Two-dimensional Pheromone in Ant Colony Optimization*

Grażyna Starzec[1[0000−0002−0813−798X]], Mateusz Starzec[1[0000−0001−8258−2443]], Sanghamitra Bandyopadhyay[2[0000−0003−4299−9599]], Ujjwal Maulik[3[0000−0003−1167−0774]], Leszek Rutkowski[1,4[0000−0001−6960−9525]], Marek Kisiel-Dorohinicki[1[0000−0002−8459−1877]], and Aleksander Byrski[1[0000−0001−6317−7012]]

[1] AGH University of Science and Technology, Krakow, Poland, gstarzec@agh.edu.pl, starzec.mateusz@gmail.com, {rutkowski,doroh,olekb}@agh.edu.pl

[2] Indian Statistical Institute, Kolkata, India, sanghami@gmail.com

[3] Jadavpur University, Department of Computer Science and Engineering, Kolkata, India, ujjwal.maulik@jadavpuruniversity.in

[4] Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland

Abstract. Ant Colony Optimization (ACO) is an acclaimed method for solving combinatorial problems proposed by Marco Dorigo in 1992 and has since been enhanced and hybridized many times. This paper proposes a novel modification of the algorithm, based on the introduction of a two-dimensional pheromone into a single-criteria ACO. The complex structure of the pheromone is supposed to increase ants' awareness when choosing the next edge of the graph, helping them achieve better results than in the original algorithm. The proposed modification is general and thus can be applied to any ACO-type algorithm. We show the results based on a representative instance of TSPLIB and discuss them in order to support our claims regarding the efficiency and efficacy of the proposed approach.

Keywords: ant-colony optimization · metaheuristics · two-dimensional pheromone.

* This research was funded in part by Polish National Science Centre, Grant no. 2021/41/N/ST6/01776 (GS). For the purpose of Open Access, the author has applied a CC-BY public copyright license to any Author Accepted Manuscript (AAM) version arising from this submission. This research received partial support from the funds assigned to AGH University of Science and Technology by the Polish Ministry of Education and Science (AB, MKD). This research was supported by the PLGrid infrastructure. This version of the contribution has been accepted for publication, after peer review (when applicable) but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: http://dx.doi.org/10.1007/978-3-031-41456-5_35. Use of this Accepted Version is subject to the publisher's Accepted Manuscript terms of use https://www.springernature.com/gp/open-research/policies/accepted-manuscript-terms

1 Introduction

Many efficient optimization algorithms are based on some kind of learning process (e.g., pheromone deposition in Ant Colony Optimization (ACO) or direction change towards the current best global solution in Particle Swarm Optimization). Modern research related to optimization algorithms (especially in the field of evolutionary algorithms) usually focuses on elitist approaches. In particular, the most popular variants of ACO learn only from the most generated feasible solutions, and the information collected is stored in a very simple structure [7,1,13,6].

The aim of our research was to extend the Ant Colony Optimization algorithm to extract knowledge from more than only the top candidate solutions, store it in a more comprehensive form than the standard pheromone table, and, presumably, improve optimization results. Our previous research [11,12] showed that allowing more solutions to improve pheromone trails improves the quality of the final results. However, the classic pheromone model is too simple to properly encode and interpret information from all feasible solutions, thus, we want to extend its structure to make it more meaningful. Using such an approach, the algorithm is able to make better use of computational effort dedicated to preparing feasible solutions and also to gain more information from negative examples. We believe that this kind of approach would increase the diversity of the search, allowing better solutions to be found in shorter time than the reference algorithms.

Multi-dimensional pheromone is an idea already present in the literature, see, e.g. [10,9], however these works are very closely related to the discussed applications (e.g. Vehicle Routing Problem when the authors actually save the local optimization outcomes, very valuable for undertaking the decisions when looking for global solution) and our idea is to propose general algorithms aimed at solving global and multi-criteria optimization problems with ACO.

This research stems from our works on metaheuristics summarized in [2] from the substantial point of view and in [8] from the technical point of view. Further sections of this paper focus on related work regarding Ant Colony Optimization, description of the idea of two-dimensional pheromone in ACO and discussion of the experimenal evaluation of this idea followed by the conclusions and future work.

2 Ant Colony Optimization

The first version of ACO was introduced by Marco Dorigo [4] to solve the Traveling Salesman Problem (TSP). The algorithm was inspired by the behavior of natural ant colonies and the way they share their knowledge. ACO as a metaheuristic algorithm was described a few years later by Dorigo and Caro [5].

The single-objective ACO meta-heuristic algorithm expects the optimization problem to be specified as a graph consisting of a finite set of components (vertices) connected by edges with assigned cost. A valid solution is a path that

respects the restrictions posed and meets the requirements defined by the problem. The cost of a solution is defined as a function of all the costs of all the connections that make up the path. The optimal solution is the valid path with a minimum cost.

The optimization process is based on a population of ants (agents). They iteratively traverse the graph creating candidate solutions. In each iteration, each ant starts from an initial vertex, which can be selected randomly, as in TSP or can be specified by the problem, as in VRP. A probabilistic decision rule (see Eq. 1) is a basis for the ant's decision regarding selection of the next vertex. The rule takes into account the heuristic attractiveness value and the values of pheromone trails left on the edges by previous generations of ants. The heuristic attractiveness (also referenced as desirability) is a function that describes the chances (based on optimization objectives) that the edge will be part of a high-quality solution. For example, in the case of TSP it can be defined as an inverted length of the edge. The pheromone trails indicate how often previous solutions contain specific edges.

The probability of moving from the vertex i to the vertex j of the ant k in iteration t is based on $\tau_{ij}(t)$ the intensity of the trail of pheromones at the edge and $\eta_{ij}(t)$ the heuristic attractiveness of the edge. The parameters α and β control the relative importance of the trail versus heuristic information. The value is relative to the values in all other possible moves A_k.

The probabilistic decision rule is defined as follows:

$$p_{ij}^k(t) = \begin{cases} \frac{\tau_{ij}^\alpha(t)\,\eta_{ij}^\beta(t)}{\sum_{l \in A_k} \tau_{il}^\alpha(t)\,\eta_{il}^\beta(t)} & \text{if } j \in A_k \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The tour ends when a complete feasible solution is found.

Based on the constructed paths, the ants update the pheromone trails. In the classic Ant System (AS) version, the update is performed at the end of each single iteration, controlled by parameters $\rho \in [0, 1)$ – a pheromone persistence coefficient, and m – the number of ants, according to the following formula:

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^k \tag{2}$$

The value of pheromone update for each ant, with the cost of the solution L_k and a constant Q, is defined as follows:

$$\Delta\tau_{ij}^k = \begin{cases} \frac{Q}{L_k} & \text{if } k\text{-th ant uses edge } (i,j) \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where, in the case of TSP, the cost of the tour is simply the length of the route and Q often equals 1.

Since the first version of the ACO algorithm, multiple variations have been proposed to further improve its effectiveness and performance. One type of such

a variation is based on introducing modifications to the original sequential algorithm but without introducing parallelism or distribution. Elitist Ant System (EAS, [7]) modifies Eq. 2, allowing only some ants with the best solutions to update pheromone trails. The rank-based Ant System (ASRank, [1]) is similar to EAS but weights the update left by the specific solution by its rank, so that the best solution modifies the trail most, while the second best modifies it slightly less, etc. Max-Min Ant System (MMAS, [13]) introduced the idea of minimum and maximum bounds of the pheromone trail value that are regularly enforced after performing standard pheromone modifications based on solutions found and evaporation. Ant Colony System (ACS, [6]) proposed two modifications: sometimes allowing an ant to choose the best option available as the next step instead of following the probability formula – Eq. 1 and the so-called local update, that is, decreasing the pheromone value, similar to standard evaporation, right after choosing a specific edge when creating a solution.

The most popular variants of ACO tend to reduce the importance of many solutions produced by the ants in each iteration, or even completely ignore most of them. In this way, they force the ants to focus on the most promising solutions. Although such an approach is natural, it wastes a lot of computational effort. In fact, it does not extract any knowledge from the rest of the proposed solutions [11].

3 Two-dimensional pheromone for Ant Colony Optimization

Despite multiple popular versions of the ACO algorithm, none of them aims at achieving effective knowledge extraction from all feasible solutions. One of the most popular variants of ACO — the Max-Min Ant System [13] – uses only one solution (iteration or global best) to update pheromones. In a colony with 25 ants, this approach discards 96% of the collected data. The experimental results of our previous research showed that larger colonies and more solutions used for the pheromone update improve the final results [11].

To collect more complex information about the feasible solutions created so far, we propose a two-dimensional pheromone structure. Each edge in the graph representing the problem will be connected not with a single value of the pheromone trail strength, but with multiple values representing pheromones left by various feasible solutions. This modification requires novel strategies for the three main components of the ACO algorithm: handling the solutions in the repository, updating the two-dimensional pheromones, and interpreting them during the solution construction.

We actually plan to apply the two-dimensional pheromone for solving multicriteria problems; however, now we would like to apply this idea to increasing the efficiency and efficacy of the single-criteria ACO. Therefore, we introduce many values for the pheromone deposited on the particular edge, and the idea for updating those values closely connects them with a certain order of the solutions produced by the ants. The path with lower cost will be marked closer

to the "top" value in the pheromone, and worse paths will be marked "lower". Therefore, we may include information about not only the best solutions found by ants, but also a wider range of them. Thus, the ant choosing its next step may have much more information than the ant perceiving only the best solution (or actually the solutions gathered in the form of pheromone marking); see Fig. 1.

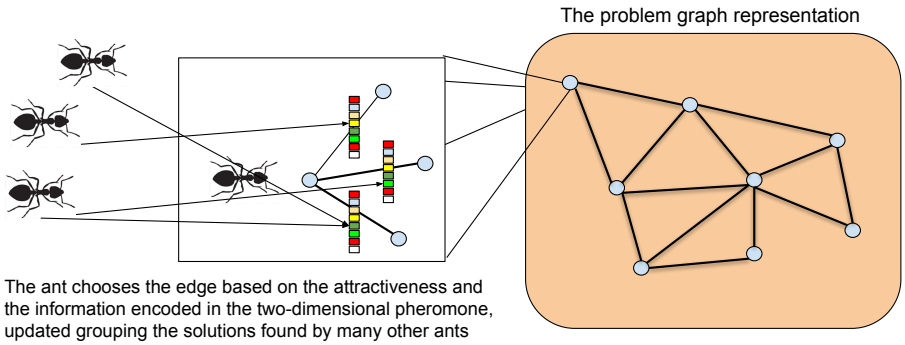

Fig. 1. Idea of functioning of ACO using 2D pheromone. The ants have more information than in the previous versions of ACO to use, i.e. vector of pheromones instead of a single value.

3.1 Depositing the pheromone

In the standard approach (let us call it a one-dimensional pheromone), each edge is associated with a single pheromone value that represents how often this edge was used by the solutions found so far by the colony (usually taking into account only the top solutions). Its value is usually modified by two mechanisms:

- After each iteration, it decreases by some percentage (*extinction*) (the so-called pheromone evaporation or extinction),
- if the edge is part of the solution selected for the pheromone update, it increases by some particular value (*increment*).

In the two-dimensional pheromone, each edge is instead associated with a set of $twoDimPheromoneSize$ values instead of just a single value. Therefore, we have to adjust the above mechanisms to this condition. For this, we need to have a context to evaluate a specific solution with respect to others. Therefore, multiple solutions (possibly all generated by the specific iteration) are passed to the pheromone update procedure and divided into groups based on their evaluation, and each group updates a single value in the two-dimensional pheromone (see Fig. 2).

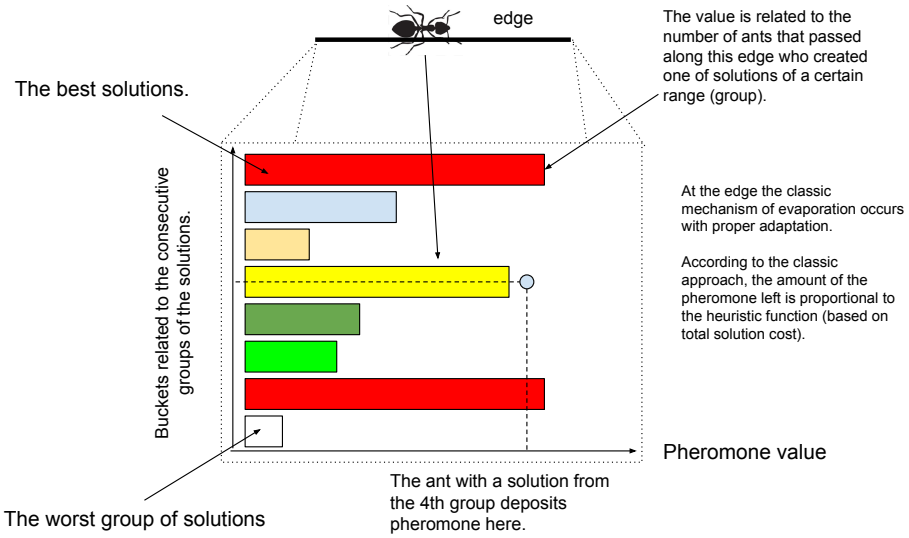

Fig. 2. Structure of the 2D pheromone.

Grouping is done in one of two ways (update types):

- PartFromEvaluation (PFE) — the cost range covered by the solutions is divided into *twoDimPheromoneSize* equal subranges and each solution is assigned to the i-th subrange according to its cost,
- PartFromIndex (PFI) — the solutions are sorted and divided equally into *twoDimPheromoneSize* groups of equal (or almost equal) sizes.

Once the solutions are divided into groups assigned to one of the values of the two-dimensional pheromone, each group updates their part of the pheromone. The value of *increment* (the algorithm parameter) is divided by the size of the group and the result value is added to the current pheromone value for each solution of the group that contains this edge.

The pheromone extinction is simply applied to all pheromone values for each edge in the same way as for the one-dimensional pheromone.

3.2 Interpreting the pheromone information

For a one-dimensional pheromone, the value for the specific edge is inserted directly into the formula 1. In the case of two-dimensional pheromones, we have a set of values instead. The simplest way to bring that to the standard ACO version is to combine these multiple values associated with the specific edge to a single value in some way and put that into the aforementioned formula as if it were the value of a one-dimensional pheromone.

We propose three versions (interpretation types) of reducing multiple values of the pheromone to a single one:

- ExponentialRandom (ER) – from the available values, we choose a single one in such a way that the value updated by the best solutions is taken 50% of the time, the second value is chosen 25% of the time, the next one is taken half as often, and so on,
- WeightedCombination (WC) – the final value is calculated as the weighted product of the values where the first value is assigned the weight of 0.5, the second one – 0.25 and so on (the last two ones are assigned the same weight so that the weights sum up to 1,
- PairingCombination (PC) – this method pairs up the values from the outside towards the center (assuming that $twoDimPheromoneSize$ is even). For each pair, we consider the first one as "positive" (updated by better solutions) and the second one as "negative" (updated by worse solutions), calculate their average and difference, multiply the difference by the decreasing index of the pair (so that the difference is reinforced the most for the extreme pair, i.e., the first and last value of the pheromone, and the least for the "middle pair") and add the average to that. Finally, we compute the average of such values calculated for the pairs. Since this method does not ensure that the ultimate value is within reasonable limits, we ensure that it does not exceed $maxValue$ and the current minimum value of the pheromone $(maxPhValue * (1 - extinction)^{iterationsSoFar})$.

4 Experimental evaluation

In order to evaluate the proposed modifications to the ant colony optimization Algorithm, a new testing framework has been developed, and a series of experiments have been conducted using it.

4.1 Testing framework

The testing framework developed for the purpose of this research was created from scratch and is written in Scala. It is designed to represent the algorithm as an extensible model that contains interchangeable components. With that we aim to support not only running various (including new, experimental) versions of the Ant Colony Optimization algorithm but also solving different types of optimization problem.

For now, the framework supports a standard ACO version as described earlier in the document (called *Basic*) and the new version that uses the proposed two-dimensional pheromone in a few variants, in lieu of the usual "one-dimensional" pheromone. As of now, it is possible to solve TSP, MTSP and CVRP optimization problems, and the research presented here is focused on TSP based on popular benchmarks available from the TSPLIB database. For the sake of clarity, the framework does not apply any local optimizations (e.g. 2-opt) to the created solutions yet.

The framework allows for setting the following common parameters:

- *repeat* – the number of repeated runs for each specific configuration,

- $iterationsNums$ — the number of iterations of the algorithm,
- $minPhValue$ — the minimum value of the pheromone,
- $maxPhValue$ — imposed maximum value of the pheromone,
- $antsNum$ — the number of ants in the colony,
- α — pheromone power in the probabilistic decision rule,
- β — heuristic value power in the probabilistic decision rule,
- $pheromoneType$ — $Basic$ or $TwoDim$ – choice between the standard "one-dimensional" and the experimental two-dimensional pheromone,
- $increment$ — the value used for the pheromone update increment,
- $extinction$ — pheromone extinction fraction,
- $updateNum$ — the number of the best solutions passed to the pheromone update procedure (with -1 meaning all the solutions from the iteration).

When $pheromoneType$ is set to $TwoDim$, there are a few more parameters that can be set:

- $twoDimensionalPheromoneSize$ – the number of values associated with a single edge,
- $interpretationType$ – $ExponentialRandom$, $WeightedCombination$, or $PairingCombination$ – choice among variants of the two-dimensional pheromone interpretation
- $updateType$ – $PartFromEvaluation$ or $PartFromIndex$ – choice between variants of the two-dimensional pheromone update type.

It is possible to define a set of values for each of the parameters and run each combination of those possibilities.

4.2 Experimental results

In order to evaluate the potential of our proposed modification of the ACO algorithm, we have conducted a wide range of experiments with various combinations of algorithm parameters for a single optimization problem – namely $Berlin52$ from $TSPLIB$.

In Figures 3 and 4, we can see the results grouped by the number of ants in the colony for the types of pheromones $Basic$ and $TwoDim$, respectively. They show an important difference between these two types. Since the pheromone does not influence the algorithm much at the beginning, we can see that they both start similarly, giving better results for more ants (better exploration, better chance of finding a better solution). However, in the case of $Basic$, it ends up giving worse final results for larger colonies, which might be interpreted as a faster fall into some local minimum. On the other hand, in the case of $TwoDim$, the results are consistently better for larger colonies for all iterations from the beginning to the very end. It might be interpreted as a sign of more effective knowledge extraction from solutions proposed by the ants. It is also worth noting that even for 20 ants, the average for two-dimensional pheromone is better than $Basic$.

Looking closer at the results generated for various combinations of parameters, we have concluded which values of some basic parameters seem to work

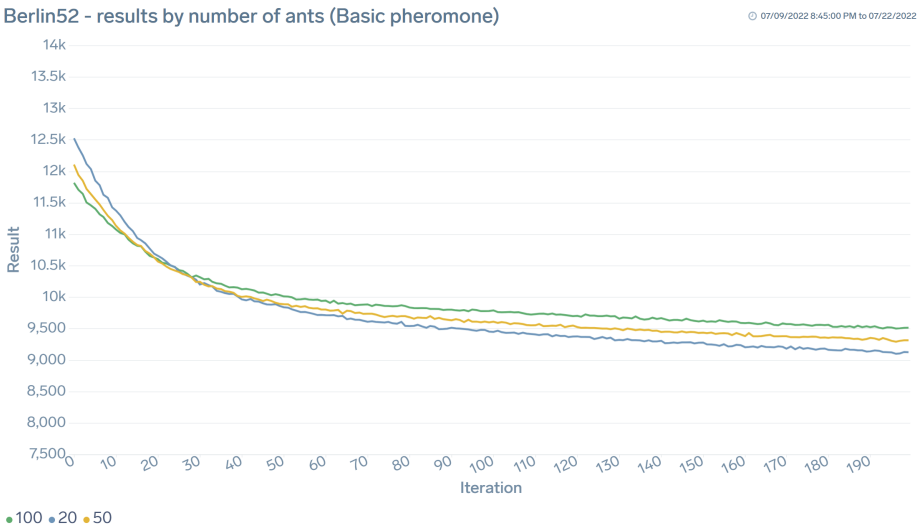

Fig. 3. Average results for various number of ants - algorithm with basic pheromones.

best, that is, 100 ants, both α and β set to 3.0, and pheromones *increment* and *extinction* set to 0.05. In Figure 5 we show average results for some selected combinations of $twoDimPheromoneSize$ and $interpretationType$ from among the runs with the basic parameters set as mentioned above. As we can see, we get the best results for 20 and $ExponentialRandom$ and the worst results for 20 and $PairingCombination$ which also shows very poor convergence.

In Table 1 we list the top configurations based on the results achieved in the last iteration. Each line of the table contains the score for the specific configuration that was calculated as an average from 20 repeats. The abbreviations of the parameter names and values have the following meanings: Inc. – increment; Ext. – extinction; UN – update solution number; TDS – Two-Dimensional Pheromone Size; InterpT. – Interpretation type; UpdateT. – update type; PC – Pairing Combination; ER – Exponential Random; WC – Weighted Combination; PFE – Part From Evaluation; PFI – Part From Index. Based on what can be seen in the table, we make the following observations:

- vast majority of configurations have 100 ants in the colony,
- both α set to 2.0 and 3.0 are popular, β is usually set to 3.0,
- most configurations have *increment* and *extinction* set to 0.05,
- $interpretationType$ set to $ExponentialRandom$ dominate the table, but both $WeightedCombination$ and $PairedCombination$ also show up,
- $updateType$ is always set to $PartFromEvaluation$, which is in alignment with our observation that $PartFromIndex$ gives good results during the optimization but does not converge to them towards the end,
- $Basic$ appears only once, roughly in the middle of the table, and uses only a single best solution from each iteration for pheromone update.

Table 1. Average last iteration results from 20 repeats (*berlin*52)

Ants	α	β	Pheromone	Inc.	Ext.	UN	TDS	InterpT.	UpdateT.	Avg. score
100	2	3	TwoDim	0.05	0.05	50	20	ER	PFE	7641.93
100	3	3	TwoDim	0.05	0.05	50	20	ER	PFE	7656.87
100	2	3	TwoDim	0.05	0.05	50	4	ER	PFE	7699.16
100	3	2	TwoDim	0.05	0.05	50	20	ER	PFE	7702.64
50	2	3	TwoDim	0.1	0.1	25	20	ER	PFE	7705.11
100	2	2	TwoDim	0.05	0.05	50	10	ER	PFE	7711.96
100	3	3	TwoDim	0.1	0.1	50	20	ER	PFE	7715.82
100	2	2	TwoDim	0.05	0.05	50	20	ER	PFE	7719.61
100	3	3	TwoDim	0.05	0.05	50	10	ER	PFE	7719.61
100	2	3	TwoDim	0.05	0.05	50	4	PC	PFE	7734.02
50	3	3	TwoDim	0.05	0.05	25	20	ER	PFE	7740.89
100	2	3	TwoDim	0.05	0.05	-1	20	PC	PFE	7742.79
100	2	2	TwoDim	0.05	0.05	50	4	ER	PFE	7743.03
100	2	2	TwoDim	0.05	0.05	-1	10	ER	PFE	7746.44
100	2	3	TwoDim	0.05	0.05	50	20	WC	PFE	7746.84
100	**2**	**3**	**Basic**	**0.05**	**0.05**	**1**	-	-	-	**7747.77**
100	2	3	TwoDim	0.05	0.05	-1	20	ER	PFE	7748.98
100	2	3	TwoDim	0.1	0.1	-1	20	ER	PFE	7750.97
100	3	3	TwoDim	0.01	0.01	50	10	WC	PFE	7754.45
100	3	3	TwoDim	0.1	0.1	50	10	ER	PFE	7755.76
100	2	2	TwoDim	0.1	0.1	50	4	ER	PFE	7758.33
50	2	3	TwoDim	0.05	0.05	25	20	ER	PFE	7768.05
100	3	3	TwoDim	0.05	0.05	50	4	ER	PFE	7768.28
100	3	3	TwoDim	0.01	0.01	-1	20	PC	PFE	7768.36
100	3	3	TwoDim	0.01	0.01	50	20	WC	PFE	7770.58
100	3	3	TwoDim	0.01	0.01	50	4	PC	PFE	7777.40
50	3	3	TwoDim	0.01	0.01	25	20	WC	PFE	7781.24
100	2	3	TwoDim	0.1	0.1	50	10	ER	PFE	7785.18

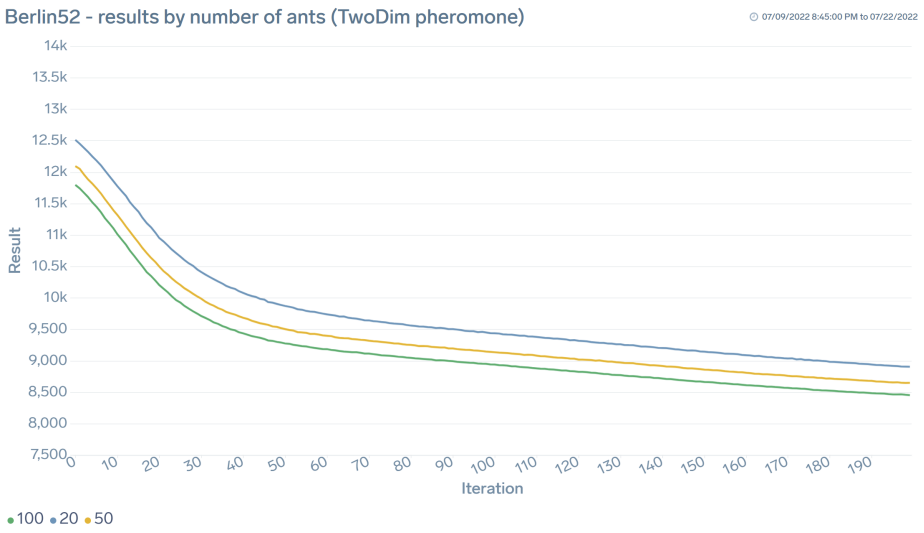

Fig. 4. Average results for various number of ants - algorithm with two dimensional pheromones.

5 Conclusion

In this article we have shown how the introduction of a new pheromone structure (2D pheromone) affects the efficiency and efficacy of ACO applied to solving one of the popular TSP benchmarks. Even though the original algorithm uses all created solutions to update the pheromone, its later modifications reduce that to only top solutions, sometimes only to a single one, and because of that achieve better results. However, our work proves that using more or all solutions is more effective, provided that the information that can be collected from such variety of solutions is encoded in a more advanced structure than just a single value per edge.

The results presented demonstrate that the proposed pheromone makes the algorithm significantly better than its predecessor utilizing the original pheromone structure. We focused in this paper on one of the most popular benchmarks; however, we are planning to publish an extended version of this paper (abridged because of lack of space) in the near future, covering more benchmark functions.

We believe that utilizing a wider range of solutions (not only the best ones) increases the diversity of the search and helps in reaching better solutions earlier. We already developed a method for measuring the diversity of ACO [3,14], however we will apply this method in future to the newly developed algorithms. Moreover, in the future, we will apply 2D pheromone for solving not only single but also multi-criteria optimization problems, encoding e.g. different levels of Pareto front in our 2D pheromone structure, tackling more benchmark and real-life problems. Even though our idea stems from the existing ones cited in this

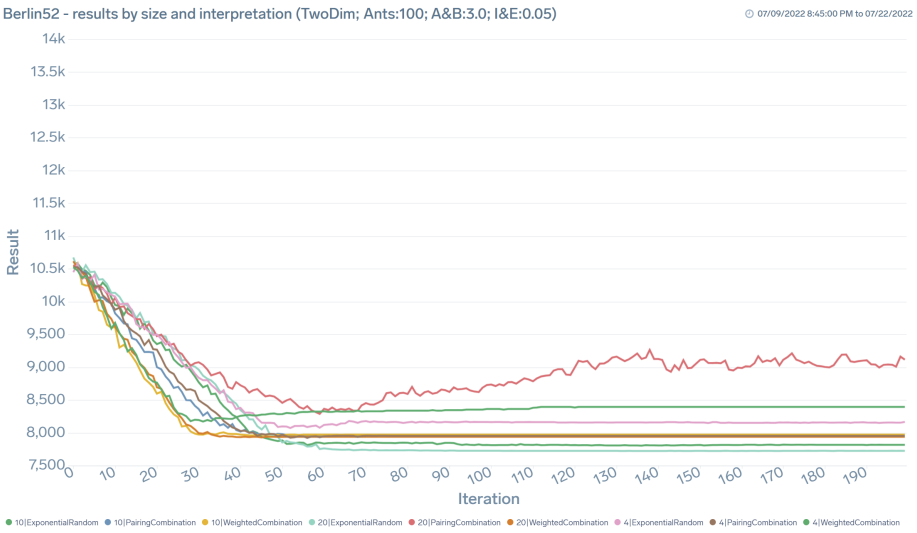

Fig. 5. Average results for various combinations of size and interpretation methods of two dimensional pheromones.

paper, we aim at working-out a general algorithm aimed at solving global and multi-criteria optimization problems with ACO, while the cited papers focus on the applications.

References

1. Bullnheimer, B., Hartl, R., Strauß, C.: A new rank based version of the ant system. a computational study. WorkingPaper 1, SFB Adaptive Information Systems and Modelling in Economics and Management Science, WU Vienna University of Economics and Business (1997)
2. Byrski, A., Kisiel-Dorohinicki, M.: Evolutionary Multi-Agent Systems - From Inspirations to Applications, Studies in Computational Intelligence, vol. 680. Springer (2017). https://doi.org/10.1007/978-3-319-51388-1, `https://doi.org/10.1007/978-3-319-51388-1`
3. Byrski, A., Wegrzynski, K., Radwanski, W., Starzec, G., Starzec, M., Bargiel, M., Urbanczyk, A., Kisiel-Dorohinicki, M.: Population diversity in ant-inspired optimization algorithms. Computer Science **22**(3) (2021). https://doi.org/10.7494/csci.2021.22.3.4301, `https://doi.org/10.7494/csci.2021.22.3.4301`
4. Dorigo, M.: Optimization, learning and natural algorithms. Ph. D. Thesis, Politecnico di Milano (1992)
5. Dorigo, M., Di Caro, G.: Ant colony optimization: a new meta-heuristic. In: Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406). vol. 2, pp. 1470–1477. IEEE (1999)

6. Dorigo, M., Gambardella, L.M.: Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on evolutionary computation **1**(1), 53–66 (1997)
7. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **26**(1), 29–41 (1996)
8. Faber, L., Pietak, K., Byrski, A., Kisiel-Dorohinicki, M.: Agent-based simulation in age framework. In: Byrski, A., Oplatková, Z., Carvalho, M., Kisiel-Dorohinicki, M. (eds.) Advances in Intelligent Modelling and Simulation - Simulation Tools and Applications, Studies in Computational Intelligence, vol. 416, pp. 55–83. Springer (2012). https://doi.org/10.1007/978-3-642-28888-3_3, https://doi.org/10.1007/978-3-642-28888-3_3
9. Guo, N., Qian, B., Na, J., Hu, R., Mao, J.L.: A three-dimensional ant colony optimization algorithm for multi-compartment vehicle routing problem considering carbon emissions. Applied Soft Computing **127**, 109326 (2022). https://doi.org/https://doi.org/10.1016/j.asoc.2022.109326, https://www.sciencedirect.com/science/article/pii/S1568494622005014
10. Li, Y., Soleimani, H., Zohal, M.: An improved ant colony optimization algorithm for the multi-depot green vehicle routing problem with multiple objectives. Journal of Cleaner Production **227**, 1161–1172 (2019). https://doi.org/https://doi.org/10.1016/j.jclepro.2019.03.185, https://www.sciencedirect.com/science/article/pii/S0959652619308790
11. Starzec, M., Starzec, G., Byrski, A., Turek, W.: Distributed ant colony optimization based on actor model. Parallel Computing **90**, 102573 (2019)
12. Starzec, M., Starzec, G., Byrski, A., Turek, W., Pietak, K.: Desynchronization in distributed ant colony optimization in hpc environment. Future Generation Computer Systems **109**, 125–133 (2020)
13. Stützle, T., Hoos, H.H.: Max–min ant system. Future generation computer systems **16**(8), 889–914 (2000)
14. Swiderska, E., Lasisz, J., Byrski, A., Lenaerts, T., Samson, D., Indurkhya, B., Nowé, A., Kisiel-Dorohinicki, M.: Measuring diversity of socio-cognitively inspired ACO search. In: Squillero, G., Burelli, P. (eds.) Applications of Evolutionary Computation - 19th European Conference, EvoApplications 2016, Porto, Portugal, March 30 - April 1, 2016, Proceedings, Part I. Lecture Notes in Computer Science, vol. 9597, pp. 393–408. Springer (2016). https://doi.org/10.1007/978-3-319-31204-0_26, https://doi.org/10.1007/978-3-319-31204-0_26