Palestine Journal of Mathematics

Vol 13(Special Issue I)(2024) , 104–112

A COMPARATIVE STUDY OF ANT COLONY OPTIMIZATION VARIANTS FOR THE PROBABILISTIC TRAVELING SALESMAN PROBLEM

Fadoua El asri, Chakir Tajani and Hanane Fakhouri

MSC 2010 Classifications: Primary 20M99, 13F10; Secondary 13A15, 13M05.

Keywords and phrases: Stochastic combinatorial optimization, Probabilistic traveling salesman problem, Metaheuristics, Ant colony optimization.

Abstract In this paper, we are interested in the probabilistic traveling salesman problem (PTSP) as a variant of the well-known traveling salesman problem (TSP), where city distances follow a probability distribution. The goal is to find a tour that visits all cities with the minimum expected length. To solve this combinatorial problem, we consider four ant colony optimization (ACO) variants: Ant System (AS), Max-Min Ant System (MMAS), a Novel Max-Min System (NMMAS) and Ant Colony System (ACS). The performance of the algorithms is evaluated, considering different examples from the literature, based on the quality of the obtained solution and the computation time. The experiment results show that MMAS is more efficient compared to the other algorithms.

1 Introduction

Optimization problems often involve uncertainty or randomness, such as traffic delays, weather conditions, or breakdowns. Stochastic Combinatorial Optimization (SCOP) studies these problems, which are challenging due to both combinatorial and stochastic complexity. SCOP is important for planning optimal decisions under uncertainty and finding robust and adaptive solutions with economic, environmental, or social impacts. One of the most studied SCOP problems is the Probabilistic Traveling Salesman Problem (PTSP) introduced by Jaillet [1]. The PTSP is a variant of the Traveling Salesman Problem (TSP) where distances between cities are random variables. The objective is to reduce the expected or variance of the total length.

To solve the PTSP, we propose the use of the Ant Colony Optimization (ACO) which is a technique based on the behavior of real ants. ACO builds candidate solutions using pheromone and heuristic information, simulating the process of ants to find short paths between their nest and a food source. The concept of ant algorithms originated in 1959 when Pierre-Paul Grasse introduced stigmergy to explain termite nest building in [2]. Ants use stigmergy to communicate with pheromones and coordinate with each other indirectly. In 1988, Moyson et al presented a paper [3] on self-organization in ants for solving TSP. In 1989, Goss et al studied in [4] the collective behavior of Argentine ants and proposed a mathematical model to describe ant behavior and optimization ability. In 1991, M. Dorigo proposed the Ant System in [5], which was the first algorithm applied to TSP. It uses a population of artificial agents that construct solutions by traveling on the graph of the problem. The algorithm updates the pheromones globally at the end of each iteration, promoting the best solutions found. In 1996, Stützle and Hoos in [6] invented the Max-Min Ant System, which improves the Ant System by a more effective global pheromone update. In 1997, Gambaredlla and Dorigo published the Ant Colony System [7], which improves the Ant System by a local pheromone update and a more effective transition rule.

Several works have been done to improve the ant colony algorithms for the PTSP, using local search techniques, learning mechanisms, variants of pheromone, or hybrid approaches with other methods. Bianchi et al. introduced in [8] an Ant Colony Optimization (ACO) algorithm for the PTSP. This algorithm employs a cost function that factors in the probabilities of city presence and a transition rule that favors cities with high presence probabilities. In addition, it uses a local search to enhance the solutions. Guntsch and Middendorf proposed in [9] a population-

based approach for ACO, utilizing several ant colonies that work together to discover solutions. Their mechanism of information exchange between colonies is based on natural selection and ant migration. Branke and Guntsch proposed in [10] an enhanced ACO algorithm for the PTSP, utilizing an approximate evaluation and new heuristics. Weyland et al. in [11] presented an ACO algorithm for the PTSP by combining ACS with a local search based on variable neighborhood search.

In this work, to solve the PTSP, we consider four ant colony algorithms based on ACO: Ant System (AS), Max-Min Ant System (MMAS), Novel Max-Min system (NMMAS) and Ant Colony System (ACS). These algorithms differ in how they select the edges to add to the solution, how they update pheromones and how they manage solution diversity. So, A comparative study is done to evluate their performance.

The paper is organized as follows: in section 2, we present the mathematical formulation of the PTSP. we present the four ant algorithms. A comparative study of the considered algorithms is performed in section 3. Experimental results obtained by the algorithms for different instances of the PTSP are discussed in section 4.

2 Mathematical Model

In the probabilistic traveling salesman problem (PTSP), we have V a set of n cities, each with a probability of presence and a distance. $d : V \times V \to \mathbb{R}^+$ the distance function gives the distance between any pair of cities, and the probability function $p : V \to [0, 1]$ gives the probability that a city is present. The presence of different cities are independent events.

The PTSP belongs to the class of a priori optimization problems, where we have to determine the order of the cities before knowing which ones are present. An a priori solution is a tour that visits all cities exactly once, as in the TSP, which is a permutation $\tau : \langle n \rangle \to V$ of the cities. Then, we obtain an a posteriori solution by removing the absent cities from the a priori tour, while keeping the order given by it. The goal is to find an a priori tour that minimizes the average length of the a posteriori tours, according to the given probabilities.

Thus, the problem can be defined as follows [12]:

Let V be a set of size n, $d : V \times V \to \mathbb{R}^+$ be a distance function and $p : V \to [0, 1]$ be a probability function. The problem is to find a permutation $\tau^* : \langle n \rangle \to V$ that minimizes the function $f_{ptsp}(\tau)$ over all permutations $\tau : \langle n \rangle \to V$.

The function $f_{\text{ptsp}}(\tau)$ is calculated as the sum of the product of the length $L_\tau(S)$ and the probability $P(S)$ for each subset $S \subset V$ of cities [8]:

$$f_{\text{ptsp}}(\tau) = \sum_{S \subseteq V} P(S) L_\tau(S) \tag{2.1}$$

where $P(S)$ is the product of probabilities p_i for the cities in S and $(1 - p_i)$ for cities outside of S [8]:

$$P(S) = \prod_{i \in S} p_i \prod_{i \in V - S} (1 - p_i) \tag{2.2}$$

The PSTP can be solved using mathematical models that estimate the probability and expected cost of each possible arc (i, j) in the circuit, depending on whether or not the cities i and j are present, and the possible absence of the cities k between them ($k = i + 1, ..., j - 1$). Such a model is considered the Campbell equation in [13], which adds three terms for different scenarios in the sequence:

$$\sum_{j=1}^{n} p_j d_{0j} \prod_{k=1}^{j-1} (1 - p_k) + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} p_i p_j d_{ij} \prod_{k=i+1}^{j-1} (1 - p_k) + \sum_{i=1}^{n} p_i d_{0i} \prod_{k=i+1}^{n} (1 - p_k) \tag{2.3}$$

The equation (2.3) allows us to find the optimal circuit that minimizes the total expected cost.

3 Variants of Ant Colony Optimization for PTSP

In this section, we examines four ant algorithms: Ant system (AS), Ant Colony System (ACS), Max-Min Ant System (MMAS)and Novel Max-Min Ant System (NMMAS). These algorithms use Ant Colony Optimization (ACO) to generate candidate solutions by utilizing pheromone levels and heuristic information through a probabilistic rule. Pheromone is a chemical substance that artificial ants deposit on graph edges, reflecting the quality of solutions that employ these edges. Heuristic information measures edge attractiveness based on the problem at hand. Artificial ants construct solutions by selecting edges based on these two factors. The focus is on the effectiveness of these algorithms in resolving the PTSP, a probabilistic combinatorial optimization problem.

3.1 Ant System principle for the PTSP

At each iteration t ($1 \leq t \leq t_{max}$), each ant k ($k = 1, ..., m$) builds a full path of $n = |V|$ steps by traversing the graph. The choice of the next city to visit depends on several factors, including the set of feasible cities S_i^k, the visibility or heuristic value between cities $\psi_{ij} = \frac{1}{d_{ij}}$, and the amount of pheromones on the connecting edge θ_{ij}. We use stochastic transition rules to determine the probability that an ant moves from one city to another, which is given by the formula mentioned in [14]:

$$r_{ij}^k(t) = \begin{cases} 0 & \text{if } j \notin S_i^k \\ \frac{(\theta_{ij}(t))^\alpha (\psi_{ij})^\beta}{\sum_{l \in S_i^k} (\theta_{il}(t))^\alpha (\psi_{il})^\beta} & \text{if } j \in S_i^k \end{cases} \tag{3.1}$$

The parameters α and β control the relative importance of pheromone trail $\theta_{ij}(t)$ and visibility ψ_{ij}. A balance between these two criteria must be found to avoid premature selection of a path, affecting diversification and intensification behaviors. If $\alpha = 0$, only visibility is considered, while if $\beta = 0$, only pheromone trails influence the decision.

After building a complete path, each ant deposits pheromones on the edges of its path [15]:

$$\Delta\theta_{ij}^k(t) = \begin{cases} 0 & \text{if } (i,j) \notin P^k(t) \\ \frac{R}{f_{\text{ptsp}}^k(\tau)} & \text{if } (i,j) \in P^k(t) \end{cases} \tag{3.2}$$

where the route of ant k at step t is $P^k(t)$, R is a standard value in the literature. The evaporation process then takes place, allowing the system to "forget" bad solutions. The pheromone update rule is applied at every step by this equation in [15] :

$$\theta_{ij}(t+1) = (1-\rho)\theta_{ij}(t) + \Delta\theta_{ij}(t) \tag{3.3}$$

where $\Delta\theta_{ij}(t) = \sum_{k=1}^m \Delta\theta_{ij}^k(t)$, and $\rho \in [0; 1]$ is the evaporation parameter.

The Ant Colony Optimization (ACO) algorithm operates in cycles. At the beginning, m ants are randomly placed at city 0 (depot). Each ant's memory list contains only its starting city. The pheromone trails are initialized to a small positive constant c, where $\theta_{ij}(0) = c$. All ants complete their trip and return to their starting city (0) based on Equation (3.1). Each ant computes its expected path length $f_{\text{ptsp}}^k(\tau)$ and the pheromone variables $\theta_{ij}^k(t)$ are calculated. The pheromone variables $\theta_{ij}(t)$ are updated based on formula (3.3), where the ant retraces its trip in reverse while leaving pheromone. The ant with the minimum expected path length is identified, and its trip is memorized if it is better than the best trip so far. The ants' memories (list of visited cities) are erased, and they start a new tour from city 0. The algorithm stops after a fixed number of cycles NC_{max} or when all ants make the same tour (stagnation). The algorithm returns the best memorized tour.

The global complexity of the algorithm is $O(n^2 \cdot m)$. If we assume that $m = n$, the complexity becomes $O(NC_{max} \cdot n^3)$.

3.2 Ant Colony System principle for the PTSP

The Ant Colony System (ACS) algorithm was proposed by Grambardella and Dorigo in 1996 to enhance the performance of Ant Colony Optimization (ACO) [14]. The ACS algorithm balances exploration and exploitation using state transition and pheromone updating rules, allowing ants to construct tours influenced by both heuristic and pheromone information. The ACS algorithm introduces several modifications to ACO:

- ACS introduces a transition rule that depends on a parameter q_0 ($0 \leq q_0 \leq 1$), which defines a trade-off between diversification and intensification. An ant k on a city i will choose a city j by the rule as we can find in [14]:

$$j = \begin{cases} \arg\max_{l \in S_i^k} \{\theta_{il}(t)^\alpha \psi_{il}^\beta\} & \text{if } q \leq q_0 \\ r_{ij}^k(t) & \text{otherwise} \end{cases} \tag{3.4}$$

  where, S_i^k is the set of cities that ant k can visit from city i, $\theta_{il}(t)$ is the amount of pheromone on the edge (i, l) at time t, ψ_{il} is the reciprocal of the distance among cities i and l (heuristic value). q is a random variable uniformly distributed over $[0, 1]$.

- The control of trails is divided into two levels: local update and global update. During the local update, each ant leaves a trail according to the rule [14]:

$$\theta_{ij}(t+1) = (1-\rho)\theta_{ij}(t) + \rho\theta_0 \tag{3.5}$$

  where θ_0 is the initial value of the trail. At each visit, the edges that are visited see their amount of pheromone decrease, favoring diversification by considering unexplored paths. At each iteration, the global update is performed as follows [14]:

$$\theta_{ij}(t+1) = (1-\rho)\theta_{ij}(t) + \rho\Delta^{best}\theta_{ij}(t) \tag{3.6}$$

In the ACS algorithm, only the best path is updated, which contributes to an intensification by selecting the best solution, as [14] show:

$$\Delta\theta_{ij}^{best}(t) = \begin{cases} \frac{R}{f_{\text{ptsp}}^{best}(t)} & \text{if edge } (i,j) \text{ belongs to the best tour} \\ 0 & \text{otherwise} \end{cases}$$

The initial amount of pheromones on each edge is a uniform distribution of a small amount and $\rho \geq 0$.

3.3 Max-Min Ant System principle for the PTSP

The Max-Min Ant System (MMAS) algorithm is a variation of the ACO algorithm that aims to improve its performance in combinatorial optimization problems. It was suggested by Stützle and Hoos in 1997 [16]. MMAS exploits the best solutions, avoids stagnation, enhances exploration, prevents dominance, and delays convergence, in the three way aspects:

- **Exploitation of Best Solutions:** MMAS updates the pheromone trails only with the best ant in each iteration or overall, enhancing the exploitation of promising solutions.
- **Pheromone Trail Limits:** MMAS bounds the pheromone values by θ_{max} and θ_{min}, maintaining pheromone diversity and preventing premature convergence.
- **Pheromone Trail Initialization:** MMAS initializes the pheromone values to θ_{max} after the first iteration, encouraging exploration in the early stages. The probabilistic choice rule for choosing a solution component is the same as in the Ant System algorithm ,see equation (3.1)

The pheromone update rule for MMAS is different from AS in two aspects: First, MMAS only updates the pheromone with best solution discovered until now, either the iteration-best or the global-best solution. Second, MMAS limits the pheromone values to a range of $[\theta_{min}, \theta_{max}]$ to avoid stagnation. The update rule given in [17] is defined as follows:

$$\theta_{ij} = (1-\rho)\theta_{ij} + \Delta\theta_{ij}^{\text{best}} \tag{3.7}$$

where ρ is the evaporation rate and $\Delta\theta_{ij}^{\text{best}} = R/f_{\text{ptsp}}(\tau_{best})$ if edge (i,j) belongs to the best solution S^{best}, and 0 otherwise. $f_{\text{ptsp}}(\tau_{best})$ is the expected path length of the best solution.

The lower and upper bounds of the pheromone values are determined in [17] by:

$$\theta_{\text{max}} = \frac{1}{1-\rho}\frac{1}{f_{\text{ptsp}}(\tau_{gb})} \tag{3.8}$$

$$\theta_{\text{min}} = \frac{\theta_{\text{max}}(1-\sqrt[n]{p_{\text{best}}})}{(\text{avg}-1)\sqrt[n]{p_{\text{best}}}} \tag{3.9}$$

Where $f_{\text{ptsp}}(\tau_{gb})$ is the value of the global-best solution, $avg = n/2$, and p_{best} is a parameter that controls the convergence speed.

The maximum possible pheromone value asymptotically converges to:

***Proposition:*[17]**

$$\lim_{t\to\infty} \theta_{ij}(t) = \tau_{ij} \leq \frac{1}{1-\rho}\frac{1}{f_{\text{ptsp}}(\tau_{opt})}$$

where $f_{\text{ptsp}}(\tau_{opt})$ is the optimal solution value for a specific problem.

3.4 Novel Max-Min Ant System principle for the PTSP

The Novel Max-Min Ant System (NMMAS) algorithm enhances the performance of the Ant Colony Optimization (ACO) algorithm in combinatorial optimization problems. NMMAS differs from the previous ACO variants in its pheromone management, which has two phases. We explore these differences in detail, emphasizing key aspects that underline NMMAS innovation:

- **Pheromone Bounds:** In contrast to traditional methods in MMAS, NMMAS dynamically determines pheromone bounds θ_{max} and θ_{min} based on the sampled tour lengths, offering adaptability to problem complexity and solution progress using a formula [18]:

$$\theta_{max} = \begin{cases} 0.1 & \text{if } f_{\text{ptsp}} \leq 2000 \\ 0.01 & \text{if } 2000 < f_{\text{ptsp}} \leq 20000 \\ \dots \end{cases}$$

  and $\theta_{min} = m_1\theta_{max}$, where m_1 is a real number between 0 and 1.

- **Adaptive Pheromone Management:** NMMAS consists of two stages: in the first one, several solutions are used to update the pheromones, enhancing the exploration; in the second one, only the best solution is used, improving the exploitation. Based on this, we use the following rule given in [18] to update the pheromones by stages:

  If $NC \geq m_2 N_{max}$, where NC is the current iteration number, m_2 is a real number between 0 and 1, and N_{max} is the maximum number of iterations, then the first l solutions of the iteration are used to update the pheromones, using Equation (3.3) (m is replaced by l in Equation (3.3)), where:

$$\Delta\theta_{ij}^{k}(t) = \begin{cases} \frac{\rho\theta_{max}}{k} & \text{if ant } k \text{ travels on edge } (i,j) \\ 0 & \text{otherwise} \end{cases}.$$

If $NC > m_2 N_{max}$, then only the best solution of the iteration is used to update the pheromones, using Equation (3.6), where $\Delta\theta_{ij}^{best}(t)$ is the amount of pheromone deposited by the best ant on edge (i, j) at time t, defined as:

$$\Delta\theta_{ij}^{best}(t) = \begin{cases} \frac{R}{f_{\text{ptsp}}(\tau_{best})} & \text{if edge } (i,j) \text{ belongs to the best tour} \\ 0 & \text{otherwise} \end{cases}.$$

This dynamic balance between exploration and exploitation elevates NMMAS's effectiveness in navigating complex solution landscapes.

The probabilistic selection process for choosing a solution component follows the same approach as the Ant System algorithm, as depicted in equation (3.1).

4 Computation experiment

In this section, we evaluate the performance of the four considered ACO algorithms for PTSP. We use different instances that are given in [13]. We generate the probabilities of the customers according to four scenarios: random between 0 and 1, random between 0.1 or 1, fixed at 0.9, and fixed at 0.1.

We set the evaporation rate to 0.5 for all algorithms, which ensures a good balance between exploration and exploitation. We set the beta to 2 which gives a moderate importance to the distance between cities. We set the number of ants to 7, wich is reasonable for the PTSP. We set the constant R to 1 for all algorithms. We set the q_0 to 0.9 for ACS, which favors the pseudo-random proportional rule, which is more deterministic and exploits more the best solutions. We set the l to 3 for NMMAS, which means that we use three best local solutions per iteration. We set the m_2 to 0.3 for NMMAS, which means that we consider that an edge is part of a best local solution if it is chosen by at least 30% of the ants. We fixed the N-max in 30 which seems suitable for the PTSP.

The initial value of the pheromone in ACO algorithms is varied between 1 and 0.01 depending on the instances of the PTSP. A value of 1 is used for instances with high probabilities (0.9) as it does not influence the choice of ants at the beginning of the search. A value of 0.01 is used for instances with low or random probabilities (0.1 and between 0.1 or 1) to favor exploration of new solutions at the beginning of the search.

We measured the computation time (CPU in seconds) required to run each algorithm, and evaluated the quality (expected values [E.V]) of the obtained solutions. Our approach is developed entirely in Python and executed on a machine equipped with an Intel Core i5-7200U processor at 2.50 GHZ, 8GB of RAM, and a 64-bit operating system with an x64 processor.

Table 1. Comparative analysis of ACO variants for the PTSP by probability Range.

	Probability Range							
	AS		MMAS		N-MMAS		ACS	
Data set	E.V	CPU	E.V	CPU	E.V	CPU	E.V	CPU
22	110.1561	0	106.6246	2	106.5561	0	157.75202	0
42	127.8422	3	127.4942	7	141.7978	3	374.6267	3
62	286.0540	9	202.6307	16	241.4319	9	840.3254	62
102	322.4177	38	334.3339	49	411.8957	36	1197.1593	40
152	578.73705	117	550.6243	134	710.1083	131	1116.7980	128

Table 1 shows that, using the Range probability, MMAS and AS are more efficient than ACS by 10% and 5% respectively, based on the average E.V. However, MMAS and ACS use more CPU than AS and N-MMAS by 15% and 4% respectively, based on the average CPU.

Table 2. Comparative analysis of ACO variants for the PTSP by probability Mixed.

	Probability Mixed							
	AS		MMAS		N-MMAS		ACS	
Data set	E.V	CPU	E.V	CPU	E.V	CPU	E.V	CPU
22	58.02704	0	55.1686	2	57.1969	0	125.2669	0
42	94.6141	3	97.8969	4	125.3496	3	302.4840	3
62	176.75912	6	150.4233	6	171.8438	5	645.6415	5
102	298.2707	20	324.5693	23	393.77123	22	1166.9133	19
152	442.7206	64	487.5452	66	671.6842	67	1074.6954	59

According to Table 2, using the mixed probability, MMAS and AS are more efficient than ACS by 9% and 7% respectively, based on the average E.V. However, MMAS and ACS use less CPU than AS and N-MMAS by 30% and 25% respectively, based on the average CPU.

Table 3. Comparative analysis of ACO variants for the PTSP by probability 0.1.

	Probability 0.1							
	AS		MMAS		N-MMAS		ACS	
Data set	E.V	CPU	E.V	CPU	E.V	CPU	E.V	CPU
22	86.7580	0	86.7520	0	87.90634	0	90.6209	0
42	156.1326	3	152.76	7	157.8014	3	173.9035	3
62	177.1796	9	177.1742	17	182.6541	10	209.0126	9
102	269.6032	44	260.2483	56	287.1617	44	311.3016	41
152	379.0038	138	368.0143	157	395.5722	137	428.7441	129

We can observe in Table 3, based on the average E.V, MMAS and AS outperform ACS by 12% and 9% respectively when using the probability 0.1. However, based on the average CPU, MMAS and ACS consume more CPU than AS and N-MMAS by 4% and 2% respectively.

Table 4. Comparative analysis of ACO variants for the PTSP by probability 0.9.

	Probability 0.9							
	AS		MMAS		N-MMAS		ACS	
Data set	E.V	CPU	E.V	CPU	E.V	CPU	E.V	CPU
22	262.1323	0	242.045	2	268.0041	0	416.7409	0
42	530.4527	2	522.2926	3	565.7910	2	989.5463	2
62	733.8831	5	668.4563	8	882.8770	5	1330.6987	5
102	1245,3260	21	1139.6283	26	1381.3755	21	2124.0433	20
152	1708.7006	64	1674.9990	89	2062.8824	67	2935.6452	62

Table 4 reveals that, using the probability 0.9, MMAS and AS achieve a higher average E.V than ACS by 42% and 39% respectively. However, MMAS and ACS have a lower average CPU than AS and N-MMAS by 44% and 43% respectively.
This results show that MMAS and AS are the most efficient algorithms for different probabilities. ACS has the highest average E.V, but it also consumes the most CPU time.

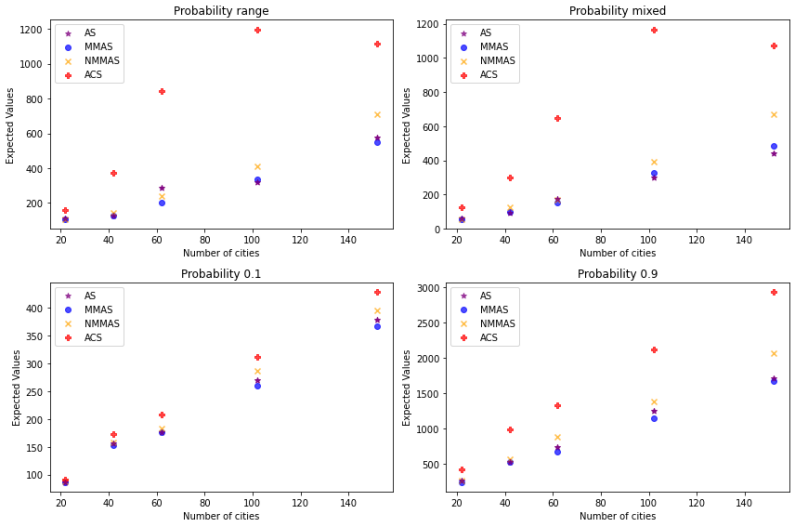

Figure 1. A Comparative Analysis of ACO Variants for the PTSP Based on Expected Values.

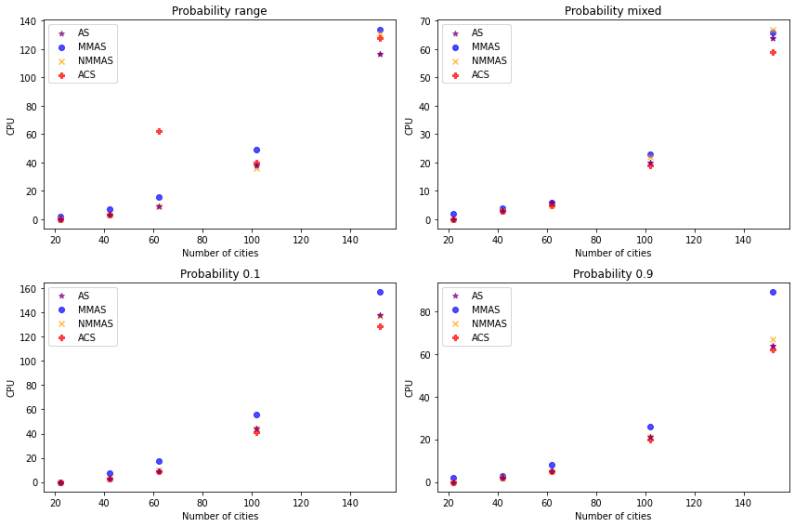

Figure 2. A Comparative Analysis of ACO Variants for the PTSP Based on CPU values.

Figure 1 and figure 2 show the results of a comparative analysis of ACO variants for the PTSP by probability Range, Mixed, 0.1, and 0.9. The analysis indicates that MMAS and AS are the most efficient variants for the PTSP, regardless of the type of probability used. They achieve the best expected values while using a reasonable amount of CPU time. In contrast, ACS is the least efficient variant, as it achieves the worst expected values. N-MMAS is an intermediate variant, achieving average expected values while using a low amount of CPU time.

5 Conclusion

In this this paper, we have compared four algorithms inspired by ant colonies to solve the probabilistic traveling salesman problem (PTSP), which consists of finding the best route to visit cities whose demand is random. The algorithms are evaluated according to the expected value of the total cost of the trip and the computation time. we have analyzed the effect of the probability distribution of the cities on the difficulty of the problem and the performance of the algorithms. The results show that the Max-Min Ant System (MMAS) and Ant System (AS) algorithms are more efficient than the Ant Colony System (ACS) and Novel Max-Min Ant System (N-MMAS)

algorithms in all cases, because they have a better balance between exploration and exploitation.

References

[1] P. Jaillet, Probabilistic traveling salesman problems (Doctoral dissertation, Massachusetts Institute of Technology) (1985).

[2] Grassé and Plerre-P, Solution of a large-scale traveling-salesman problem.La reconstruction du nid et les coordinations interindividuelles chez Bellicositermes natalensis et Cubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs, *the Insectes sociaux* **6**, 41–80 (1959).

[3] B. Manderick and F. Moyson, The Collective behaviour of Ants: An Example of Self-Organisation in Massive Parallelism (1988).

[4] S. Goss, S. Aron, J. Deneubourg and J. Pasteels, Self-organized shortcuts in the Argentine ant, *Naturwissenschaften* **76(12)**, 579–581 (1989).

[5] M. Dorigo, Optimization, learning and natural algorithms, Ph. D. Thesis, Politecnico di Milano (1992).

[6] M. Dorigo, V. Maniezzo and A. Colorni, Ant system: optimization by a colony of cooperating agents, *IEEE transactions on systems, man, and cybernetics, part b (cybernetics)* **26(1)**, 29–41(1996).

[7] M. Dorigo and L. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, *IEEE Transactions on evolutionary computation* **1(1)**, 53–66 (1997).

[8] L. Bianchi, L. Gambardella and M. Dorigo, An ant colony optimization approach to the probabilistic traveling salesman problem, In Parallel Problem Solving from Nature—PPSN VII: 7th International Conference Granada, Spain, *Springer Berlin Heidelberg*, 883–892 (2002).

[9] M. Guntsch and M. Middendorf, A population based approach for ACO, *Springer Berlin Heidelberg*, 72–81(2002).

[10] J. Branke and M. Guntsch, Solving the probabilistic TSP with ant colony optimization, *Journal of Mathematical Modelling and Algorithms* **3**, 403–425 (2004).

[11] D. Weyland, R. Montemanni and L. Gambardella, An enhanced ant colony system for the probabilistic traveling salesman problem. In Bio-Inspired Models of Network, Information, and Computing Systems, *Springer International Publishing* **7**, 237–249 (2014).

[12] D. Weyland, Stochastic vehicle routing: from theory to practice, Doctoral dissertation, Università della Svizzera italiana (2013).

[13] A.Campbell, and B. Thomas, Probabilistic traveling salesman problem with deadlines, *Transportation Science* **42(1)**, 1–21 (2008).

[14] M. Dorigo and L. Gambardella, Ant colonies for the travelling salesman problem, *biosystems* **43(2)**, 73–81 (1997).

[15] J. Dréo, Adaptation de la méthode des colonies de fourmis pour l'optimisation en variables continues. Application en génie biomédical, PhD thesis (2004).

[16] T. Stützle and H. Hoos, MAX-MIN ant system and local search for the traveling salesman problem, *In Proceedings of 1997 IEEE international conference on evolutionary computation (ICEC'97)*, 309–314 (1997).

[17] T. Stützle and H. Hoos, MAX–MIN ant system. *Future generation computer systems* **16(8)**, 889-914 (2000).

[18] Z. Zhang and Z. Feng, A novel max-min ant system algorithm for traveling salesman problem, *In 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems* **1**, 508–511 (2009).

Author information

Fadoua El asri, Chakir Tajani and Hanane Fakhouri, SMAD Team, Polydisciplinary Faculty of Larache, Abdelmalek Essaadi University, Tetouan, Morocco.
E-mail: elasri.fadoua@etu.uae.ac.ma