout the run. However, when DACS3 searches for solutions for slightly bigger problems (Berlin50 and KroC100), it performs poorly at the beginning of the search but produces good performance in the middle and end of the run.

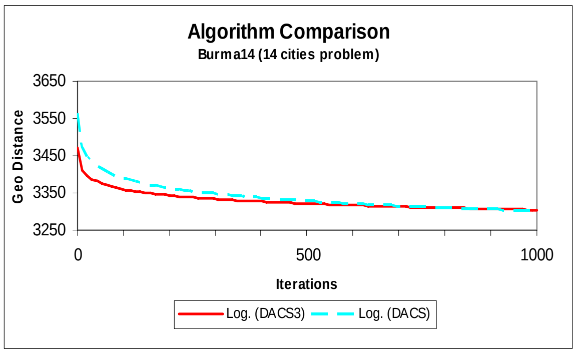

Fig.5 (a) Log graph algorithm comparison for Burma14 problem

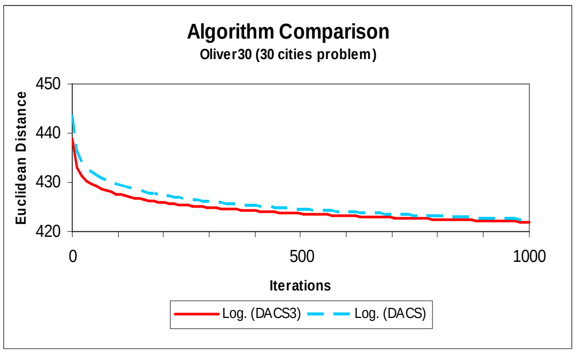

Fig. 5(b) Log graph algorithm comparison for Oliver30 problem

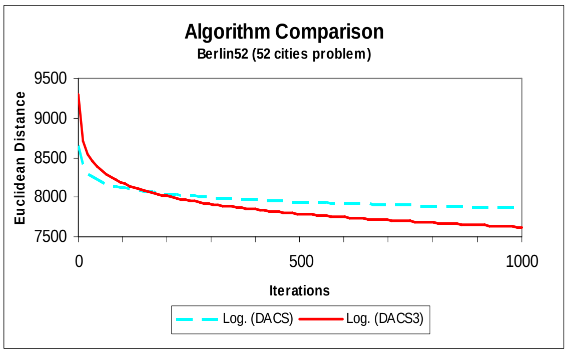

Fig. 5(c) Log graph algorithm comparison for Berlin52 problem

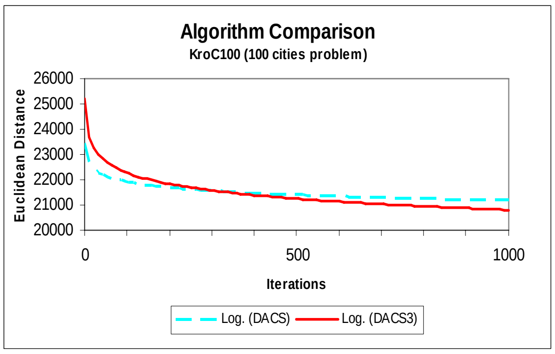

Fig. 5 (d) Log graph algorithm comparison for KroC100 problem

VII. Conclusion and Further Research

Based on the result stated above, we can concludes that manipulating and empowering the available knowledge of the individuals can provide a significant advantage in order to solve the whole perspective of the problem. Harnessing experiences of every single individual and in this case, the ants, can expedite the process of finding a good or better solution either shorter distance or/and performance time to get solution. Based on the graph we can see that DACS3 has an outstanding searching performance for smaller data but slightly poor at the beginning of the search for bigger data but outperformed every algorithm at the middle of the run. Therefore, the next step is to optimise the iteration process in DACS3 using various strategic techniques such as candidate list, pheromone trails smoothing (PTS), and the elitist ant concepts.

References

[1] M. Dorigo, V. Maniezzo, and A. Colorni, The Ant System: Optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man and Cybernatics-Part B*, vol. 26, no. 1, pp.1-13, 1996.

[2] M. Dorigo, and L. M. Gambardella, Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem, *IEEE Transaction of Evolutionary Computation*, vol.1, no 1, pp.53-66, 1997.

[3] Y. Li, and S. Gong, Dynamic ant colony optimization for TSP, Int J Adv Manuf Technol, vol. 22, pp. 528-533, July 2003.

[4] L. M. Gambardella, E. Taillard, and G. Agazzi, MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows, Technical Report IDSIA (IDSIA-06-99), 1999.

[5] R. Montemanni, L. M. Gambardella, A. E. Rizzoli, and A. V. Donati, Ant colony system for a dynamic vehicle routing problem, *Journal of Combinatorial Optimization*, vol. 10, no. 4, pp. 327-343, December 2005.

[6] M. Dorigo, and K. Socha, An Introduction to Ant Colony Optimization, Book Chapter in Approximation Algorithms and Metaheuristic, CRC Press 2007.

[7] M. Dorigo, M. Birattari, and T. Stutzle, Ant Colony Optimization – Artificial Ants as a Computational Intelligence Technique, *IEEE Computation Intelligent Magazine*, 2006.

[8] M. Dorigo, E. Bonabeau, and G. Theraulaz, Ant algorithms and stigmergy, *Journal of Future Generation Computer Systems*, vol. 16, pp. 851 – 871, 2000.

[9] M. Dorigo, and K. Socha, An Introduction to Ant Colony Optimization, Technical Report IRIDIA 2006-010, April 2006.

[10] J-L. Deneubourg, S. Aron, S. Goss, and J.M. Pasteels, The self-organizing exploratory pattern of Argentine Ant, *Journal of Insect Behavior*, vol. 3, pp. 59-168, 1990.

[11] M. Dorigo, G. Di Caro and L. M. Gambardella, Ant Algorithms for Discrete Optimization. *Journal of Artificial Life*, vol. 5, no. 2x, pp. 137-172, 1990.

[12] C. Anderson, and N. R. Franks, Teams in animal societies, *Journal of Behavioral Ecology*, vol. 12, no. 5, pp. 534-540, September 2000.

[13] L.M. Gambardella, and M. Dorigo, Ant-Q: A Reinforcement Learning Approach to the Traveling Salesman Problem. Proceedings of ML-95, Twelfth International Conference on Machine Learning, 1995, pp. 252-260.

[14] T. Stützle, and H. H. Hoos, MAX-MIN Ant System, *Journal of Future Generation Computer Systems*, vol. 16, no. 8, pp. 889-914, 2000.

[15] M. Fleischer, Foundation of Swarm Intelligence: From Principles to Practice, Conference on Swarming: Network Enabled C4ISR, January 2003.

[16] M. Dorigo, and G. Di Caro, The Ant Colony Optimization Meta-Heuristic, In D. Corne, M. Dorigo and F. Glover, editors, New Ideas in Optimization, McGraw-Hill, 1999, pp. 11-32.

[17] M. Gendreau, and J. Y. Potvin, Metaheuristic in Combinatorial Optimization, Annals of Operation Research, vol. 140, pp. 189-213, 2005.

[18] B. Fox, W. Xiang, and H. P. Lee, Industrial applications of the ant colony optimization algorithm, Int J Adv Manuf Technol, July 2005.

[19] E. Bonabeau, and C. Meyer, Swarm Intelligence: A Whole New Way to Think about Business, Harvard Business Review, May 2001.

[20] D. Gaertner, and K. Clark, On Optimal Parameters for Ant Colony Optimization algorithms, Proceedings of International Conference on Artificial.

[21] P. E. Merloti, Optimization Algorithms Inspired by Biological Ants and Swarm Behavior, San Diego State University, Artificial Intelligence Technical Report, CS550, San Diego, June 2004.

[22] C. Grosan, and A. Abraham, Stigmergic Optimization: Inspiration, Technologies and Perspectives. Studies in Computational Intelligence Journal, vol. 31, Springer Publishing 2006.

[23] H. Md Rais, Z.A. Othman and A. R. Hamdan, Improved Dynamic Ant Colony System (DACS) on Symetric Traveling Salesman Problem (TSP), International Conference on Intelligent and Advanced System (ICIAS 2007), November 2007.