Controlling the Balance of Exploration and Exploitation in ACO Algorithm

Ayad Mohammed Jabbar
School of Computing, College of Arts and Sciences, University Utara Malaysia, 06010 Sintok, Kedah, Malaysia

ayad_mohammed@ahsgs.uum.edu.my

Abstract

Ant colony optimization is a meta-heuristic algorithm inspired by the foraging behavior of real ant colony. The algorithm is a population-based solution employed in different optimization problems such as classification, image processing, clustering, and so on. This paper sheds the light on the side of improving the results of traveling salesman problem produced by the algorithm. The key success that produces the valuable results is due to the two important components of exploration and exploitation. Balancing both components is the foundation of controlling search within the ACO. This paper proposes to modify the main probabilistic method to overcome the drawbacks of the exploration problem and produces global optimal results in high dimensional space. Experiments on six variant of ant colony optimization indicate that the proposed work produces high-quality results in terms of shortest route.

Keywords*:* Ant colony optimization, ant system, local search, traveling salesman problem, exploration, exploitation.

الخلاصة

خوارزمية النمل هي واحده من خوارزميات البحث عن الحلول المثلى ضمن فضاء واسع من الاحتمالات على نحو شبيه بطريقة النمل في البحث والتقفي لإيجاد الحلول لبعض المشاكل المعقدة التي يصعب حلها باستخدام خوارزميات الذكاء الاصطناعي التقليدية. تستخدم هذه الخوارزمية عمليه البحث في فضاء الحالات للاستنتاج حلول مختلفة اثناء عمليه البحث معتمدة على التوازن بين استكشاف حلول جديدة لتوسيع رقعة البحث وبين استغلال الحلول الجيدة لتحسين الحلول المستخرجة مسبقا. ان عمليه خلق توازن بين هاتين العمليتان يؤدي لتحسين النتائج والخروج بحلول أكثر امثليه. هدف هذا البحث هو ايجاد قانون احتمالي أكثر ملاءمة وقادر على خلق توازن أفضل بين عمليتي الاستكشاف والاستغلال. بعد اجراء ستة تجارب مختلفة من حيث أشكال البينات تم اثبات ان التحسين في هذه الخوارزمية يؤدي الى انتاج حلول عالية الجودة من ناحية قصر طول المسار المكتشف.

الكلمات المفتاحية: خوارزمية النمل، انظمة النمل، البحث المحلي، مشكلة البائع المتجول، الاستكشاف، الاستغلال.

1. Introduction

Ant colony optimization (ACO) is a meta-heuristic algorithm utilized for solving combinatorial optimization problems. The algorithm is a population-based solution encompassing the process of multiple solutions improvement for optimization problem. Food foraging is a mechanism represents the behavior of ants searching for food source which is later simulated as a model called probabilistic model. The model is the core of ACO, which calculates the shortest route in traveling salesman problem (TSP). ACO variants developed based on the foundations algorithm inspired of the nature called ant system algorithm (AS) (Dorigo *et al.*, 2004). The main feature of the developed family of AS is to find the appropriate balance between exploration and exploitation (Blum, 2002). However, poor balance causes the search process to stagnate and produces slow convergence (Boussaïd, *et al.*, 2013). The justification of previous studies was based on modifying the pheromone memory matrix. The pheromone memory matrix is employed by the algorithm to control the search space. This paper proposed to improve the ACO by modifying the ACO formula. The purpose of modification is to increase the balance between exploration and exploitation. The structure of the paper is organized as follows: Section II reviews the swarm intelligence and optimization methods, while Section III provides a comprehensive overview on ant colony optimization and its variants. Section IV presents the importance of exploration and exploitation and an overview of related work. The proposed solution and experimental results are mentioned in the section V and VI, respectively. The conclusion and recommendations for future works are given in final section.

2. Swarm Intelligence Algorithm

Optimization methods generally consist of various categories. Firstly, the exact approach produces an optimal solution in undefended time (Birattari, 2009). Second, the estimation approach produces optimal or near-optimal solutions based on a predefined range of candidate solutions. The third is the approximation approach, which includes two sub approaches: single-based solution and population-based solution. The single-based approach improves the single available solution over time, while the population-based approach allows for multiple candidate solutions produced by different functions. Population-based approaches include swarm intelligence and evolutionary algorithm. Evolutionary algorithms (EA) such as genetic algorithm, evolutionary programming, evolution strategy, genetic programming, and learning classifier systems are the spine of evolutionary algorithm (Eiben & Smith, 2003). The main idea of EA remover of poor individuals allowing good individual to survive and modifies the individual chromosomes to refine the population of candidate solutions (see Figure 1).

```
Procedure Evolutionary-Computation
Begin
    t ⟵ ();
    Initialize P(t);
    Evaluate P(t);
    While (terminating condition not reached) do
      Begin
       t⟵ t+1;
       Select P(t) from P(t-1);
       Alter P(t);
       Evaluate p(t);
     End While;
 End.
```

Figure 1: Evolutionary Computing Process

Swarm intelligence (SI) is a collection of natural behaviors inspired by social insects that are employed in artificial intelligence (Beni & Wang, 1989). Particle swarm optimization (PSO) and Ant Colony optimization (ACO) are the foundation of SI (Shelokar, Siarry, Jayaraman, & Kulkarni, 2007). The process of SI builds based on the cooperative work in the social insects such as termites or other animal societies (see Figure 2), such as bee swarms, fish schools and bird flocks to produces optimal solution. The importance of optimization method is due to the flexibility, robustness, decentralization and self-organization.

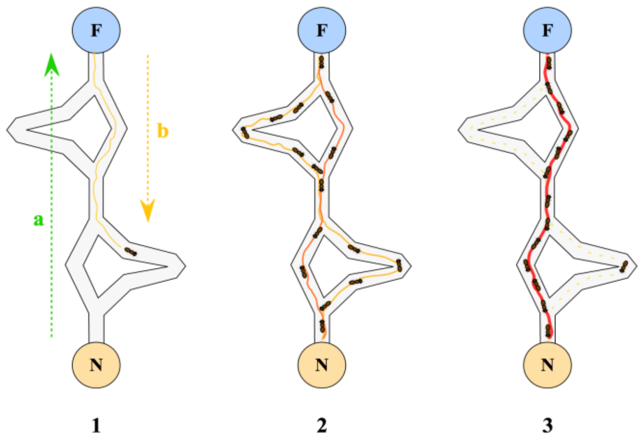

Figure 2: Searching process of ants

Animal society demonstrates effective and intelligent behavior in solving complex problems in nature. As shown in fig 1, 2 and 3, ant convergence is incremental and increases over the time due to the high value of pheromone density deposited by ants. The longest path progressively becomes less favorable over time and automatically loses pheromone value due the evaporation process. Ant colony optimization to produce more optimal paths is the core of this research paper.

3. Ant Colony Optimization

ACO is a meta-heuristics algorithm proposed by Dorigo to solve the traveling sales man problem (Dorigo *et al.*, 2004). Meta-heuristics are frameworks that govern heuristics methods (Sörensen, 2015). The meta-heuristics contain operators and concepts that are applied to control the heuristic algorithm. The algorithm is population-based and utilizes multiple solutions to construct an optimal or near-optimal solution. The algorithm is applied to solve the problem of finding optimal path in the TSP problem. ACO incorporates three important functions, which are the solution construction, pheromone update trail, and the daemon actions. The first is used to coordinate an ant colony by moving ants through neighbors in the construction graph. The second is used to control the colony by updating pheromone trails. Pheromone density may either increase, when ants move through the construction graph, or decrease with pheromone evaporation over time via an evaporation mechanism. The new updated pheromone increases the probability of an ant colony to produce a more optimal solution due to increases in the probability of exploration. Daemon actions are the process of centralized actions. This procedure is performed by multiple agents, such as local optimization procedure or improving the quality of results using additional pheromones based on global information collected by ants. Evaporation is the process of decreasing the pheromone quantity on the trails. The purpose is to increase the probability of exploring the graph widely and avoiding stagnation, which happens when ants cover on a few trails because of the higher pheromone density available on those trails. Marco Dorigo and Luca Maria converted this ant behavior into a computational system, as shown in Equation 1.

$$P_{ij}^{K}(t) = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}.[\eta_{ij}]^{\beta}}{\sum_{l \notin tabu_{k(t)}} [\tau_{ij}]^{\alpha}.[\eta_{ij}]^{\beta}} & if\ j \notin tabu_k(t) \\ & otherwise \\ 0 & \end{cases} \qquad (1)$$

Equation (1) is a probabilistic decision method which is frequently used by ants at step in the node r for choosing the next node j when node j $\notin tabu_k(t)$, where □$\notin tabu_k(t)$, which means is not a member in the $tabu$ list. In other words, the node has not visited yet by ant k. The value of heuristic function is estimated based on the value of parameters α and β. τ_{ij} is the amount pheromone deposited by ants which initially represent the inverse proportional to the total number of nodes N as defined by $\tau_{ij}(0) = \frac{1}{N}$. The quality of a chosen path is determined by the quantity of the pheromones deposited by ants. The mechanism of updating pheromone performance is based on using an algorithm. For example, in the global update, best paths are determined, then updated with a high quantity of pheromone as a reward. The following formula achieves the global updating task.

$$\tau_{ij} = (1\text{-}p)\,\tau_{ij} + p\Delta_{\tau^k} \quad ,\forall(i,j)\ \in B^k \qquad (2)$$

Where Δ_{τ^k} represents the quality of solution and p represents the constant value in range of [0,1] provided by user. A local update is performed to overcome the creation of connections which has strong amount of pheromone. Frequently, connections lose some pheromone density when ants chose a particular connection. Formula (3) achieves the task of local pheromone update.

$$\tau_{ij} = (1\text{-}\delta)\,\tau_{ij} + \delta_{\tau_0} \qquad (3)$$

The value of δ is a predefined declared by the user in the range of [0, 1], while τ0 represents the quality solution produced by the ACO algorithm when the heuristic function represents the edge length. The TSP procedure of the ACO algorithm applied as in the following:

```
procedure ACO Metaheuristic
    ScheduleActivities
        ConstructAntsSolutions
        UpdatePheromones
        DaemonActions    % optional
    end-ScheduleActivities
end-procedure
```

Figure 3: Ant colony optimization Framework

At first, ants m are randomly placed in a chosen city. The number of ants m are less or equal to the number of cities. At each iteration, ant k currently located at city i constructs a tour by selecting the next city j at the tth iteration. The chosen performed based on the probabilistic formula of Equation 1. The probability $p_{ij}^{k}\,(s)$ of choosing the next city is based on the pheromone heuristic τ_{ij} that deposited by ant m on the edge between city i and city j and the available heuristic information η_{ij}. ACO utilized a $tabu$ list to store each ant location and the current partial tour, i.e. $tabu_k(s)$ stores a set of cities visited by ant k so far at time s. The pheromone density is updated via tour construction based on all ants m, permitting to add the pheromone on all visited edges. At the end of the iteration, the $tabu$ list is emptied and each ant can choose an alternative path for the next cycle. The evaporation performs simultaneously in the pheromone update stage. The produced solution obtained by each ant is chosen frequently, eventually leading to the selection of the optimum in the final iterations.

4. Exploration and Exploitation

Exploration and exploitation are the most critical yet vague topics. Exploration is the process of searching in the whole search space locking for more optimal and diversity solutions. An example of exploration is the ACO evaporation process which decreases the pheromone density to escape stagnation and occurs due to a high amount of pheromone concentrated on a few connections. The exploitation is locally applied. The process performs by improving the available obtained solution. An example is applying local search in ACO which improved the result by performing some medication in the obtained neighborhood solution. The key success of any meta-heuristic algorithm is based on the two aforementioned components. Entropy indicator employed by Pellegrini in (Angus, 2009) for characterizing the amount of pheromone to increases the exploration of ACO algorithm. Colas et al. utilized the entropy indicator to provide information as a feedback of ACO parameters (Christian Blum, 2012). In the solution construction, the indicator is simultaneously applied at each node. Each node of the selection probabilities is calculated and evaluated for all nodes by the ants as follows:

$$\mathcal{E}_i = -\sum_{j=1}^{l} p_{ij} \, log \; p_{ij} \qquad (4)$$

Where p_{ij} represents the probability to select the appropriated arc between node i and node j when ants are located in node i and l ,$1 \leq l \leq n-1$, represents the possible number of available choices. Therefore, the average entropy for each node will be calculated in the same way. Blum (A. Eiben & Smit, 2011) used a convergence indicator, employed as feedback indicator in the hypercube framework (Pellegrini, Stützle, & Birattari, 2012). The framework used to increase the ant convergence. The convergence factor is calculated as follows:

$$\frac{\sum_{t_{ij}\in T} min\{t_{max} \text{-} \, t_{ij}, t_{ij} \text{-} \; t_{min}\}}{n^2} \qquad (5)$$

Where the pheromone matrix represents as T and pheromone trails represents as t_{ij} which is the amount of pheromone deposited by ants. Acceptance a criterion (AC) is another tool used as a factor of restarting strategies to increase the exploration amount in ACO (Sagban, Ku-mahamud, & Bakar, 2017; Sagban, Ku-Mahamud, & Bakar, 2015). If the iteration i_{last} was the appropriate solution found in the ACO, then the optimal point for restarting the strategies can be calculated as follows:

$$AC = \begin{cases} s'' \;\; if \; f(s'') < f(s) \\ s''' \; if \; f(s'') \geq f(s), i\text{-}i_{last} > i_r \\ s \qquad\qquad\qquad\qquad otherwise \end{cases} \qquad (6)$$

The value of s''' is randomly generated based on the new initial solution which corresponds on the restart strategies of the algorithm. The exploration increases with no guarantee that the ants will visit the same location which has been visited in the previous iteration.

5. The Proposed Solution

To produce more optimal solution, we have to shed the lights on the probabilistic formula. Modification can be useful to decrease the complexity cost time and parameters reduction. The novel of the paper is modifying the probabilistic formula to be simplest formula which can perform the same job with better quality of result. The probabilistic formula can be simplified as follows:

$$P_{ij}^{K}(t) = \begin{cases} \dfrac{log\left([\eta_{ij}]^{t_{ij}}\right)}{\sum_{l \notin tabu_{k(t)}} log\left([\eta_{ij}]^{t_{ij}}\right)} & if\ j \notin tabu_k(t) \\ 0 & otherwise \end{cases} \qquad (7)$$

According to formula 7, the first justification is to modify the probability to select node i based on the amount of pheromones, which are based on the probabilistic formula that returns numbers which are semi convergent. Thus, ants explore the space in more deeply looking forward for better optimal solution with no attention for paths that have high density of pheromone. The second justification omits the parameters α and β, which decreases the cost time.

6. Experimental Results

This section explores the performance of different experiments used in different popular benchmarks. Six experiments were conducted for different benchmarks in variant of ACO. The amount of time that set for the algorithm is about 10 seconds for each experiment, to ensure that the algorithms would produce optimal or near-optimal solutions. The proposed work was coded in Java language as provided by Dorigo et al. 2004 using C. The settings of the algorithm are used by default without employing local search (LS). The algorithm utilized as follows. Twenty-five (25) is the number of ants used. The (α) parameter is equal to 1 while (β) parameter is equal to 2 for all variants of ACO. The benchmarks consist of six datasets such as pr2392.tsp, pcb3038.tsp, d1291.tsp, rat783.tsp, d198.tsp and eil51.tsp (Hahsler & Hornik, 2007). Table 1 shows the results produced by the variants of ACO and the results of our work for fixed number of iterations equal to 10. The results can conclude that the new formula performs better in high dimensional dataset such as pr2329 and pcb3038 while progressively the performance going worse in the small dataset such as 2at783, d198 and eil51. The results show strong outperformance compared with the original formula. The results are divided into two columns, labelled original formula and proposed formula. The original formula represents the equation mentioned in 1, while the other represents the proposed formula in equation 7.

Table 3: Optimized Results of Various Experiments

ACO	TSP instance											
	pr2392		**pcb3038**		**d1291**		**rat783**		**d198**		**eil51**	
	Original formula	Proposed formula	Original formula	Proposed formula	Original formula	Proposed formula	Original formula	Proposed formula	Original formula	Proposed formula	Original formula	Proposed formula
AS	515428.7	462369.0	186813.1	170799.9	62902.2	59841.5	13252.5	12409.0	17514.6	18037.7	434.4	484.4
EAS	650182.2	463392.7	237569.1	170447.4	83903.2	59695.4	14590.7	12395.9	17809.2	17984.5	444.0	483.8
RAS	708512.0	460249.1	255023.4	169571.5	98528.0	58772.4	17989.3	12304.5	21671.7	17598.5	577.1	480.6
BWAS	678472.2	462446.7	246807.4	171049.0	91614.6	59611.0	16164.5	12424.1	17840.3	18055.3	484.1	484.8
ACS	703614.7	459405.5	253424.7	169465.2	97784.5	58724.7	17769.3	12279.0	21552.7	17620.0	546.3	482.0
MMAS	554756.7	461784.7	206303.6	171132.9	69383.0	59416.3	13431.6	12457.5	16912.5	18013.3	427.2	483.0

7. Conclusion

This work proposes a new formula to increase the exploration of ACO. The new formula decreases processing time due to simplify and explore the search space deeply. The experiments performed with ACO variant produced promising results, while the results obtained show that the new formula produced robust results in high dimensional data. For future work, it is highly recommended to use the proposed formula in another application such as classification and clustering. In addition, the proposed formula requires some enhancements in terms of parameters to produce more optimal results.

8. References

Angus, D. (2009). Niching for ant colony optimisation. *Studies in Computational Intelligence*, *210*, 165–188. http://doi.org/10.1007/978-3-642-01262-4_7

Beni, G., & Wang, J. (1989). Swarm Intelligence in Cellular Robotic Systems. In *NATO Advanced Workshop on Robots and Biological Systems*. Il Ciocco, Tuscany, Italy.

Birattari, M. (2009). Tuning Metaheuristics: A Machine Learning Perspective. In *Tuning Metaheuristics: A Machine Learning Perspective* (Second edi, Vol. 197, p. 37). Berlin: Springer. http://doi.org/10.1007/978-3-642-00483-4

Blum, C. (2002). ACO applied to group shop scheduling: A case study on intensification and diversification. *Ant Algorithms*, 14–27. http://doi.org/10.1007/3-540-45724-0_2

Blum, C. (2012). Hybrid metaheuristics in combinatorial optimization: A tutorial. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 7505 LNCS, pp. 1–10). http://doi.org/10.1007/978-3-642-33860-1_1

Boussaïd, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. *Information Sciences*, *237*, 82–117. http://doi.org/10.1016/j.ins.2013.02.041

Dorigo, M., Birattari, M., & Stutzle, T. (2004). *Ant colony optimization*. *Bradford Company* (Vol. 1). Scituate, MA, USA.: ACM. http://doi.org/10.1109/MCI.2006.329691

Eiben, A. E., & Smith, J. E. (2003). What is an Evolutionary Algorithm? *Introduction to Evolutionary Computing*, 15–35. http://doi.org/10.1007/978-3-662-05094-1_2

Eiben, A., & Smit, S. K. (2011). Parameter tuning for configuring and analyzing evolutionary algorithms. *Swarm and Evolutionary Computation*, *1*(1), 19–31. http://doi.org/10.1016/j.swevo.2011.02.001

Hahsler, M., & Hornik, K. (2007). TSP – Infrastructure for the Traveling Salesperson Problem. *Journal Of Statistical Software*, *1*(1), 1–21. http://doi.org/10.1002/wics.10

Pellegrini, P., Stützle, T., & Birattari, M. (2012). A critical analysis of parameter adaptation in ant colony optimization. *Swarm Intelligence*, *6*(1), 23–48.

Sagban, R., Ku-mahamud, K. R., & Bakar, M. S. A. (2017). Reactive max-min ant system with recursive local search and its application to tsp and qap. *Intelligent Automation and Soft Computing*, *23*(1), 127–134. http://doi.org/10.1080/10798587.2016.1177914

Sagban, R., Ku-Mahamud, K. R., & Bakar, M. S. A. (2015). Nature-Inspired Parameter Controllers for ACO-based Reactive Search. *Research Journal of Applied Sciences, Engineering and Technology*, *11*(1).

Shelokar, P. S., Siarry, P., Jayaraman, V. K., & Kulkarni, B. D. (2007). Particle swarm and ant colony algorithms hybridized for improved continuous optimization. *Applied Mathematics and Computation*, *188*(1), 129–142. http://doi.org/10.1016/j.amc.2006.09.098

Sörensen, K. (2015). Metaheuristics-the metaphor exposed. *International Transactions in Operational Research*, *22*(1), 3–18. http://doi.org/10.1111/itor.12001