International Journal on Recent and Innovation Trends in Computing and Communication ISSN 2321 – 8169
Volume: 1 Issue: 2 75 – 80

A Preliminary Study for Ant Colony Optimization with a new Reinforcement Strategy

Issmail M. Ellabib
Computer Engineering Department
University of Tripoli
Tripoli, Libya
i_ellabib@yahoo.com

Ahmed Ab. Arara
Computer Engineering Department
University of Tripoli
Tripoli, Libya
ahmedarara@yahoo.fr

***Abstract*—Different Ants Colony Optimization (ACO) algorithms use pheromone information differently in an attempt to improve their relative performance. In this paper, we describe a new systematic reinforcement strategy as a means to improve the pheromone update rules of existing ACO algorithms. We examine the proposed strategy and compare it with other improvement strategies using the well-known Traveling Salesman Problem (TSP). The results indicate that the performance of both the Ant System (AS) and the Ant Colony System (ACS) algorithms is improved by applying the proposed strategy. We postulate that the proposed strategy allows the ants, in some sense, to both refine the search in promising regions, and escape explored areas of the search space more consistently and effectively than other reinforcement strategies.**

***Index Terms*— Ant Colony Optimization, combinatorial optimization problems, Meta-Heuristics, Traveling Salesman Problem.**

*****

I. INTRODUCTION

Various models of natural swarm intelligence that have been transformed into useful artificial swarm intelligence algorithms have been appeared in the literature [2]. A number of these algorithms are designed to mimic the foraging behavior of ant colonies and are classified as Ant Colony Optimization (ACO) algorithms. ACO algorithms have been applied to many hard combinatorial optimization problems [1]. Dorigo et al. in [4] originated the concept of applying the underlying properties of the ant colony behavior to create a new computational approach, called Ant System (AS) algorithm, to solve combinatorial optimization problems. Performance on the Traveling Salesman Problem (TSP) indicated that the AS algorithm did not scale well to larger instances. Subsequently, several modifications were introduced to improve its performance. The first improvement was in the structure of the AS pheromone update rule. An elitist reinforcement strategy, inspired from genetic algorithms (GA), was introduced in the AS pheromone update rule [4]. The purpose of this strategy was to give a strong additional reinforcement to the edges belonging to the current best tour since these edges were assumed to be part of the optimal solution. In [3] a different reinforcement strategy, called the ranking strategy, was introduced into the AS pheromone update rule. In this strategy, the pheromone trails of the best tour are updated in a manner similar to the elitist strategy. In addition, the ants (i.e. agents) are ranked based on their solution quality and a predetermined number of highest ranked ants are allowed to update the pheromone trails of their tours. In [10] Stutzle et al. introduced the MAX-MIN Ant System (MMAS). The MMAS is different from the AS in three ways. Firstly, only the ant that generates the best tour is allowed to deposit pheromone on its trail. Secondly, the pheromone level on any edge is limited to a given range. Finally, the pheromone levels on each edge are initialized to their maximum allowable value. A smoothing mechanism for the pheromone trail and a modified 3-opt local search was also introduced in MMAS in order to improve performance. Dorigo and Gambardella in [5] introduced an Ant Colony System (ACS) to improve the performance of the AS for solving the TSP. This ACS approach is based on their earlier Ant-Q algorithm [5]. ACS and Ant-Q differ in the local pheromone update rule. Ant-Q uses a specific type of reinforcement learning mechanism called Q-learning. ACS and Ant-Q both adopt a new transition rule in which a given parameter is identifying the relative importance of exploitation versus exploration in the search process.

With the notable exception of the ranking strategy introduced in [3], the reinforcement strategies applied in pheromone update rules of ACO algorithms are most often designed to force the ants in future iterations to favor tours in the vicinity of the best tour found thus far or the best tour found in the current iteration. In this paper, we introduce and describe a new reinforcement strategy for the pheromone update rule of both the AS and the ACS algorithms designed to force the ants in future iterations to favor tours in the vicinity of the current high quality tours. We use an adaptive weighting scheme that weighs the contribution of the ants based on their quality and state of convergence.

The paper is organized as follows. In section II we discuss ACO algorithms including the different reinforcement strategies employed in the pheromone update rule. We then describe the methodology of the proposed reinforcement strategy in section III and introduce it to the pheromone update rule of both AS and ACS algorithms. Experimental results of the AS and the ACS algorithms employing this weighting strategy are presented in section IV along with a comparative performance analysis involving other AS-based approaches. Finally, section V provides some concluding remarks.

II. ACO ALGORITHMS

In general, the basic idea behind ACO algorithms is to simulate the foraging behavior of a large number of real ants using simple artificial ants working as co-operative agents to generate high quality solutions to a combinatorial optimization problem via interaction between the ants and their environment. Artificial pheromone is accumulated during the construction phase of ACO algorithms through a learning mechanism implied in the algorithm's pheromone update rules. Artificial ants move from one node to another on a representation graph using a transition rule that favors shorter edges and edges with greater amounts of pheromone. They update the pheromone trail of their generated tours based on a pheromone update rule. This rule often deposits a quantity of pheromone proportional to the quality (or length) of the corresponding tour. Most ACO algorithms were initially designed to solve the TSP and then extended to other combinatorial optimization problems. The general framework of ACO algorithms for solving combinatorial optimization problems is outlined in Figure 1.

```
Initialize: Initial pheromone trails
While (termination condition not met) do
        Construct & update: ants construct solutions based on transition rule and
        update the pheromone trails locally
        Improve: solutions can be improved by local search procedure
        Evaluate: solutions are evaluated
        Global update: pheromone trails updated globally
End
```

Figure 1. General framework of ACO algorithms.

A. Ant System (AS) algorithm

An artificial ant in an AS algorithm moves from city to city on a TSP graph based on a probabilistic transition rule; each ant starts from a randomly chosen city, and moves to an unvisited city based on this rule until all the cities have been visited.

The pheromone update rule used in the first AS algorithm (ant-cycle model) **[4]** is given by

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) \qquad (1)$$

where ρ, the evaporation rate or the persistent factor, is a parameter in the range $0 \le \rho \le 1$ that regulates the reduction of pheromone on the edges. $\Delta\tau_{ij}(t)$, the total amount of pheromone added by all ants on the edge joining cities i and j, is given by

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{m} \Delta\tau^{k}_{ij}(t) \qquad (2)$$

where $\Delta\tau^{k}_{ij}(t)$, the amount of pheromone added by ant k, is given by

$$\Delta\tau^{k}_{ij}(t) = \begin{cases} Q/L^k & \text{if } edge(i,j) \in T^k \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

where Q is a parameter that represents the quantity of the pheromone laid by each ant, and L^k is the length of tour T^k found by ant k.

The evaporation rate ρ reduces the amount of pheromone on each edge of a tour based on the quality of that tour. After a few iterations, the pheromone update rule (Equation 1) increases the amount of pheromone on edges belonging to high quality solutions and decreases the amount of pheromone on edges belonging to low quality solutions. Consequently, edges belonging to the high quality solutions have a greater chance of being selected in the next iterations. Reinforcement strategies such as the elitist strategy and the ranking strategy are introduced later in order to improve the performance of the pheromone update rule. In what follows we provide a summary of these two strategies.

1) Elitist strategy

The aim of this strategy is to intensify the search in the next iteration (t+1) within the neighborhood of the best tour found so far by adding an extra amount of pheromone to the edges in this best tour using the following pheromone update rule

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) + e\,\Delta\tau^{e}_{ij}(t) \qquad (4)$$

where the amount added (the last term in Equation **Error! Reference source not found.**) is the product of e, a given number of elitist ants, and $\Delta\tau^{e}_{ij}(t)$, the quantity of pheromone added to the edges in the best tour (T^b). This quantity is a function of the best tour length (L^b) and is given by

$$\Delta\tau^{e}_{ij}(t) = \begin{cases} Q/L^b & \text{if } edge(i,j) \in T^b \\ 0 & \text{otherwise} \end{cases} \qquad (5)$$

2) Ranking strategy

In this strategy, the pheromone on the edges of the best tour is updated in a manner similar to the elitist strategy as in Equation 4. In addition, the ants are ranked based on the quality of their solution and a number of the highest ranked ants are allowed to update the pheromone on the edges of their tours. Thus $\Delta\tau_{ij}(t)$ term in Equation 4 is redefined to be

$$\Delta\tau_{ij}(t) = \sum_{\mu=1}^{e-1} \Delta\tau^{\mu}_{ij}(t) \qquad (6)$$

where the index μ ranges ranked ants. $\Delta\tau^{\mu}_{ij}(t)$, the amount of pheromone added to the tour (T^{μ}) of ant μ, is given by

$$\Delta\tau^{\mu}{}_{ij}(t) = \begin{cases} (e-\mu)\dfrac{Q}{L^{\mu}} & if \;\; edge(i,j) \in T^{\mu} \\ 0 & otherwise \end{cases} \quad (7)$$

B. MAX-MIN Ant System (MMAS)

The MMAS is based on the following three modifications of the original AS algorithm:

1) The ant with the best tour in each iteration t is allowed to deposit amount of pheromone on its tour T^b using the modified pheromone update rule given by

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \Delta\tau_{ij}^{b}(t) \; if \;\; edge(i,j) \in T^{b} \quad (8)$$

where the amount added by the ant having the best tour is given by

$$\Delta\tau^{b}{}_{ij}(t) = 1/L^{b} \quad (9)$$

2) To avoid stagnation, the range of pheromone levels on any edge is bounded by $\tau_{\min} \le \tau_{ij}(t) \le \tau_{\max}$

3) The pheromone levels are initialized to their maximum allowable value (τ_{max}) to encourage a higher degree of path exploration at the start of the algorithm.

For more details on MMAS algorithm the reader may refer to Stutzle et al. **[10]**.

C. Ant Colony System (ACS)

The ACS approach features two major changes to the original AS algorithm:

1) A new transition rule is introduced that favors either exploitation or exploration.
2) The pheromone is updated in two different ways:

Local updating: As the ant moves between cities i and j, it updates the amount of pheromone on the traversed edge using the following formula:

$$\tau_{ij}(t) = (1-\rho)\tau_{ij}(t-1) + \rho\tau_0 \quad (1)$$

where ρ is a pheromone decay parameter for local updating τ_0 is the initial amount of pheromone on all edges and is calculated as $\tau_0=(n\ L^h)^{-1}$, n is the number of cities and L^h is the length of the tour produced by the nearest neighbor heuristic.

Global updating: When all ants have completed their tours the ant that found the best tour updates the edges belonging to its tour using the following formula:

$$\tau_{ij}(t) = (1-\alpha)\tau_{ij}(t-1) + \alpha(1/\mathrm{L}^{b}) \quad (2)$$

where L^b is the length of the best tour found so far and α is a global updating decay parameter.

Local updating diminishes the amount of pheromone, indirectly favoring the exploration of unvisited edges, whereas global updating encourages the ants in future generations to search in the vicinity of the best tour. For more details on ACS the reader may refer to Bonabeau et al. **[2]** and Dorigo et al. **[5]**.

III. A NEW REINFORCEMENT STRATEGY

In the previous section we outlined examples of different representations of pheromone information involving reinforcement learning strategies that have been previously proposed in order to improve the performance of ACO algorithms. An efficient strategy should improve the performance in which the search for high quality solution is guided by a probable representation of the pheromone information. The underlying assumption in this paper is that in the neighborhood of high quality solutions there may be further high quality solutions. Thus, we propose to modify the pheromone update rule to reinforce the search for tours in the vicinity of high quality tours.

To this end we introduce a new reinforcement strategy based on a weighting scheme. Virtual ants are weighted based on the quality of their solutions as well as on the current convergence state of the population. Accordingly, the pheromone trails of their tours are updated adaptively in response to their weights. An extra amount of pheromone is deposited on the edges of the high quality tours based on both the quality of generated tours and the current state of convergence. In order for this to work, it is essential to identify whether the population is converging toward one solution or scattered in the search space. One possible way of detecting convergence is to calculate the difference between the current average length of tours and the length of the best tour found so far. The difference between the average length of the tours L^{avg} of a population and the best tour length L^b found so far is likely to be less for a population that has converged to a local optimum solution than that for a population scattered in the search space. We have observed this property in our experiments with the AS algorithm and Figure 2 illustrates this property for a typical case in the original AS algorithm. We therefore used the difference in average and best tour length (L^{avg} - L^b) as a yardstick for detecting convergence. Ants with tours that were shorter than the population average were assigned a weight, in the range (0,1], favoring ants with better tours.

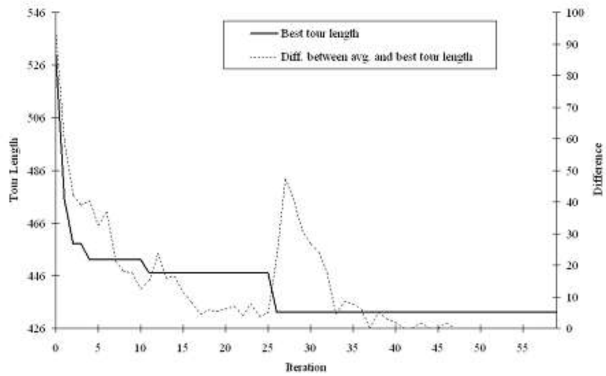


Figure 2. Variation of the difference between population tour length average and best tour length with the progress of the best tour length found (Test problem: eil51).

The ants with tours that were longer than the population average were assigned a weight of zero. These weights were calculated as follows:

$$w^k = \begin{cases} \dfrac{L^{avg} - L^k}{L^{avg} - L^b} & if\ \{L^k < L^{avg}\} \\ 0 & otherwise \end{cases}, \qquad (3)$$

This weighting value depends not only on the measure of convergence but also on the difference between the length of ant *k's* tour, L^k, and the length of the best tour, L^b: the closer L^k is to L^b the closer w^k is to 1. We aim at achieving a trade-off between exploration and exploitation by incorporating this weight in the pheromone update rule of both AS and ACS algorithms. The proposed approach will adjust the quantity of pheromone added to edges iteratively to account for solution quality and the current state of convergence.

A. Ant System employing the weighting strategy (AS_w)

In the standard AS algorithm, Equation **Error! Reference source not found.** defines the amount of pheromone added by ant *k* in iteration *t*. We redefine this amount to incorporate the proposed weighting strategy:

$$\Delta\tau_{ij}^k(t) = \begin{cases} (1+w^k)\dfrac{Q}{L^k} & if\ edge(i,j) \in T^k \\ 0 & otherwise \end{cases}, \qquad (4)$$

where $w^k Q / L^k$ is the amount of pheromone added to the edges belonging to the high quality solutions.

In order to study the effect of the proposed weighting strategy, we implemented the AS algorithm employing both the elitist (AS_e) and the weighting strategy (AS_w) and applied both approaches to test problem eil51. The experiments were conducted with the following parameters: $\alpha=1$, ($\beta=4$ for AS_w and $\beta=5$ for AS_e), $Q=100$, $\tau_0=0.000001$, $\rho=0.5$, $m=n$. In the first case, we set *e* to *5* as proposed by Bonabeau et al. [2]. In the second case, we set *e* to *n* as proposed by Stutzle et al. [10]. In each run, the progress of the best tour length over 10,000 iterations is reported. Figure 3 depicts the progress of the average best tour length over 10 independent runs of both the AS_e and the AS_w. The figure illustrates an improvement in the performance of AS employing the weighting strategy over that employing the elitist strategy.

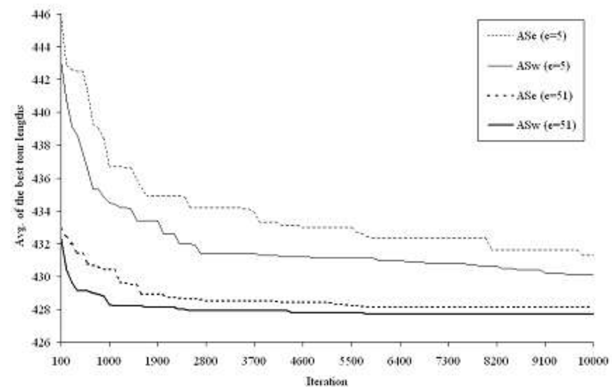
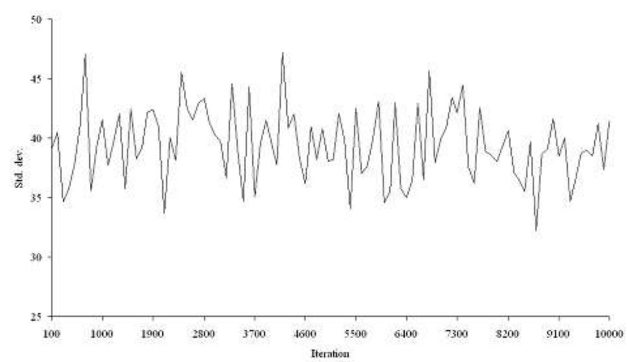


Figure 3. Evolution of the average best tour lengths of both AS_e and AS_w over 10 runs (Test problem:eil51).

We tested the ability of AS_w to explore more solutions and to direct the search away from local minima. We monitored the standard deviation of the population's tour lengths for all experiments. Figure 4 depicts the evolution of the standard deviation over 10,000 iterations. In this graph, the evolution of the standard deviation suggests that different solutions are persistently discovered despite the extra amount of pheromone added to some edges.



Figue 4. Evolution of the standard deviation of the population's tour lengths of the AS_w (Test problem:eil51).

B. Ant Colony System employing the weighting strategy (ACS_w)

In the standard ACS algorithm, the global updating (defined in Equation 2) deposits an extra amount of pheromone to the edges belonging to the best solution. This equation can be modified so that edges belonging to high quality tours receive an additional quantity of pheromone, as a result of incorporating the proposed weighting strategy.

For the ACS employing the weighting strategy, we redefine global updating as follows:

$$\tau_{ij}(t)=\begin{cases}(1-\alpha)\tau_{ij}(t-1)+\alpha(w^k/L^b) & \text{if } edge(i,j)\in T^k\\ (1-\alpha)\tau_{ij}(t-1)+\alpha(1/L^b) & \text{if } edge(i,j)\in T^b\end{cases},\quad (5)$$

In order to study the effect of the proposed weighting strategy, we implemented the ACS algorithm employing the weighting strategy (ACS_w). The experiments were conducted with the following parameters: For the standard ACS algorithm, $\beta=2$, $\rho=0.1$, $\alpha=0.1$, $q_0=0.9$, and $m=10$ *ants* as proposed by Bonabeau et al. [2]. For ACS_w, $\beta=3$, $\rho=0.1$, $\alpha=0.94$, $q_0=0.93$, and $m=10$ *ants*. The progress of the best tour length over 51,000 iterations is reported in each run. Figure 5 depicts the progress of the average best tour length over 25 independent runs of both ACS and ACS_w. The figure illustrates an improvement in the performance of ACS employing the weighting strategy over the standard ACS.

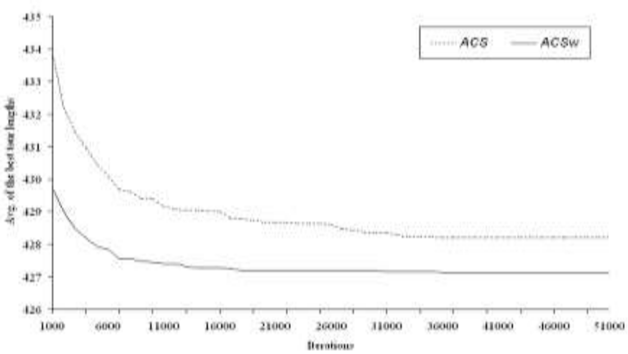


Figure 5. Evolution of the average best tour lengths of both the ACS and the ACS_w over 25 runs. (Test problem:eil51 problem).

In the same manner, as with the AS algorithm, we tested the ability of ACS_w to explore the solution space and to direct the search away from local minima. We monitored the standard deviation of the population's tour lengths for all our experiments. Figure 6 depicts the evolution of the standard deviation over 51,000 iterations. In this graph, the evolution of the standard deviation suggests that different solutions are persistently discovered despite the extra amount of pheromone added to some edges.

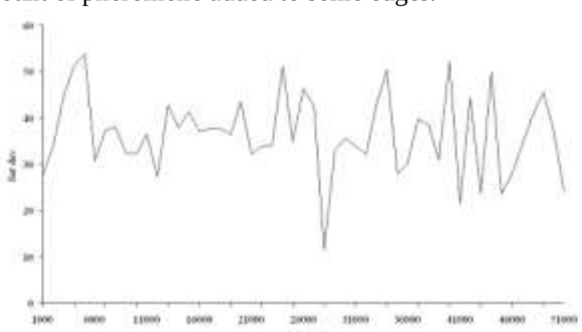


Figure 6. Evolution of the standard deviation of the population's tour lengths of the ACS_w. Typical run of eil51 problem.

IV. COMPUTATIONAL EXPERIMENTS

In this section, we present some computational results obtained by both AS and ACS algorithms employing the proposed weighting strategies. We compare our results with those obtained from other AS-based approaches. For the test problem sets, we consider different sizes of Symmetric Traveling Salesman Problem (STSP) and Asymmetric Traveling Salesman Problem (ATSP) instances found in the library TSPLIB: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/. The code was written in the C++ language.

1) Comparison between AS-based approaches

We present some computational results obtained by the AS algorithm employing weighting strategies (AS_w). We compared, for different problem instances taken from TSPLIB, the performance of AS_w against the following AS-based approaches reported in [10]: Ant System (AS), Ant System employing ranking strategy (AS_{rank}), Ant System employing elitist strategy (AS_e). The computational results given in Table I demonstrate the competitive performance of the proposed improvements on AS when compared to other approaches. For the symmetric instances (problems ei151, kroA100 and d198), the proposed strategies prove to be effective, and better than any of the other approaches. For the asymmetric instances (problems ry48p, ft70, kro124p and ftv170), AS_{rank} outperformed all other strategies on each of the instances except ft70. However, the computational results in [5] indicate that the optimal solutions for the ATSP instances were reached all the time by ACO algorithm employing a local search procedure except for the ft70 instance, a problem considered relatively hard [5]. The proposed approach outperformed other approaches in this problem.

TABLE I
PERFORMANCE OF THE DIFFERENT AS BASED APPROACHES FOR DIFFERENT TSP INSTANCES. AVERAGE OF THE BEST TOUR LENGTHS OVER 25 INDEPENDENT RUNS

Instance	**Opt.**	**AS**	**AS_e**	**AS_{rank}**	**AS_w**
eil51	426.0	437.3	428.3	434.5	**427.6**
kroA100	21282.0	22471.4	21522.8	21746.0	**21499.0**
d198	15780.0	16702.1	16205.0	16199.1	**16063.4**
ry48p	14422.0	15296.4	14685.2	**14511.4**	14719.5
ft70	38673.0	39596.3	39261.8	39410.1	**39231.5**
kro124p	36230.0	38733.1	37510.2	**36.973.5**	37899.8
ftv170	2755.0	3154.5	2952.4	**2854.2**	2931.3

2) Comparison between the most recent AS based approaches

We present some computational results obtained by the ACS algorithm employing the weighting strategy (ACS_w). We compare the performance of the proposed approach against the standard ACS algorithm as well as MMAS algorithm.

Table I compares the performance, over 25 independent runs, of the standard ACS reported in [6], MMAS obtained from [9] and our implementation of ACS_w. The computational results demonstrate that the proposed weighting strategy improves on the performance of the standard ACS algorithm and competes favorably with MMAS.

TABLE I
PERFORMANCE OF ACS BASED APPROACHES AND MMAS FOR STSP AND ATSP INSTANCES

Instance	**Opt.**	**MMAS**	**ACS**	**ACS_w**
eil51	426.0	427.6	428.1	**427.0**
kroA100	21282.0	**21230.3**	21420.0	21334.5
d198	15780.0	15972.5	16054.0	**15939.9**
ry48p	14422.0	14553.2	14565.4	**14485.3**
ft70	38673.0	39040.2	39099.0	**39040.1**
kro124p	36230.0	**36773.5**	36857.0	36795.8
ftv170	2755.0	2828.8	**2826.5**	2844.6

In general, the test results indicate that the ACS algorithm employing the proposed weighting strategy is among the best AS-based approaches for finding high quality solutions for such problem instances. Moreover, the results indicate that the weighting strategy consistently finds high quality solutions. To obtain results competitive with the best performing algorithms, a local search procedure has to be embedded in the ACO algorithms as introduced by Dorigo et al. in [5] and Stutzle et al. in [10].

V. CONCLUSION AND FURTHER WORK

In this paper we described a method of incorporating a new reinforcement strategy into the pheromone update rule of both the original (Ant System) and the most recent (Ant Colony System) ACO algorithms. We have chosen one particular strategy of adapting the amount of pheromone based on current solutions found by the population of ants. The strategy weighs all ants based on the quality of their tours and their state of convergence.

Experimental results suggest that the proposed approach is successful in terms of balancing convergence goals with exploration goals. Using some TSP instances from TSPLIB as a test bed, the AS algorithm employing the proposed weighting strategy yields considerable improvement in performance compared to AS approaches attempting other reinforcement strategies. Moreover, the ACS algorithm employing the weighting strategy outperformed the standard ACS algorithm and the MMAS algorithm in some instances and remained competitive on all other instances.

One feature of the proposed approach over the other AS-based approaches is the fact that the high quality solutions found are evaluated in terms of quality and state of convergence to direct the search toward promising regions. Future work should be directed at developing such strategies and employing them in the ACO algorithms to solve other combinatorial optimization problems.

REFERENCES

[1] M. Dorigo and T. Stũtzle. Ant Colony Optimization. Massachusetts Institute of Technology, 2004.

[2] E. Bonabeau, M. Dorigo, and G. Theraulaz. Swarm Intelligence: From Natural to Artificial systems, New York: Oxford University Press, 1999

[3] B. Bullnheimer, R. Hartl, and C. Strauss. 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.

[4] M. Dorigo, V. Maniezzo, and A. Colorni. The Ant System: Optimization by a Colony of cooperating agents, IEEE Trans. on Syst., Man and Cybern., Part B Vol.26, 1-13, 1996.

[5] M. Dorigo and L. Gambardella. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem, IEEE Trans. on Evol. Comp. Vol.1: 53-66, 1997.

[6] L. Gambardella, and M. Dorigo. Solving Symmetric and Asymmetric TSPs by Ant Colonies, BioSystems, In Proc. of the IEEE International Conf. on Evol. Comp. (ICE'96), IEEE Press, Piscataway, USA, 622-627, 1996.

[7] L. Gambardella and M. Dorigo. Ant- Q: A Reinforcement Learning approach to the TSP, Proc. of the 12th. International Machine Learning Conference (ML-95), 252-260, 1995.

[8] L. Kaelbling, M. Littman, and A. Moore. Reinforcement Learning: A Survey, Journal of AI Res. 4,237-285, 1996.

[9] M. Srinivas and L. Patnaik. Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Trans. On Syst., Man and Cybern. , 24(4), 656-667,1994.

[10] T. Stützle and H. Hoos. MAX-MIN Ant System, FGCS Journal, 16(8), 889-914, 2000.