Second International Conference
Modelling and Development of Intelligent Systems
Sibiu - Romania, September 29 - October 02, 2011

Ant colony optimization pheromone correction strategies

Milan Tuba [1]

Abstract

Ant colony optimization (ACO) algorithm was recently successfully used to find suboptimal solutions to many hard optimization problems. There are few versions of the algorithm that improve its performance. The main problem is that algorithm can be trapped into local minima unable to escape. We implemented a hybridization for the ACO algorithm that proved to be efficient in avoiding that problem. The difference is that, while all other methods for avoiding local minima widen the search randomly, our algorithm excludes undesirable pars of the solution. We tested our method on four different graph optimization problems where in all cases it improved results.

1 Introduction

Many real-life problems can be represented as some kind of optimization problem, often an untractable one. Nature inspired metaheuristics have been used recently to find suboptimal solutions to hard optimization problems by simulating various natural phenomena. Swarm intelligence algorithms are a class of nature inspired algorithms that try to mimic collective intelligence of colonies of ants, bees, birds etc. By trying to simulate implicit intelligence of these swarms we talk about bee colony food finding or ant colony path finding but in essence, in all these diverse mimicking, we do two things. We exploit good found solutions, but also go to unknown less promising places in order to avoid being trapped in local minima. The successfulness of any algorithm is determined by proper balance between exploitation and exploration. This paper examines ant colony pheromone correction strategies which change exploitation and exploration behavior of the original algorithm and application of these strategies to some combinatorial problems.

2 Ant colony optimization (ACO) algorithm

The ant colony optimization (ACO) is a relatively new meta-heuristic for solving combinatorial problems. ACO is, like genetic algorithms (GA), population based. It was first used for the travelling salesman problem (TSP) by M. Dorigo [1] with very good results.

The basic idea of ACO is to imitate the behavior of ants in a colony while gathering food. Each ant starts from the nest and walks toward food. It moves until an intersection where it decides which path to take; in the beginning it seems to be a random choice, but after some time the majority of ants are using the optimal path (Figure 1). This happens because the colony works as a group and not just as individual ants and this is achieved by using pheromone. Each ant deposits pheromone while walking, which marks the route taken. The amount of pheromone indicates the usage of a certain route. Pheromone trail evaporates as time passes. Due to this a shorter path will have more of pheromone

[1]This research is founded by Ministry of Education and Science of Republic of Srbia, research Grant III-44006.

because it will have less time to evaporate before it is deposited again. The colony behaves intelligent because each ant chooses paths that have more pheromone.

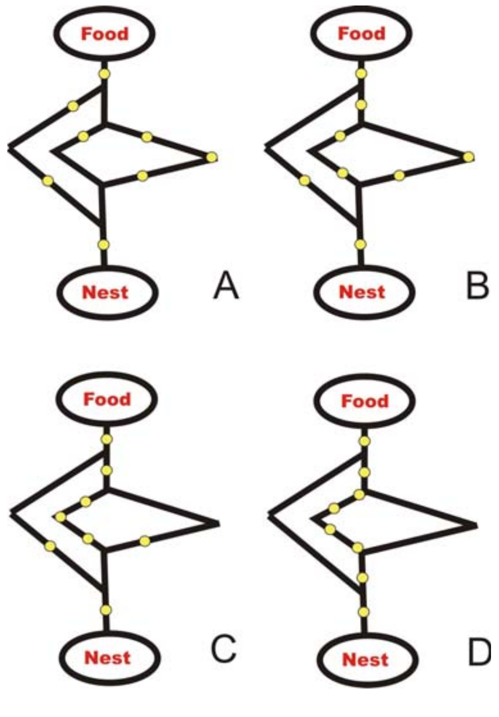

Figure 1: : Ant colony behavior over time

There are many different ways of converting the presented behavior into a computational system. We outline the one presented by Marco Dorigo and Luca Maria Gambardella [2], with small modifications,

$$s = \begin{cases} \arg\max\limits_{u \notin M_k} \{{\tau_{rs}}^{\alpha}{\eta_{rs}}^{\beta}\} & , q \leq q_0 \\ S & , q > q_0 \end{cases} \tag{1}$$

$$p_{rs}^{k} = \begin{cases} \frac{{\tau_{rs}}^{\alpha}{\eta_{rs}}^{\beta}}{\sum\limits_{u \notin M_k} {\tau_{ru}}^{\alpha}{\eta_{ru}}^{\beta}} & , s \notin M_k \\ 0 & , s \in M_k \end{cases} \tag{2}$$

Equations 1 and 2 give the probabilistic decision method that artificial ant k, currently at node r, after visiting nodes in M_k uses for choosing the next node s.

- q is a random variable chosen uniformly from [0, 1]
- q_0 is a predefined parameter that gives us a balance between exploitation (use of known good paths, $q <= q_0$) and exploration (search for new paths, $q > q_0$)
- In the case of exploitation, the next node is selected by the highest value of S, which gives the value of desirability of an edge depending on the amount of pheromone and its length. In Equations (1) and (2), τ_{rs} is the value corresponding to the amount of pheromone deposited on edge connecting r and s, and η_{rs} is the length of rs which is used as a heuristic.
- α and β are predefined parameters that specify the influence of the pheromone and the heuristic, respectively.
- In the case of exploration the next node is chosen at random with a probability distribution given by Equation (2), where p_{rs} is the probability of choosing edge rs.

The pheromone trail is created using two types of updates. Global update is used to reward good paths, or in other words more pheromone should be deposited on better paths. This is achieved by using the following formula

$$\tau_{ij} = (1-\gamma)\tau_{ij} + \gamma\Delta\tau^k \qquad ,\forall ij \in B^k \tag{3}$$

B_k is the set of all edges in the path ant k used, $\Delta\tau^k$ is the quality of that solution, and γ is a predefined constant.

The local updating is used to avoid creation of a very strong edge used by all ants, and it emulates pheromone evaporation. Every time an edge is chosen by an ant it loses some pheromone by the following formula where δ is a predefined constant.

$$\tau_{ij} = (1-\delta)\tau_{ij} + \delta\tau_0 \tag{4}$$

3 Variations of the ACO algorithm

There are different methods of improving ACO like certain types of hybridizations. Standard hybridizations are the combination of the basic algorithm with a local search [3], [4], [5] or some other algorithm. Combining ACO with genetic algorithms (GA) has resulted in algorithms that gave better results than separate use of these methods on a wide range of different problems [6], [7]. These hybridizations are effective in increasing the efficiency of ACO, but are often complicated for implementation. The complexity of their implementation is due to the fact that to separate algorithms need to be developed: one for the ACO and another for the local search or for the genetic algorithm. Also, it has been shown that hybridization may prevent the ACO from finding the optimal solution [8]. The other method of improving the performance of ACO is the use of different variations of the basic algorithm. On TSP different variations of ACO gave different quality of results, and no variation can be considered the best [9].

Variations of the basic ACO like elitist ant colony, rank based ant colony system, min-max ant system (MMAS) have been developed to improve the performance on TSP [10]. Similar variations have been used on other problems solved by ACO. All these variations have the problem of becoming trapped in local optima. An interesting approach, named the minimum pheromone threshold strategy (MPTS), was proposed for the quadratic assignment problems to solve the stagnation problem [11].

Ant System (AS), is the most basic implementation of ACO; in this version of the algorithm all ants are equal and leave pheromone. It is defined with Equations 5 and 4. AntS is the set of all the solutions created by ants in the current step of the algorithm.

$$\forall i \in \bigcup_{V_i^{'} \in AntS} V_i^{'}$$

$$\tau_i = \sum_{V_k^{'} \in AntS} \frac{1}{\sum\limits_{j \in V_k^{'}} w(j)} \tag{5}$$

Reinforced Ant System (RAS), which is the same as Ant system, except that the global best solution is reinforced each iteration.

$$\forall i \in V_{gb}^{'} \cup \bigcup_{V_i^{'} \in Ants} V_i^{'}$$

$$\tau_i = \frac{1}{\sum\limits_{j \in V_{gb}^{'}} w(j)} + \sum_{V_k^{'} \in AntS} \frac{1}{\sum\limits_{j \in V_k^{'}} w(j)} \tag{6}$$

In some variations of this method the iteration best solution is also reinforced each iteration. With this approach the basic AS is made to be slightly greedier. It is defined with Equations 6 and 4.

Elitist Ant System (EAS) is the version of the ACO algorithm where individual ants do not automatically leave pheromone. In each iteration step of the colony, or in other words when all the ant complete their solutions, only the global best solution will be used to update the pheromone trail. In this way, the search is even more centralized around the global best solution. It is defined with Equations 7 and 4.

$$\tau_i = \frac{1}{\sum\limits_{j \in V'_{gb}} w(j)}, \quad \forall i \in V'_{gb} \tag{7}$$

MinMax Ant System (MMAS) is the same as the Elitist Ant colony System, but with an extra constraint that all pheromone values are bounded. We adopt the formulas presented in article [12] in which max is calculated dynamically as new best solutions are found by Equation 8, and min is calculated at the beginning of calculations by Equation 9. Parameter avg is the average number of vertexes that are possible to be chosen, $pbest$ is the possibility of the best overall solution being found and τ_0 is the initial value of the pheromone trail calculated as the quality measure of the greedy algorithm solution.

$$\tau_{max} = \frac{1}{(1-p)}\tau_{gb} \tag{8}$$

$$\tau_{min} = \frac{\tau_0(1 - \sqrt[n]{p_{best}})}{(avg - 1)\sqrt[n]{p_{best}}} \tag{9}$$

This variation has two effects that improve the effectiveness of EAS. First, the pheromone trial will not become very strong on some good vertexes and making them a part of almost all newly created solutions. By giving a lower bound to the pheromone trail the potential problem of certain parts of the solution being totally excluded from the search due to very weak values of pheromone is avoided.

Rank Based Ant Colony System (RANKAS) [13] is a modification in which besides the quality we also use the rank (R) of found solutions. Rank is defined by the quality of the solution compared to solutions found by other ants in the same iteration. It is defined with Equations 4 and 10:

$$BRank = \{V | (R(V) < RK)\hat{(}V \in AntS)\}$$

$$\forall i \in V'_{gb} \cup \bigcup_{V'_i \in BRank} V'_i$$

$$\tau_i = \frac{1}{\sum\limits_{j \in V'_{gb}} w(j)} + \sum_{V'_k \in BRank} \frac{(RK - R(V))}{RK} \frac{1}{\sum\limits_{j \in V'_k} w(j} \tag{10}$$

In the implementation of this algorithm, it is important how many best solutions will be taken into account when updating the pheromone trail. In Equation 10 parameter RK is used to define the number of best ranked ants which will affect the trail. This parameter is user defined and it is very important for the effectiveness of this algorithm. In its extreme cases when RK is equal to 0 $RANKAS$ is equivalent to EAS.

4 Avoiding stagnation in the ACO algorithm

The ACO meta-heuristic algorithm has several different variations which are used of which elitist ant system (EAS) and Min-max ant system (MMAS) are used most commonly.

EAS increases the efficiency of the basic ACO by making its search more greedy; this is done through intensifying the search near the global best solution. In this version only the global best solution (or in some variations only the iteration best solution) deposits the pheromone. This has the weakness of some edges becoming very strong and becoming a part of almost all solutions, while others becoming very weak and being chosen very rarely.

Min-max ant system (MMAS) [12] is an improvement of the EAS that tries to solve this problem. The improvement is done by adding an extra constraint that all pheromone values are bounded, $\tau_i \in [\tau_{\min}, \tau_{\max}]$. There are two effects of this. First, the pheromone trial will not become too strong on good edges and second, the other potential parts of the solution will not be totally excluded from the search due to very weak values of the pheromone. This approach improves the results of the EAS, but the stagnation still happens. Since τ_{min} has to be set to a low value, once the trail of a vertex reaches τ_{min} it becomes chosen very rarely. The search is never intensified near that point unless it becomes a part of the global best solution. The basic solution for this problem is to add some criteria for search stagnation and if stagnation has occurred to reset the pheromone trail to initial values [14].

Minimum pheromone threshold strategy (MPTS) is another approach to solving the early stagnation problem whose effectiveness has been shown on the quadratic assignment problems [11]. The idea behind MPTS it to use minimum threshold value τ_{mt} that is bounded $\tau_{min} < \tau_{mt} < \tau_{max}$. In the beginning of the algorithm it is set to some initial value and then adjusted during the search, depending on the performance. Threshold τ_{mt} is used for updating the pheromone trail. When the search is performed, values in the pheromone trail τ_i are compared to the τ_{mt} and if τ_i is lower than τ_{mt}, than τ_i is changed to τ_{max}. Thus the MPTS avoids reinitialization of the pheromone trail and explores the solution search space more systematically. No loss of information occurs related to the pheromone trail reset, while the good properties of the MMAS are preserved.

The idea of our hybridization is to use the information about the best-found solution to perform corrections on the pheromone trail. We introduce the concept of suspicion that some edges of the best found tour are not good. We try to direct the search to new areas of solution space that are less suspicious which means with less undesirable properties. Directing the search is done by greatly decreasing the pheromone trail values at suspicious edges.

Before we explain in detail this concept we wish to point out important differences between the MPTS for the basic MMAS and our hybridization. In the MMAS the increasing of diversification is done non-selectively to the whole search space, first by adding τ_{min} to prevent total exclusion of some vertexes from the search and, in the case of stagnation, by resetting the pheromone trail to initial values. When MPTS is added to the MMAS the reinitialization of the trail is avoided by testing at each iteration the pheromone values for all vertexes and if they have dropped below τ_{mt} they are set to τ_{max}. This way some edges are added to the small group of edges that are frequently chosen by ants. A big drawback of this approach is that added edges are chosen just for having low values of τ_i which is a relatively random process. As a result, edges that do not belong to good solutions are often reintroduced in the search. Once the pheromone value for an edge is increased, it will take a long time for it to be removed from the intensively tested group. In our hybridization however, we do not add edges to the "popular" group but rather remove edges with suspicious properties, making the group smaller. In the following iterations ants will first select edges from the popular set, and when none are left, edges with better properties. This way we direct to new search areas that are less suspicious which means with less undesirable properties.

5 Suspicious elements exclusion pheromone correction strategy

Here we describe our new type of hybridization for the ACO and implement it for the TSP. We improve the ACO with a strategy for leaving local optima i.e. avoiding stagnation in search for the best solution. This method is based on correcting the pheromone trail used in the ACO. We calculate this correction based on the properties of the best-found solution so far. The basic idea of this correction is to lower the possibility of edges with high level of undesirability to belong to the optimal solution. We do not claim that our method gives the best results on TSP compared to all other developed algorithms, but we show that our strategy improves results acquired by the ACO and that it is simple to add the method to the existing algorithms.

5.1 Application to the TSP

When the ACO algorithm gets trapped in local optima for TSP, in many cases it was obvious from visual observation which corrections should be made, or more precisely what should not appear in the shortest path (Fig.2). There where two simple criteria used on the edges belonging to the best found tour: very long edges and intersecting edges are very unlikely to be a part of the optimal path. The next step was to find a way to, without of major corrections to the ACO algorithm, remove them from the ants search path. The solution was significantly lowering the amount of pheromone on randomly selected highly suspicious edges belonging to the best path and letting the colony resume its search.

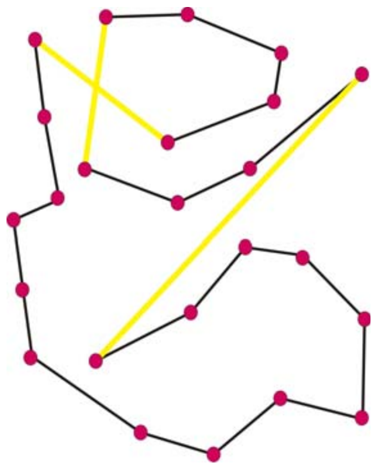

Figure 2: : Example of local optima found with ACO for TSP with suspicious edges on path

We have divided the correction into two parts: one considering the edge lengths, and the second that is connected to intersecting ones. All intersecting edges would be considered as highly suspicious and the pheromone trail would be corrected on them. Edges for which pheromone trail correction will be applied due to their length is defined in the following way. First we define a heuristic for suspicion Eq. 11

$$Sus(rs) = length(rs) \tag{11}$$

The next step is defining the probabilities of edges being selected for pheromone correction

$$p_{rs}(selected) = \frac{RK - RankSus(rs)}{RK} \tag{12}$$

In Eq. (12) instead of using the value of Sus for edges, we used $RankSus$ which represents their rank by suspicion. RK is the maximal number of edges that are considered for correction. The final step is to lower the pheromone trail for the selected edges:

$$\begin{aligned} &\forall rs \in Selected \\ &\tau_{rs} = \delta \tau_{rs} \end{aligned} \tag{13}$$

The use of suspicion defined in Eq. (11) is not fully effective because the same group of edges would be repetitively selected until a better tour was found. Because of this we introduce an improved suspicion criteria:

$$CorSus(rs) = \ Sus(rs) \ * ExSusepect(rs) \tag{14}$$

The improvement consists of tracking which edges have already been selected and preferring the selection of new edges. To do this, a new array $ExSuspect$ is introduced with elements initially set to 1. If edge rs is selected, the following correction is done:

$$\begin{aligned} &0 < \lambda < 1 \\ &ExSuspect(rs) = ExSuspect(rs) * \lambda \end{aligned} \tag{15}$$

If a new best set is found, the values of *ExSuspect* are reset to 1. A more complex suspicion criteria could have been used, that would better analyze the properties of the best found solution but we wished to show that even with a relatively simple heuristic improvements can be archived.

The last step that is needed to completely describe this hybridization is to define some stagnation criteria. We used the following: stagnation occurred if there was no change to the global best solution in at least n iterations. If these criteria are satisfied, we apply previously defined correction algorithm to the pheromone trail. The pseudo-code for an iteration step for the improved ACO is:

```
Reset Solution for all Ants

 while  (! AllAntsFinished)
   For All Ants
     If(AntNotFinished)
       begin
         add new edge AB to solution
         based on probability
         local update rule for A
       End
     End for
   End while


 Compute Global Update

 If(Stagnation)and(UsingSuspisionImprovemnet)
   Use SuspisionCorrectionMethod
```

Experimental results show that our pheromone correction strategy for removing undesirable elements improves ACO algorithm for the TSP without any increase in computational complexity.

5.2 Other applications

Methods similar to one described in the previous section for the TSP were developed for three other graph problems: the minimum weight vertex cover problem, the minimum weight dominant set and the minimum connected dominant set problem. They are very similar and all three proved to be very efficient. For some problems we obtained significantly better results [15] than previous algorithms. The hybridization, although simple, has to be implemented as a module to the software system. That was easy since we developed a software framework [16] for the ACO that allows exactly that: easy incorporation of new module. A framework is a special kind of software library, that is similar to an application program interface (API) in the class of packages, that makes possible faster development of applications. However, while an API consists of a set of functions that user calls, a framework consists of a hierarchy of abstract classes. The user only defines suitable derived classes that implement the virtual functions of the abstract classes. Frameworks are characterized by using the inverse control mechanism for the communication with the user code: the functions of the framework call the user-defined functions and not the other way round. The framework thus provides full control structures for the invariant part of the algorithms and the user only supplies the problem-specific details. Such environment was suitable to implement our new pheromone correction strategy as a module that can be called from the framework.

6 Conclusion

We developed a pheromone correction strategy that is simple and universally applicable. It helps ACO algorithm escape being trapped in local minima, but in a more efficient way than other known methods.

While other used methods add new (usually random) elements to the currently investigated part of the solution space, our hybridization removes undesirable elements from that space, allowing more desirable elements to be included. Tests show that, without increasing computational complexity of the ACO algorithm, better results are achieved. Future development will include implementation in the software framework of our hybridization for other graph problems.

Acknowledgment: This paper is founded by Ministry of Education and Science of Republic of Serbia from the research Grant III-44006.

References

[1] Dorigo M, Maniezzo V: Ant Colony system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics - Part B*, Vol. 26, No.1, pp. 29-41, 1996.

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

[3] Y.J Feng and Z.R. Feng: Ant colony system hybridization with simulated annealing for flow-shop scheduling problems. *WSEAS Transaction on Business and Economics*, Vol. 1, No. 1, pp. 133-138, 2004.

[4] Qi Chengming: An ant colony algorithm with stochastic local search for the vrp. *IEEE Computer Society, 3rd International Conference on Innovative Computing Information and Control*, pp. 464-468, 2008.

[5] J. Levine, F. Ducatelle: Ant colony optimization and local search for bin packing and cutting stock problems *Journal of the Operational Research Society*, pp. 705-716, 2004.

[6] Z.-J. Lee, S.-F. Su, C.-C. Chuang, K.-H. Liu: Genetic algorithm with ant colony optimization (ga-aco) for multiple sequence alignment *Applied Soft Computing* pp. 55-78 2008.

[7] Hong-hao Zuo, Fan-lun Xiong: Time Ant Colony Algorithm with Genetic Algorithms. *IEEE International Conference on Information Acquisition*, pp. 1057-1061, 2006.

[8] Frank Neumann, Dirk Sudholt, and Carsten Witt: Rigorous analyses for the combination of ant colony optimization and local search. In *Ant Colony Optimization and Swarm Intelligence, LNCS, Springer-Verlag* Vol. 5217, pp. 132-143, Berlin, Heidelberg, 2008.

[9] D.Asmar ,A. Elshamli, S. Areibi: A Comparative Assessment of ACO Algorithms Within a TSP Environment. *In DCDIS: 4th International Conference on Engineering Applications and Computational Algorithms*, pp. 462-467, 2005.

[10] Thomas Stützle and Marco Dorigo. Aco algorithms for the traveling salesman problem. In *Evolutionary Algorithms in Engineering and Computer Science: Recent Advances in Genetic Algorithms, Evolution Strategies, Evolutionary Programming, Genetic Programming and Industrial Applications, K Miettinen, P Niettaanmaki, M M Makela and J Periaux, editors*, pp. 500, Willey, 1999.

[11] Kuan Y. Wong and Phen C. See: A new minimum pheromone threshold strategy (mpts) for maxmin ant system. *Applied Soft Computing*, Vol. 9(3), pp. 882-888, 2009.

[12] T. Stützle and H.H. Hoos: MAX MIN Ant System. *Future Generation Computer Systems*, Vol. 16, pp. 889-914, 2000.

[13] B. Bullnheimer, R. F. Hartl, and C Strauss: A new rank-based version of the ant system: a computational study. *Central European Journal for Operations Research and Economics*, Vol. 7 No. 1, pp. 25-38, 1999.

[14] T. Stützle, H. Hoos: Improvements on the ant system: Introducing the max min ant system. In *Third International Conference on Artificial Neural Networks and Genetic Algorithms, Springer Verlag, University of East Anglia, Norwich, UK,* pp. 245249, 1998.

[15] Jovanovic R, Tuba M. An ant colony optimization algorithm with improved pheromone correction strategy for the minimum weight vertex cover problem. *Applied Soft Computing,* Vol. 11, No. 8, pp. 5360-5366, 2011.

[16] R. Jovanovic, M. Tuba, D. Simian: An object-oriented framework with corresponding graphical user interface for developing ant colony optimization based algorithms. *Wseas Transactions on Computers,* Vol. 7, No. 12, pp. 1948-1957, 2008.

Milan Tuba
Megatrend University Belgrade
Faculty of Computer Science
Bulevar umetnosti 29, N. Belgrade
SERBIA
E-mail: *tuba@ieee.org*