Qubahan Academic Journal
گوڤارا قبهان یا ئەکادیمی
مجلة قبهان الأكاديمية
Doi: 10.48161/Issn.2709-8206

Review of Swarm Intelligence for Solving Symmetric Traveling Salesman Problem

Awaz Ahmad Shaban
Researcher,
Duhok, Kurdistan Region - Iraq

Jayson A. Dela Fuente
Northern Negros State College of Science Technology, Philippines

Merdin Shamal Salih
Researcher,
Duhok, Kurdistan Region - Iraq

Resen Ismail Ali
Researcher,
Duhok, Kurdistan Region - Iraq

https://doi.org/10.48161/qaj.v3n2a141

Abstract: Swarm Intelligence algorithms are computational intelligence algorithms inspired from the collective behavior of real swarms such as ant colony, fish school, bee colony, bat swarm, and other swarms in the nature. Swarm Intelligence algorithms are used to obtain the optimal solution for NP-Hard problems that are strongly believed that their optimal solution cannot be found in an optimal bounded time. Travels Salesman Problem (TSP) is an NP-Hard problem in which a salesman wants to visit all cities and return to the start city in an optimal time. In this article we are applying most efficient heuristic based Swarm Intelligence algorithms which are Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Bat algorithm (BA), and Ant Colony Optimization (ACO) algorithm to find a best solution for TSP which is one of the most well-known NP-Hard problems in computational optimization. Results are given for different TSP problems comparing the best tours founds by BA, ABC, PSO and ACO.

Keywords: Swarm Intelligence, NP-Hard problem, Travels Salesman Problem (TSP), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Bat algorithm (BA), Ant Colony Algorithm (ACO)

I. INTRODUCTION:

Swarm Intelligence algorithms are computational intelligence techniques studies the collective behavior in decentralized systems. Such systems are made up of a population of simple individual's agents interacting locally with each other and with the environment around themselves [1]. This paper focuses on the comparative analysis of most successful methods of optimization techniques inspired by Swarm Intelligence (SI): Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO), Bat Colony Optimization (BA), and Artificial Bee Colony (ABC) [2]. Ant Colony Optimization (ACO) is a population based optimization algorithm developed by Marco Dorigo as an inspiration of the behavior of ants in finding the optimal way (best path) between their nest and a food source [3]–[5]. The Bat algorithm is a natural inspired metaheuristic algorithm for global optimization problems[6] . It was inspired by the echolocation behavior of microbats, with varying pulse rates of emission and loudness. The Bat algorithm (BA) was developed by the scientist Xin-She Yang [7].

Artificial bee colony algorithm (ABC) is a natural inspired metaheuristic optimization algorithm based on the intelligent foraging behavior of honey bee swarm, proposed by the scientist Karaboga for solving combinatorial optimization problems[8]. Particle Swarm Optimization (PSO) is a population-based optimization algorithm developed by Eberhart and Kennedy as an inspired by bird flocks' behavior when searching for food [9], [10]. The travel salesman problem (TSP) which is an NP-hard problem that is impossible to find the optimal tour with in an optimal time has been studied extensively over the past several decades. In this paper ACO and PSO are used to find the solution of TSP[11]–[14].

II. Swarm Intelligence algorithms

Swarm intelligence (SI), which is an artificial intelligence (AI) field, is concerned with the designing of intelligent interactive multi-agent systems that cooperate to gather to achieve a specific goal. Swarm intelligence is defined by Dorigo M as "The emergent collective intelligence of groups of simple agents"[15]. Swarm-based algorithms are inspired from behaviors of some social living beings (insects, animal, and bacteria's) in the nature, such as ants, birds, bats, bees, termites, and fishes. The most remarkable features of swarm systems are Self-organization and decentralized control that naturally leads to an emergent behavior in the colony. Emergent behavior is an interactive property that emerges a local interaction among all system components (agents) and it is not possible to be achieved alone by any agent in the system [16]. In computer science there are many algorithms that are designed as an inspiration of real collective behavior systems in the nature, swarm intelligence algorithms includes Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Artificial Immune System, Bat algorithm, Bacterial Foraging, Stochastic diffusion search, Glowworm Swarm Optimization, Gravitational search algorithm, Cat Swarm Optimization, and other optimization algorithms[17].

Swarm intelligence works on two basic principles: self-organization and stigmergy (e.g., Fig. 1).

1. Self organization: This can be characterized by three parameters like structure, multi stability and state transitions. In swarms, interpreted the self-organization through four characteristics: (i) positive feedback,(ii) negative feedback, (iii) fluctuations, and (iv) multiple interactions.
2. Stigmergy: It means stimulation by work. Stigmergy is based on three principles:(i) work as a behavioral response to the environmental state; (ii) an environment that serves as a work state memory(iii)work that does not depend on specific agents.

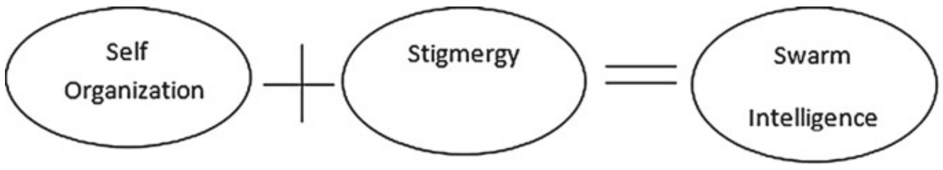

Figure 1 SI component

A. Categories of SI

Swarm intelligence refers to the collective behavior of decentralized, self-organized systems composed of numerous interacting individuals or agents. These systems often exhibit emergent properties and can solve complex problems through simple interactions. While there are various ways to categorize swarm intelligence, the main categories shown in fig(2)[18]:

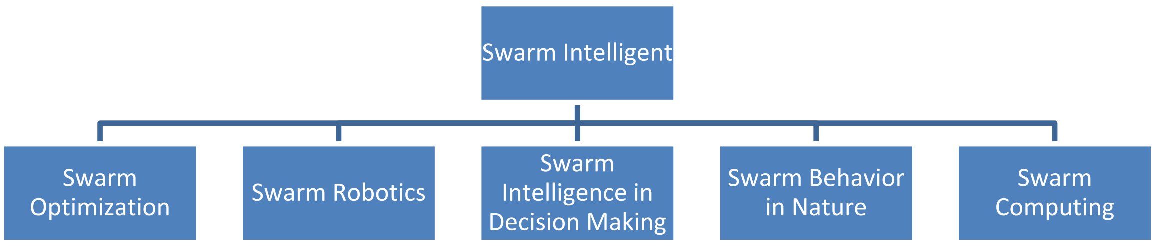

Figure 2 Swarm Intelligent Categories

1. Swarm Optimization: This category focuses on using swarm intelligence to optimize functions, find optimal solutions, or search for the best configuration in a given problem space. Swarm optimization algorithms, such as Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO), utilize the collective behavior of agents to explore and exploit the search space efficiently. Examples of SI algorithms belongs to Swarm Optimizations includes:
   - Particle Swarm Optimization (PSO): PSO is an optimization algorithm inspired by the collective behavior of bird flocks or fish schools. It has been used in various optimization problems, such as finding optimal solutions in engineering design or parameter optimization in machine learning algorithms[19].
   - Ant Colony Optimization (ACO): ACO simulates the foraging behavior of ants to solve optimization problems. It has been applied to tasks like the traveling salesman problem, vehicle routing, and network routing optimization[4], [20].
2. Swarm Robotics: Swarm robotics involves the study of large groups of relatively simple robots that work together to achieve specific tasks. These robots often communicate and coordinate their actions using local interactions, mimicking the behavior of natural swarms, such as ants or bees. Swarm robotics aims to develop

robust and scalable systems that can perform tasks collectively, even when individual robots have limited capabilities. Examples of SI algorithms belongs to Swarm Robotics includes:

- Kilobots: Kilobots are a swarm of simple, low-cost robots developed at Harvard University. They can communicate and coordinate their movements to form shapes, explore the environment, or collectively transport objects.
- RoboBees: RoboBees are miniature robotic bees developed by researchers at Harvard University. They aim to mimic the collective behavior of real honeybees and perform tasks such as environmental monitoring or crop pollination.

3. 3. Swarm Intelligence in Decision Making: In this category, swarm intelligence is used to support decision-making processes. It involves aggregating the opinions, preferences, or judgments of a group of individuals to make a collective decision. Systems like the "wisdom of crowds" approach or the Bounded Confidence Model leverage swarm intelligence to enhance decision-making accuracy and improve problem-solving capabilities. Examples of SI algorithms belongs to Swarm Intelligence in Decision Making includes:
   - Prediction Markets: Prediction markets allow participants to trade on the outcome of future events. By aggregating the information and opinions of participants, these markets can provide accurate predictions for various domains like politics, finance, or sports.
   - Opinion Dynamics Models: Models such as the Bounded Confidence Model simulate the interactions and opinion exchanges among individuals. These models help understand how consensus or polarization emerges in a population and can guide decision-making processes.
4. Swarm Behavior in Nature: This category examines the behavior of natural swarms, such as bird flocks, fish schools, or insect colonies, to understand the principles underlying their collective actions. Researchers study how individual agents interact, communicate, and self-organize to exhibit emergent behaviors that benefit the entire swarm. Understanding natural swarm behavior can inspire the development of artificial swarm systems and provide insights into self-organization and cooperation. Examples of SI algorithms belongs to Swarm Behavior in Nature includes:
   - Bird Flocking: The collective behavior of bird flocks, where individual birds coordinate their movements, maintain formation, and respond to environmental changes, is studied to understand principles of self-organization and coordination in swarms.
   - Ant Colony Behavior: Ant colonies demonstrate efficient foraging, nest-building, and trail-following behaviors. Researchers study these behaviors to gain insights into robust and scalable systems for tasks like resource allocation, routing, or task allocation in distributed systems.
5. Swarm Computing: Swarm computing focuses on solving complex computational problems by utilizing the collective power of a large number of relatively simple computing units, often inspired by biological swarms. Each unit (or agent) may have limited computational capabilities, but by working collaboratively, they can achieve complex tasks such as distributed data processing, load balancing, or distributed problem-solving. Examples of SI algorithms belongs to Swarm Computing includes:
   - Distributed Data Processing: Systems like Apache Hadoop or Apache Spark utilize distributed computing frameworks to process large datasets across a cluster of machines, where each machine acts as a computing unit working collectively to analyze and process the data.
   - Load Balancing: Load balancing algorithms distribute tasks or workload across multiple computing units to optimize resource utilization and ensure efficient processing in distributed systems.

These categories are not mutually exclusive, and there can be overlap and interdisciplinary approaches. Swarm intelligence continues to be a fascinating field of study, offering insights into decentralized and self-organizing systems that can inspire advancements in various domains. some examples of applications or systems in each of the main categories of swarm intelligence includes:

III. NP-HARD PROBLEM

Non-deterministic Polynomial-time hard (NP-hard) problems are those problems that are impossible to obtain their optimal solution within a polynomial bounded computation time. Generally the methods for solving real-life optimization problems are belonging to one the following categories: complete approach or approximate approach. Complete approach requires exponential time (long execution time) for solving NP-hard problems; on the other hand the approximate approaches are divided in to two search approaches, population-based search and single-based search. Both population-based search and single-based search are focusing on obtaining a relatively good solution in a relatively less time (short execution time) instead of finding an optimal solution which is not easy to compute and have a long execution time. Using exact algorithms for NP-hard problems are not preferable, because they takes unbounded (long) time to execute, for this reason researches often use approximates methods, which tries to obtain a near optimal solution for NP-hard problems in a significantly short bounded time[13].

IV. TRAVELLING SALESMAN PROBLEM (TSP):

Traveling Salesman Problem (TSP) was first proposed by the scientist K. Menger in 1932. Since then, it has been a case of study to many researchers. It is one of the well-known intensively studied problems in optimization and evaluations problems and is used as a benchmark for many optimization methods[21], [22].

TSP is an NP-hard problem in combinatorial optimization. The travelling salesman trays to visit all stations (cities) in a map and return to the start station (city) without visiting any station (city) twice, the best tour is the shortest tour among all possible tours. The length of the optimal tour of TSP problems can be found as shown below.

$$optimal\ tour = d_{p(n)\,p(1)} + \left(\sum_{i=1}^{n-1} d_{p(i)\,p(i+1)}\right)$$

Where p is a probability list of cities with minimum distance between city (p_i and p_{i+1}).

A. Algorithms for Solving a TSP

Even though, the TSP has received a great attention, the exploration and research on the MTSP is relatively confined and most of the work is linked to MTSP applications. Other Nature-inspired optimization algorithms such as ant colony optimization (ACO), particle swarm optimization (PSO) and artificial neural network (ANN) is used to solve the TSP/MTSP. These algorithms help to avoid local minima, when used in conjunction with various heuristic techniques and also reduces the computational cost. For example, R. Jayasutha and Dr. B. S. E. Zoraida (2013) proposed the novel Genetic algorithm[5], [23], [24]. The traveling salesman problem (TSP) explained the concept in which single salesperson travels the route visiting all n cities only once and returns to the starting location from where it started. In this paper, mTSP has also been explained by using GA in the sort of the vehicle scheduling problem. The work in this paper presents a chromosome methodology determining the MTSP to be solved using a GA [25]. In addition to this, the tools developed for a modified mTSP concerned with the optimization of one-to-many distribution systems will also be studied. Table 1, summarize some SI algorithm in solving TSP.

Table 1: summary of solving TSP by SI algorithms

#	Author	Details
1	Varunika et al. (2014)	presents the technique to solve the multiple travelling salesperson problem using a modified genetic algorithm. Travelling Salesman Problem (TSP) is an optimization problem. According to this, salesperson must make a path through a certain number of cities and visiting each only once and must minimize the total distance travelled by it. The Multiple Traveling Salesman Problem (mTSP) is the generalization of TSP and is a complex combinatorial optimization problem, in which one or more than one salesmen can be used in the solution. For this problem, the Constraint in the optimization task is that each salesman returns to starting point at end of trip, travelling to a specific set of cities in between and except for the first, every city is visited by exactly one salesman. The Cost Function is to search for the shortest path with the minimal distance and each salesman must travel from the starting location to individual cities and back to the location from where he begin the travelling. mTSP is a complex NP-Hard problem and has a multiplicity of applications. The amount of computation time to solve this problem grows exponentially as number of cities is increased so, the heuristic optimization algorithms, such as genetic algorithms (GAs) need to take into

		account. GA generates a population of solutions at each iteration& the best point in the population approaches an optimal solution. The aim of this paper is to review that how genetic algorithm can be applied to solve these problems and propose an efficient and optimal solution to mTSP.
2	A. Kiraly and J. Abonyi (2011)	presented the way to apply genetic algorithms to solve the various problems and proposed a novel interpretable representation based algorithm. The Vehicle Routing Problem (VRP) is also a complex combinatorial optimization problem consisted of group of vehicles with unvarying capacity, a common station and several costumer demands and the goal is to find the set of routes with minimum cost. The multiple travelling salesman problem (mTSP) is an abstraction of the widely used traveling salesman problem (TSP), in which solution set consists of more than one salesman. The MTSP-based algorithms can also be applied in various VRP's by incorporating some constraints.
3	M.Sedighpour et al. (2011)	has evaluated the performance of different meta-heuristic algorithms. The multiple traveling salesman problem (mTSP) includes assigning m salesmen the n nodes such that each node is visited exactly once. The MTSP has a multiple of application in many fields and is also an example of combinatorial optimization problems. In this paper, for solving the MTSP a modified hybrid meta-heuristic algorithm named GA2OPT is proposed. Firstly, the MTSP is solved by the modified genetic Algorithm (GA) by using number of iterations and secondly, the 2-Opt local search algorithm is used for enhancing solutions for that iteration. The proposed algorithm was trialed on a 6 benchmark instances from the TSPLIB and in all of the four instances the best known solution was improved and for the rest of benchmarks, the quality of the produced solution varies less than 0.01% from the best known solutions ever
4	H .Larki and M. Yousefikhoshbakht (2014)	presented an effective and evolutionary optimization algorithm which has been formulated through combination of Modified Imperialist Competitive Algorithm and Lin-Kernigan Algorithm (MICA) in order to solve the MTSP. An absorption function and several local search algorithms as a revolution operator are used for work in the proposed algorithm. The performance of this algorithm was quite well and competitive when tested on several MTSP benchmark instances and the results confirmed that the MICA performs well with other meta-heuristic algorithm. In this paper, a hybrid algorithm called MICA was proposed for solving the MTSP. This algorithm is more efficient than SA+EAS, ICA and the modified genetic algorithms for dealing with MTSP. The algorithm was tested in 26 benchmark problems with 20– 150 nodes and it was found capable of improving the BKS of 6 instances.
5	T.Mohammadpour and M. Yadollahi (2014)	proposed an algorithm that would lead to better solutions when compared with other algorithms. Multiple Travelling Salesman Problem (mTSP) is a NP hard problem and is the general form of the well-known Traveling Salesman Problem (TSP). The MTSP will be more appropriate for posing realworld situations because it is capable for handling more than one salesman and many of the

		situations are related to different scheduling and routing area. In the MTSP, the two works must be done simultaneously: Firstly, each salesman is allocated a separate city and in other case, an order is specified by which salesman will visit each city. That’s why MTSP is more complicated than TSP. The one way to solve MTSP is to translate it in to standard TSP. Till now; lots of algorithms have been demonstrated for solving this problem. In this paper, a new Hybrid method has been introduced for solving the MTSP, by combining gravity and Genetic algorithmic program. Experimentally the results showed that the suggested algorithm would lead to better solutions when compared with other algorithms.
6	Xu et al. (2011)	evaluated an extended Christofides heuristic for the k-depot TSP (multiple depot multiple travelling salesman problem) and specified proof that showed that it achieves a closed approximation ratio of (2 - 1/k), which is close to 3/2 when k is close to 2which is better than the algorithm available in the current literature. The proof of this is established on the derivation of bounds for the minimal perfect matching used in the extended heuristic.
7	Shuai et al. (2013)	proposed an operator called twopart chromosome crossover (TCX) operator for solving the multiple travelling salesmen problem (MTSP) using a genetic algorithm (GA) for near-optimum solutions. They adopt the proven two-part chromosome representation technique which minimizes the size of the problem location. The existing crossover method for the two-part chromosome representation has two limitations. First of all, in the second part of the chromosome it has extremely limited diversity that greatly confines the search ability of the GA and the other limitation is that, in the first part of the chromosome the pre-existing crossover approach tends to break useful building blocks that reduces the effectiveness of GA and its solution quality. Hence, they proposed the TCX to overcome the above two limitations and to improve solution. For minimizing total travel distance and minimizing longest tour they evaluated and compared the TCX with three distinct crossover methods. The result show that compared to three existing crossover approaches TCX can better improve the solution quality of the GA.
8	Arthur E. Carter (2003)	focused on solving vehicle scheduling problems by using GAs. It consist of scheduling fleet of m vehicles to visit n number of cities with each city being visited by one vehicle. The VSP typically includes constraints on the number of cities each vehicle can visit due to the capacity of each truck available and the size load to be picked up at each city. The cities must be visited between specific times called time windows in some cases. These issues lead to a number of different possible configurations for the VSP, VSP with heterogeneous/homogeneous vehicle capacities, VSP for minimum distance/minimum vehicle requirements and VSP with/without time windows. It also consist of number of objectives, such as: minimize the number of vehicles required, minimize the total distance and minimize lateness (if time windows are used).

9	R.Hassan et al. (2004)	analyzed that PSO has similarity to the Genetic Algorithm (GA) because both are populationbased search methods. Also, PSO and the GA use a combination of deterministic and probabilistic rules by moving from a set of points (population) to another set of points in a single iteration with liable improvement. In addition, the GA and its many versions is popular in academia and the industry due to its ease of implementation ,intuitiveness and the ability to effectively solve mixed integer optimization problems that are complex for engineering systems. The expensive computational cost is the drawback of the GA. The work in this paper is an attempt to examine the correctness of the claim that PSO has the same effectiveness as the GA but with significantly better computational efficiency by implementing formal hypothesis testing and statistical analysis. The comparison of the GA and PSO is enforced using two space systems design optimization problems known as telescope array configuration and spacecraft reliability-based design as well as a set of benchmark test problems.
10	X.H. et al. (2007)	has focused a novel discrete PSO algorithm and has presented it by adding an uncertain strategy into the approach. Moreover, the algorithm is extended for solving the generalized TSP problems by introducing the generalized chromosome technique. To the best of our knowledge, it is the first time that the PSO-based algorithm has been used to solve the GTSP problems. Some benchmark problems are used to examine the effectiveness of the proposed algorithms. Numeric results show that the proposed algorithms are efficacious. It has also been shown that the proposed algorithms can solve larger size problems than those solved using the existing algorithms.
11	Q. Bai (2010)	has analyzed that Particle swarm optimization is a heuristic global optimization method, which is based on swarm intelligence. The idea came from the research on the bird and fish flock movement activity. This algorithm is widely used and implemented because it is easy implement and also few particles need to be adjusted. In this paper, principle of PSO is presented and also the advantages and the shortcomings are summed up. Finally this paper presents some kinds of improved versions of PSO and research condition, and the future research issues are also given.
12	Liu et al. (2009)	proposed an algorithm for solving the mTSP known as ACO. Their algorithm includes, the pheromone trail updating and limits followed the MAX–MIN Ant System strategy, and to improve the performance of the algorithm a local search procedure was used. In this paper, authors compared the existing GA approaches with the results of the algorithm on standard benchmark instances in the literature. Computational results show that their algorithm was competitive over two typical objective functions.
13	M. Yousefikhoshbakht et al. (2013)	presented a new modified edition of the ant colony optimization(ACO) mixed with insert, swap and 2-opt algorithm called NMACO for solving the multiple traveling salesman problem (MTSP) which utilizes an effective and efficient standard for escaping from the local optimum. The objective of MTSP is to

		minimize the total distance traveled by several salesmen for servicing a set of nodes. As this problem belongs to NP-hard Problems, indeed some meta-heuristic acts have been used to solve it in the recent centuries. In contrast to the classical Ant colony optimization, the proposed algorithm uses only a global updation for the current best solution and the best found solution until now. Moreover, a new state transition rule and an efficient candidate list are used for assessing the efficiency of the proposed algorithm. This proposed algorithm was tested on some standard benchmark instances available from the literature and when their results were compared with other well-known metaheuristic algorithms then their results indicates that the NMACO has been able to improve the efficiency of the ACO in all instances.
14	M. Dorigo and T. Stutzle (2010	studied that Ant Colony Optimization (ACO) is a meta-heuristic Algorithm that is encouraged by the pheromone trail laying and adopting behavior of some ant species. Artificial ants in ACO are random solution construction processes that build candidate solutions for the problem instance under concern by employing (artificial) pheromone information that is adapted based on the ants' search experience and possibly available heuristic information. One thing to notice in ants is their caliber to create "ant streets". This paper focused on the development of high-performing algorithmic variants, the development of a generic algorithmic framework for ACO algorithms, successful applications of ACO algorithms to a wide range of computationally hard problems and the theoretical understanding of properties of ACO algorithms. This paper reviews these developments and gives an overview of recent research trends in ACO.

The most well-known SI algorithms in solving TSP includes:

- **Particle Swarm Optimization (PSO):**
  The Particle Swarm Optimization (PSO) algorithm concept roots from the social behavior of organisms such as fishing schooling bird flocking, it was first introduced by Kennedy and Eberhart in 1995 and is widely used to solve computational problems. PSO particles cooperate between themselves as one group to achieve their goal efficiently and effectively. PSO simulates this social behavior as an optimization tool to solve some optimization problems and NP-Hard problems such as Travelling Salesman Problem (TSP). PSO pseudo code is shown below in (algorithm 1):

Algorithm 1: Pseudocode of the basic PSO

```
Initialize:
  - Set the number of particles, n
  - Initialize the position and velocity of each particle randomly
  - Set the personal best position and the personal best distance for each particle
  - Set the global best position and the global best distance
  - Set the parameters: w (inertia weight), c1 (cognitive/individual component weight), c2
(social component weight), vmax (maximum velocity)
  - Repeat until termination condition met:
    - For each particle i = 1 to n:
      - Evaluate the distance traveled by the particle's current tour
      - Update the personal best position and distance if the current tour is better than the
personal best
      - Update the global best position and distance if the personal best of particle i is better
than the global best
      - Update the velocity of the particle using the formula:
        v(i) = w * v(i) + c1 * rand() * (pbest(i) - x(i)) + c2 * rand() * (gbest - x(i))
      - Clip the velocity to ensure it does not exceed the maximum velocity, vmax
      - Update the position of the particle using the formula:
        x(i) = x(i) + v(i)
    - Return the particle with the best tour (shortest distance)
```

In this pseudocode, particles represent potential solutions to the TSP. Each particle adjusts its velocity and position based on its own experience (personal best) and the experience of the entire swarm (global best). The velocity update equation consists of three components: inertia, cognitive (based on personal best), and social (based on global best). The velocity is then used to update the position of the particle. The process is repeated until a termination condition is met, such as a maximum number of iterations or convergence criteria.

- **Bat algorithm (BA):**

Bat algorithm (BA) is a population based metaheuristic algorithm proposed by Yang in 2010 for solving continuous optimization problems [6], [7]. The basic BA algorithm is bio-inspired metaheuristic on the bio-sonar or echolocation characteristics of bats. In nature, bats release ultrasonic waves to the environment around it for the purposes of hunting or navigation. After the emission of these waves, it receives the echoes of the waves, and based on the received echo they locate themselves and identify obstacles in their ways and preys. Furthermore, each agent in swarm is capable of finding the most “nutritious" areas or moving towards a previous best location found by the swarm. Bat algorithm has showed grate efficiency in finding solution in continuous optimization problems. This article have adapted BA for to find solutions for travelling salesman problem (TSP), which is one of NP-hard problem in combinatorial optimization. BA pseudo code is shown below in (Algorithm 2):

Algorithm 2: Pseudocode of the basic BA

```
Initialize:
  - Set the number of bats, n
  - Set the maximum number of iterations, maxIterations
  - Initialize the positions of each bat randomly as a permutation of cities
  - Initialize the velocities of each bat randomly
  - Set the loudness of each bat as 1.0
  - Set the pulse emission rate, r ∈ [0, 1]
  - Set the loudness decay rate, α ∈ [0, 1]
  - Set the minimum frequency, fmin
  - Set the maximum frequency, fmax
  - Set the termination criteria: maximum iterations reached or convergence
  - Repeat until termination criteria met:
    - For each bat i = 1 to n:
      - Evaluate the distance traveled by the bat's current tour
      - If the bat's loudness is the highest among all bats, update the global best tour
      - Generate a new solution by updating the bat's position and velocity:
        newVelocity = oldVelocity + (globalBestPosition - currentPosition) * frequency
        newPosition = currentPosition + newVelocity
        Apply a local search operator (e.g., 2-opt) to the new position
      - With a probability r, update the bat's position randomly
    - Update the loudness of each bat:
      loudness = α * loudness
    - Update the frequency of each bat:
      frequency = fmin + (fmax - fmin) * random()
  - Return the bat with the best tour (distance)
```

In this pseudocode, bats represent potential solutions to the TSP. The algorithm iteratively updates the positions and velocities of the bats based on their fitness (shortest distance). Bats generate new solutions by adjusting their positions and velocities, with the global best tour influencing their movement. A local search operator, such as 2-opt, can be applied to improve the new positions. The loudness of each bat decreases over time, and with a certain probability, bats update their positions randomly. The frequency of the bats is also adjusted during the iterations.

- **Artificial Bee Colony (ABC):**

Artificial Bee Colony (ABC) is one of the most recent swarm intelligence algorithms. It was proposed by Dervis Karaboga in 2005 for solving multi-dimensional and multi-modal optimization problems[8]. ABC algorithm is inspired from the intelligent, interactive and foraging behavior of real honey bees in searching for food sources “nectar”, and announcing other bees in the nest about the information of food source. In the bees nest each agent store the information of the previously visited food source area in her memory and searches for a new better food source, if the nectar (fitness) of the new solution (food source) is greater than the previous one, then the agents bee forgets the old source and store the information of the new source.
In order to apply ABC to TSP problem, a food source corresponds to a feasible tour. Also, the nectar amounts of a food source represent the quality of the solution tour. Firstly, a random set of food sources are generated by agents’ bees. Then, the nectar (fitness) of the visited food sources are calculated as an objective function. As a result, the quality of the initial solutions is evaluated. ABC pseudo code is shown below in (algorithm 3):

Algorithm 3: Pseudocode of the basic ABC

```
Initialize:
  - Set the number of employed bees, n
  - Set the number of onlooker bees, m
  - Set the maximum number of iterations, maxIterations
  - Initialize the positions of each bee randomly as a permutation of cities
  - Set the parameter: limit (maximum number of trials without improvement)
  - Set the termination criteria: maximum iterations reached or convergence
  - Repeat until termination criteria met:
    - Employed bees phase:
      - For each employed bee i = 1 to n:
        - Select a random neighbor j ≠ i
        - Generate a new solution by applying a local search operator (e.g., 2-opt) to the position
of bee i and neighbor j
        - Evaluate the distance traveled by the new solution
        - If the new solution is better than the current position, replace the current position with
the new solution
        - If the new solution is not better, increase the trial counter for bee i
    - Calculate the fitness of each employed bee
    - Onlooker bees phase:
      - For each onlooker bee k = 1 to m:
        - Select an employed bee based on roulette wheel selection with probabilities
proportional to their fitness values
        - Apply the same procedure as in the employed bees phase
    - Calculate the fitness of each onlooker bee
    - Determine the best solution among the employed bees and onlooker bees
    - Scout bees phase:
      - If a bee's trial counter exceeds the limit, generate a new random solution for that bee
    - Return the best solution (shortest distance)
```

- **Ant Colony Optimization (ACO):**

The Ant Colony Optimization (ACO) is a heuristic algorithm inspired from the behavior of real ant in finding the shortest way to a source of food [1, 2, 6].
Naturally, ants in a swarm are indirectly communicates by an odorous chemical substance that ants may deposit and smell called pheromone trails. In a swarm, each ant which represent an agent in a collection randomly, laying down a pheromone trail in its way to a food source, if any agent founds a food, it return to the nest by smelling pheromone trail, in case of increase of pheromone in any path all the other agents follow that path [1,3].
In this article, ACO is adapted to find solution for TSP problem, in a graph first each agent of colony use the pheromone trail to choose a nearest station to its current station, the process continues by adding stations one by one until it complete TSP tour by visiting all stations and back to the starting station. ACO update the pheromone trail after each agent complete its tour. ACO pseudo code is shown below in (algorithm 4):

Algorithm 4: Pseudocode of the basic ACO

```
Initialize:
  - Set the number of ants, n
  - Set the pheromone matrix τ with initial values
  - Set the heuristic information matrix η based on distances between cities
  - Set the parameters: α (pheromone influence), β (heuristic information influence), ρ
(pheromone evaporation rate), and Q (pheromone deposit constant)
  - Repeat until termination condition met:
    - For each ant k = 1 to n:
      - Initialize an empty tabu list for ant k
      - Choose a random starting city for ant k
      - Repeat until all cities are visited:
        - For each unvisited city i:
          - Calculate the selection probability for moving to city i using the formula:
              $P(k, i) = (\tau(k, i)^\alpha) * (\eta(k, i)^\beta) / \Sigma[(\tau(k, j)^\alpha) * (\eta(k, j)^\beta)]$ (for all unvisited cities j)
          - Select the next city to visit based on the selection probabilities
          - Add the selected city to the tabu list of ant k
      - Calculate the total distance traveled by ant k
    - Update the pheromone trails:
      - Evaporate the pheromone on all edges: $\tau(i, j) = (1 - \rho) * \tau(i, j)$ (for all edges (i, j))
      - Deposit pheromone on edges visited by ants: $\tau(i, j) = \tau(i, j) + \Delta\tau(i, j)$ (for all edges (i, j)
visited by ants, where $\Delta\tau(i, j) = Q / totalDistance$)
  - Return the ant with the best tour (shortest distance)
```

In this pseudocode, ants construct solutions by probabilistically selecting the next city to visit based on the pheromone levels and heuristic information. Pheromone trails are updated by evaporating the existing pheromone and depositing pheromone based on the quality of the solutions found by ants. The process is repeated for multiple iterations until a termination condition is met, such as a maximum number of iterations or convergence criteria.

- **Elephant Herding Algorithm (EHA)**

The Elephant Herding Algorithm (EHA) is a nature-inspired optimization algorithm that simulates the herding behavior of elephants. While it is primarily designed for continuous optimization problems [26], it can potentially be adapted for discrete problems like the Traveling Salesperson Problem (TSP). However, it's important to note that the EHA is not as commonly used or well-established for the TSP compared to algorithms like Ant Colony Optimization (ACO) or Particle Swarm Optimization (PSO).
To use the Elephant Herding Algorithm for the TSP, it is required to define how the algorithm's components, such as herds, leaders, and elephant movements, can be mapped to the TSP problem representation. You would also need to consider how the algorithm's search and optimization mechanisms would be adapted to find high-quality TSP solutions[22], [27].
Adapting the Elephant Herding Algorithm for the TSP would likely involve defining suitable representations for tours, designing appropriate update rules for the elephant movements based on the problem constraints, and determining how herds and leaders can be utilized to guide the search process. EHA pseudo code is shown below in (algorithm 5):

Algorithm 5: Pseudocode of the basic EHA

```
Initialize:
  - Set the number of elephants, n
  - Set the maximum number of iterations, maxIterations
  - Initialize the positions of each elephant randomly as a permutation of cities
  - Set the parameter: α (scaling factor)
  - Set the termination criteria: maximum iterations reached or convergence
  - Repeat until termination criteria met:
    - For each elephant i = 1 to n:
      - Evaluate the distance traveled by the elephant's current tour
    - Sort the elephants based on their fitness (shortest distance first)
    - For each elephant i = 1 to n:
      - Calculate the total distance of the elephants ahead of elephant i
      - Calculate the probability of each elephant i as:
        Pi = exp(-α * Di / Dtotal) (where Di is the distance traveled by elephant i and Dtotal
is the total distance)
      - Update the position of the elephant i by selecting a city from the previous tour with
probability Pi
    - Apply a local search operator (e.g., 2-opt) to improve the solutions of the elephants
  - Return the best elephant with the shortest tour (distance)
```

In this pseudocode, elephants represent potential solutions to the TSP. The algorithm iteratively updates the positions of the elephants based on their fitness (shortest distance) and the probability calculations. The elephants are sorted based on their fitness, and the probability of each elephant is calculated using a scaling factor α. The position of each elephant is updated by selecting a city from its previous tour with a probability determined by its fitness. Additionally, a local search operator, such as 2-opt, can be applied to improve the solutions of the elephants.

- **Cuckoo Search Optimization (CSO)**

Cuckoo Search Optimization (CSO) is a nature-inspired metaheuristic algorithm inspired by the brood parasitic behavior of some cuckoo species. It was first proposed by Xin-She Yang and Suash Deb in 2009[28]. The CSO algorithm is designed to solve optimization problems by mimicking the process of a cuckoo laying eggs in the nests of other bird species and replacing their eggs. In the context of optimization, nests represent potential solutions, and eggs represent new candidate solutions.

The basic concept of CSO involves three main steps: egg laying, egg discovery, and local random walk. During the egg laying process, cuckoos generate new solutions based on their current nests and apply a local search operator to improve these solutions. The new solutions are then sorted based on their fitness. In the egg discovery process, nests with poor fitness are randomly discovered and replaced with new solutions generated by other cuckoos. This process promotes exploration of the search space. Finally, the local random walk step involves updating the position of each cuckoo using a Lévy flight, which is a random step inspired by Lévy distribution. This step enhances the global search capability of the algorithm[29].

When applying CSO to solve the Traveling Salesperson Problem (TSP), the nests represent potential TSP tours, and the goal is to find the shortest tour that visits all cities. The positions of the nests correspond to the order in which the cities are visited. By iteratively updating the positions of the nests and applying local search operators, CSO explores the search space to find high-quality TSP solutions. The Lévy flight component enables long-range exploration, while the egg discovery process facilitates the exchange of information among nests, improving the search efficiency.

To use CSO for solving the TSP, specific adaptations are required. These include representing nests as permutations of cities, defining suitable fitness measures based on the tour lengths, and incorporating local search operators like 2-opt or other heuristics to improve the solutions. Additionally, appropriate parameter tuning and handling of constraints, such as visiting each city exactly once, need to be considered for effective application of CSO to the TSP. CSO pseudo code is shown below in (algorithm 6):

Algorithm 6: Pseudocode of the basic CSO

```
Initialize:
  - Set the number of nests, n
  - Set the maximum number of iterations, maxIterations
  - Initialize the positions of each nest randomly as a permutation of cities
  - Set the probability of cuckoo egg discovery, pa ∈ [0, 1]
  - Set the step size scaling factor, α
  - Set the termination criteria: maximum iterations reached or convergence
  - Repeat until termination criteria met:
    - Generate a new solution by applying a local search operator (e.g., 2-opt) to each nest's
position
    - Evaluate the distance traveled by each new solution
    - Sort the nests based on their fitness (shortest distance first)
    - Generate new solutions by performing Lévy flights:
      - For each nest i = 1 to n:
        - Generate a random Lévy step
        - Generate a random nest index j ≠ i
        - Update the position of nest i using the Lévy step:
          newPosition = currentPosition + α * LévyStep * (currentPosition - nestPosition(j))
        - Apply a local search operator (e.g., 2-opt) to the new position
        - Evaluate the distance traveled by the new solution
        - If the new solution is better than the current position and with a probability pa, replace
the current position with the new solution
    - Return the nest with the best tour (distance)
```

In this pseudocode, nests represent potential solutions to the TSP. The algorithm iteratively updates the positions of the nests based on their fitness (shortest distance) and Lévy flights. Lévy flights are random search steps inspired by Lévy distribution. Each nest generates a new solution by updating its position using a Lévy step, with the aim of exploring the search space. Local search operators, such as 2-opt, can be applied to improve the new positions. The probability of replacing the current position with a new solution depends on the fitness and the probability of cuckoo egg discovery. The process continues until a termination condition is met, and the nest with the best tour (shortest distance) is returned.

- **Spider Optimization Algorithm (SOA)**

The Spider Optimization Algorithm (SOA) is a nature-inspired metaheuristic algorithm that draws inspiration from the hunting behavior of spiders. It was proposed by F. Bastani et al. in 2011. SOA aims to find optimal or near-optimal solutions for optimization problems by simulating the hunting behavior of spiders and their ability to capture prey[30].

In the context of the Traveling Salesperson Problem (TSP), SOA can be adapted to find high-quality solutions that minimize the total distance traveled by the salesperson. The algorithm starts with a population of spiders, where each spider represents a potential solution (TSP tour). The spiders move in the search space, updating their positions based on local search operations and random movements[30].

The core idea of SOA is to balance the exploration and exploitation of the search space. The algorithm achieves this through two main components: local search and random walk.

- Local Search:
  - The best spiders, determined by their fitness (shortest distance), undergo local search to refine their positions.
  - A local search operator, such as the 2-opt heuristic, is applied to improve the tour of the best spiders.
  - By exploiting the local neighborhood, the best spiders aim to fine-tune their solutions and converge towards better solutions.
- Random Walk:
  - The remaining spiders, after the best spiders have undergone local search, move randomly using a random walk process.

- Each spider selects a random direction for its movement.
- The spiders update their positions by performing a random walk, exploring the search space and potentially discovering new promising solutions.

The spiders' movements, both in the local search and random walk steps, are guided by the fitness (distance) of their solutions. The algorithm iteratively updates the positions of the spiders, evaluating their fitness and comparing them to determine the best spider with the shortest tour.

By combining the exploitation capability of local search with the exploration capability of random movements, SOA seeks to efficiently search the solution space and find good solutions for the TSP. However, as with any metaheuristic algorithm, parameter tuning and problem-specific adaptations may be necessary to enhance its performance and convergence speed. SOA pseudo code is shown below in (algorithm 7):

Algorithm 7: Pseudocode of the basic SOA

```
Initialize:
  - Set the population size, n
  - Set the maximum number of iterations, maxIterations
  - Initialize the positions of spiders randomly as a permutation of cities
  - Set the number of best spiders, b
  - Set the parameter for random walk, α
  - Set the termination criteria: maximum iterations reached or convergence
  - Repeat until termination criteria met:
    - Evaluate the fitness (total distance) of each spider
    - Sort the spiders based on their fitness (shortest distance first)
    - Update the positions of the best spiders by performing a local search:
      - For each best spider i = 1 to b:
        - Generate a new solution by applying a local search operator (e.g., 2-opt) to the
position of spider i
        - Evaluate the fitness of the new solution
        - If the new solution is better than the current position, replace the current position
with the new solution
    - Update the positions of the remaining spiders:
      - For each spider j = b+1 to n:
        - Generate a random direction for the spider's movement
        - Update the position of spider j using a random walk:
          newPosition = currentPosition + α * randomDirection
        - Apply a local search operator (e.g., 2-opt) to the new position
    - Return the best spider with the shortest tour (distance)
```

V. CONCLUSIONS:

Swarm Intelligence (SI) algorithms have demonstrated great potential for solving the Symmetric Traveling Salesman Problem (TSP). SI algorithms, inspired by collective behavior observed in social insect colonies and other natural systems, offer efficient and effective approaches to tackle the combinatorial optimization challenges posed by the TSP. The collective behavior and cooperation exhibited by swarms in nature serve as inspiration for SI algorithms, which leverage the power of decentralized decision-making and information sharing among individuals within a population. By simulating the interactions and cooperation observed in natural systems, SI algorithms effectively explore the search space, balance exploration and exploitation, and converge towards high-quality solutions for the TSP. SI algorithms such as Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), and other variants have been successfully applied to the TSP. These algorithms iteratively construct and update candidate solutions using mechanisms such as pheromone trails, individual particle movements, or the interactions between individuals in a swarm. Through local search operators, global communication, and intelligent decision-making, SI algorithms optimize the TSP by iteratively improving candidate solutions and converging towards near-optimal solutions.

The strengths of SI algorithms lie in their ability to handle large-scale TSP instances, their adaptability to dynamic environments, and their robustness in finding good solutions. They are also capable of handling complex constraints and variations in the TSP problem formulation. Additionally, SI algorithms often provide trade-offs between exploration and exploitation, allowing them to strike a balance between searching the solution space and exploiting promising solutions. However, it's important to note that the performance of SI algorithms for the TSP depends on several factors, including the algorithm's parameters, problem-specific adaptations, and the nature of the TSP instance. Proper parameter tuning, efficient representation of solutions, and the choice of appropriate local search operators can significantly impact the algorithm's performance. Additionally, for large-scale TSP instances, parallel computing and hybridization with other optimization techniques may be employed to enhance the algorithm's efficiency and effectiveness.

In conclusion, Swarm Intelligence algorithms offer a promising and effective approach to solving the Symmetric Traveling Salesman Problem. By harnessing the power of collective behavior and decentralized decision-making, these algorithms are capable of exploring the solution space, optimizing TSP tours, and converging towards near-optimal solutions. With appropriate adaptations and parameter tuning, SI algorithms provide valuable tools for addressing the complexities of the TSP and related combinatorial optimization problems.

VI. References:

[1] S. M. Almufti, "Historical survey on metaheuristics algorithms," *International Journal of Scientific World*, vol. 7, no. 1, p. 1, Nov. 2019, doi: 10.14419/ijsw.v7i1.29497.

[2] S. M. Almufti, A. Ahmad Shaban, R. Ismael Ali, and J. A. Dela Fuente, "Overview of Metaheuristic Algorithms," *Polaris Global Journal of Scholarly Research and Trends*, vol. 2, no. 2, pp. 10–32, Apr. 2023, doi: 10.58429/pgjsrt.v2n2a144.

[3] Asaad, R. R., Abdurahman, S. M., & Hani, A. A. (2017). Partial Image Encryption using RC4 Stream Cipher Approach and Embedded in an Image. Academic Journal of Nawroz University, 6(3), 40–45. https://doi.org/10.25007/ajnu.v6n3a76

[4] S. M. Almufti, R. B. Marqas, P. S. Othman, and A. B. Sallow, "Single-based and population-based metaheuristics for solving np-hard problems," *Iraqi Journal of Science*, vol. 62, no. 5, pp. 1710–1720, May 2021, doi: 10.24996/ijs.2021.62.5.34.

[5] Rajab Asaad, R., & Masoud Abdulhakim, R. (2021). The Concept of Data Mining and Knowledge Extraction Techniques. Qubahan Academic Journal, 1(2), 17–20. https://doi.org/10.48161/qaj.v1n2a43

[6] A. Acan and A. Unveren, "A shared-memory ACO+GA hybrid for combinatorial optimization," in *2007 IEEE Congress on Evolutionary Computation*, IEEE, Sep. 2007, pp. 2078–2085. doi: 10.1109/CEC.2007.4424729.

[7] Asaad, R. R., Ahmad, H. B., & Ali, R. I. (2020). A Review: Big Data Technologies with Hadoop Distributed Filesystem and Implementing M/R. Academic Journal of Nawroz University, 9(1), 25–33. https://doi.org/10.25007/ajnu.v9n1a530

[8] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, and A. Cosar, "A survey on new generation metaheuristic algorithms," *Comput Ind Eng*, vol. 137, p. 106040, Nov. 2019, doi: 10.1016/j.cie.2019.106040.

[9] Asaad, R. R. (2019). Güler and Linaro et al Model in an Investigation of the Neuronal Dynamics using noise Comparative Study. Academic Journal of Nawroz University, 8(3), 10–16. https://doi.org/10.25007/ajnu.v8n3a360

[10] A. W. Marashdih, Z. F. Zaaba, S. M. Almufti, and Z. Fitri Zaaba, "The Problems and Challenges of Infeasible Paths in Static Analysis Bat Algorithm (BA): Literature Review various types and its Applications View project Hybrid Metaheuristic in solving NP-Hard Problem View project The Problems and Challenges of Infeasible Paths in Static Analysis," *International Journal of Engineering & Technology*, pp. 412–417, 2018, doi: 10.14419/ijet.v7i4.19.23175.

[11] Asaad, R. R. (2021). Penetration Testing: Wireless Network Attacks Method on Kali Linux OS. Academic Journal of Nawroz University, 10(1), 7–12. https://doi.org/10.25007/ajnu.v10n1a998

[12] A. Yahya Zebari, S. M. Almufti, and C. Mohammed Abdulrahman, "Bat algorithm (BA): review, applications and modifications," *International Journal of Scientific World*, vol. 8, no. 1, p. 1, Jan. 2020, doi: 10.14419/ijsw.v8i1.30120.

[13] Asaad, R. R., & Abdulnabi, N. L. (2018). Using Local Searches Algorithms with Ant Colony Optimization for the Solution of TSP Problems. Academic Journal of Nawroz University, 7(3), 1–6. https://doi.org/10.25007/ajnu.v7n3a193

[14] F. S. Abu-Mouti and M. E. El-Hawary, "Overview of Artificial Bee Colony (ABC) algorithm and its applications," in *2012 IEEE International Systems Conference SysCon 2012*, IEEE, Mar. 2012, pp. 1–6. doi: 10.1109/SysCon.2012.6189539.

[15] Asaad, R. R., & Ali, R. I. (2019). Back Propagation Neural Network(BPNN) and Sigmoid Activation Function in Multi-Layer Networks. Academic Journal of Nawroz University, 8(4), 216–221. https://doi.org/10.25007/ajnu.v8n4a464

[16] S. Almufti, "Using Swarm Intelligence for solving NPHard Problems," *Academic Journal of Nawroz University*, vol. 6, no. 3, pp. 46–50, 2017, doi: 10.25007/ajnu.v6n3a78.

[17] Rajab Asaad, R. (2021). Review on Deep Learning and Neural Network Implementation for Emotions Recognition . Qubahan Academic Journal, 1(1), 1–4. https://doi.org/10.48161/qaj.v1n1a25

[18] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of ICNN’95 - International Conference on Neural Networks*, IEEE, pp. 1942–1948. doi: 10.1109/ICNN.1995.488968.

[19] Asaad, R. R., Abdulrahman, S. M., & Hani, A. A. (2017). Advanced Encryption Standard Enhancement with Output Feedback Block Mode Operation. Academic Journal of Nawroz University, 6(3), 1–10. https://doi.org/10.25007/ajnu.v6n3a70

[20] Ridwan B. Marqas, Saman M. Almufti, Pawan Shivan Othman, and Chyavan Mohammed Abdulrahman, “Evaluation of EHO, U-TACO and TS Metaheuristics algorithms in Solving TSP,” *JOURNAL OF XI’AN UNIVERSITY OF ARCHITECTURE & TECHNOLOGY*, vol. XII, no. IV, Apr. 2020, doi: 10.37896/jxat12.04/1062.

[21] Abdulfattah, G. M., Ahmad, M. N., & Asaad, R. R. (2018). A reliable binarization method for offline signature system based on unique signer’s profile. INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING INFORMATION AND CONTROL, 14(2), 573-586.

[22] R. Asaad and N. Abdulnabi, “Using Local Searches Algorithms with Ant Colony Optimization for the Solution of TSP Problems,” *Academic Journal of Nawroz University*, vol. 7, no. 3, pp. 1–6, 2018, doi: 10.25007/ajnu.v7n3a193.

[23] Asaad, R. R., Sulaiman, Z. A., & Abdulmajeed, S. S. (2019). Proposed System for Education Augmented Reality Self English Learning. Academic Journal of Nawroz University, 8(3), 27–32. https://doi.org/10.25007/ajnu.v8n3a366

[24] S. M. Almufti, “U-Turning Ant Colony Algorithm powered by Great Deluge Algorithm for the solution of TSP Problem.”

[25] Asaad, R. R. (2020). Implementation of a Virus with Treatment and Protection Methods. ICONTECH INTERNATIONAL JOURNAL, 4(2), 28-34. https://doi.org/10.46291/ICONTECHvol4iss2pp28-34

[26] S. M. Almufti, “U-Turning Ant Colony Algorithm powered by Great Deluge Algorithm for the solution of TSP Problem.”

[27] Boya Marqas, R., M. Almufti, S., & Rajab Asaad, R. (2022). FIREBASE EFFICIENCY IN CSV DATA EXCHANGE THROUGH PHP-BASED WEBSITES. Academic Journal of Nawroz University, 11(3), 410–414. https://doi.org/10.25007/ajnu.v11n3a1480

[28] S. M. Almufti, R. B. Marqas, P. S. Othman, and A. B. Sallow, “Single-based and population-based metaheuristics for solving np-hard problems,” *Iraqi Journal of Science*, vol. 62, no. 5, 2021, doi: 10.24996/ijs.2021.62.5.34.

[29] Ihsan, R. R., Almufti, S. M., Ormani, B. M., Asaad, R. R., & Marqas, R. B. (2021). A survey on Cat Swarm Optimization algorithm. Asian J. Res. Comput. Sci, 10, 22-32.

[30] A. Gogna and A. Tayal, “Metaheuristics: review and application,” *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 25, no. 4, pp. 503–526, Dec. 2013, doi: 10.1080/0952813X.2013.782347.

[31] Rajab Asaad, R., & Luqman Abdulnabi, N. (2022). A Review on Big Data Analytics between Security and Privacy Issue. Academic Journal of Nawroz University, 11(3), 178–184. https://doi.org/10.25007/ajnu.v11n3a1446

[32] S. M. Almufti, R. Boya Marqas, and V. Ashqi Saeed, “Taxonomy of bio-inspired optimization algorithms,” *Journal of Advanced Computer Science & Technology*, vol. 8, no. 2, p. 23, Aug. 2019, doi: 10.14419/jacst.v8i2.29402.

[33] Yahya Hussien , A., & Rajab Asaad, R. (2022). Review on Social Media and Digital Security. Qubahan Academic Journal, 2(2), 1–4. https://doi.org/10.48161/qaj.v2n2a119

[34] S. M. Almufti and A. A. Shaban, “U-Turning Ant Colony Algorithm for Solving Symmetric Traveling Salesman Problem,” *Academic Journal of Nawroz University*, vol. 7, no. 4, p. 45, Dec. 2018, doi: 10.25007/ajnu.v7n4a270.

[35] Asaad, R. R. (2022). Keras Deep Learning for Pupil Detection Method . Academic Journal of Nawroz University, 10(4), 240–250. https://doi.org/10.25007/ajnu.v10n4a1328

[36] S. M. Almufti, A. Yahya Zebari, and H. Khalid Omer, “A comparative study of particle swarm optimization and genetic algorithm,” *Journal of Advanced Computer Science & Technology*, vol. 8, no. 2, p. 40, Oct. 2019, doi: 10.14419/jacst.v8i2.29401.

[37] Asaad, R. R., & Segerey, R. I. (2018). School Management Application Using iOS. Academic Journal of Nawroz University, 7(4), 38–44. https://doi.org/10.25007/ajnu.v7n4a269

[38] S. M. Almufti, “Hybridizing Ant Colony Optimization Algorithm for Optimizing Edge-Detector Techniques,” *Academic Journal of Nawroz University*, vol. 11, no. 2, pp. 135–145, May 2022, doi: 10.25007/ajnu.v11n2a1320.

[39] Asaad, R. R., Saeed, V. A., & Abdulhakim, R. M. (2021). Smart Agent and it’s effect on Artificial Intelligence: A Review Study. ICONTECH INTERNATIONAL JOURNAL, 5(4), 1-9.

[40] M. T. Adham and P. J. Bentley, “An artificial ecosystem algorithm applied to the travelling salesman problem,” in *GECCO 2014 - Companion Publication of the 2014 Genetic and Evolutionary Computation Conference*, Association for Computing Machinery, 2014, pp. 155–156. doi: 10.1145/2598394.2598438.

[41] Asaad, R. R., & Saeed, V. A. (2022). A Cyber Security Threats, Vulnerability, Challenges and Proposed Solution. Applied Computing Journal, 2(4), 227-244. https://doi.org/10.52098/acj.202260

[42] S. M. Almufti, R. Boya Marqas, and R. R. Asaad, “Comparative study between elephant herding optimization (EHO) and U-turning ant colony optimization (U-TACO) in solving symmetric traveling

salesman problem (STSP),” *Journal of Advanced Computer Science & Technology*, vol. 8, no. 2, p. 32, Aug. 2019, doi: 10.14419/jacst.v8i2.29403.

[43] Renas Rajab Asaad. (2022). Support vector machine classification learning algorithm for diabetes prediction. International Research Journal of Science, Technology, Education, and Management, 2(2), 26–34. https://doi.org/10.5281/zenodo.6975670

[44] D. A. Zebari, S. S. Sadiq and D. M. Sulaiman, "Knee Osteoarthritis Detection Using Deep Feature Based on Convolutional Neural Network," 2022 International Conference on Computer Science and Software Engineering (CSASE), Duhok, Iraq, 2022, pp. 259-264, doi: 10.1109/CSASE51777.2022.9759799.

[45] J. Luis and C. Sequera, “7 Tune Up of a Genetic Algorithm to Group Documentary Collections.” [Online]. Available: www.intechopen.com

[46] Ibrahim, D. A., Zebari, D. A., Mohammed, H. J., & Mohammed, M. A. (2022). Effective hybrid deep learning model for COVID-19 patterns identification using CT images. Expert Systems, 39(10), e13010. https://doi.org/10.1111/exsy.13010

[47] D. A. Zebari, D. M. Sulaiman, S. S. Sadiq, N. A. Zebari and M. S. Salih, "Automated Detection of Covid-19 from X-ray Using SVM," 2022 4th International Conference on Advanced Science and Engineering (ICOASE), Zakho, Iraq, 2022, pp. 130-135, doi: 10.1109/ICOASE56293.2022.10075586.

[48] A. Acan, H. Altincay, Y. Tekol, and A. Unveren, “A genetic algorithm with multiple crossover operators for optimal frequency assignment problem,” in *The 2003 Congress on Evolutionary Computation, 2003. CEC ’03.*, IEEE, pp. 256–263. doi: 10.1109/CEC.2003.1299583.

[49] D. A. Zebari, H. Haron, D. M. Sulaiman, Y. Yusoff and M. N. Mohd Othman, "CNN-based Deep Transfer Learning Approach for Detecting Breast Cancer in Mammogram Images," 2022 IEEE 10th Conference on Systems, Process & Control (ICSPC), Malacca, Malaysia, 2022, pp. 256-261, doi: 10.1109/ICSPC55597.2022.10001781.

[50] “Genetic Algorithm based Feature Selection Technique for Writer Verification: A case study with Handwritten Bangla Document ARTICLE TYPE Genetic Algorithm based Feature Selection Technique for Writer Verification: A case study with Handwritten Bangla Document.”

[51] D. A. Zebari, A. R. Abrahim, D. A. Ibrahim, G. M. Othman and F. Y. H. Ahmed, "Analysis of Dense Descriptors in 3D Face Recognition," 2021 IEEE 11th International Conference on System Engineering and Technology (ICSET), Shah Alam, Malaysia, 2021, pp. 171-176, doi: 10.1109/ICSET53708.2021.9612430.

[52] Sadeeq, H. T., Abdulazeez, A. M., Kako, N. A., Zebari, D. A., & Zeebaree, D. Q. (2021). A New Hybrid Method for Global Optimization Based on the Bird Mating Optimizer and the Differential Evolution. *2021 7th International Engineering Conference “Research & Innovation amid Global Pandemic" (IEC)*, 54–60. https://doi.org/10.1109/IEC52205.2021.9476147

[53] S. M. Almufti, R. R. Asaad, and B. W. Salim, “Review on Elephant Herding Optimization Algorithm Performance in Solving Optimization Problems,” *Article in International Journal of Engineering and Technology*, vol. 7, no. 4, pp. 6109–6114, 2018, doi: 10.14419/ijet.v7i4.23127.

[54] D. A. Zebari, D. A. Ibrahim and A. Al-Zebari, "Suspicious Region Segmentation Using Deep Features in Breast Cancer Mammogram Images," 2022 International Conference on Computer Science and Software Engineering (CSASE), Duhok, Iraq, 2022, pp. 253-258, doi: 10.1109/CSASE51777.2022.9759633.

[55] S. M. Almufti, R. R. Asaad, and B. W. Salim, “Review on Elephant Herding Optimization Algorithm Performance in Solving Optimization Problems,” *Article in International Journal of Engineering and Technology*, vol. 7, no. 4, pp. 6109–6114, 2018, doi: 10.14419/ijet.v7i4.23127.

[56] I. Fister, X.-S. Yang, D. Fister, and I. Fister, “Cuckoo Search: A Brief Literature Review,” 2014, pp. 49–62. doi: 10.1007/978-3-319-02141-6_3.

[57] Mustafa Zebari, G. ., Assad Zebari, D. . ., & Al-zebari, A. . (2021). FUNDAMENTALS OF 5G CELLULAR NETWORKS: A REVIEW. Journal of Information Technology and Informatics, 1(1), 1–5. https://doi.org/10.6084

[58] I. Fister, X.-S. Yang, D. Fister, and I. Fister, “Cuckoo Search: A Brief Literature Review,” 2014, pp. 49–62. doi: 10.1007/978-3-319-02141-6_3.

[59] D. A. Zebari, D. M. Sulaiman, S. S. Sadiq, N. A. Zebari and M. S. Salih, "Automated Detection of Covid-19 from X-ray Using SVM," 2022 4th International Conference on Advanced Science and Engineering (ICOASE), Zakho, Iraq, 2022, pp. 130-135, doi: 10.1109/ICOASE56293.2022.10075586.

[60] S. Almufti, “The novel Social Spider Optimization Algorithm: Overview, Modifications, and Applications,” *ICONTECH INTERNATIONAL JOURNAL*, vol. 5, no. 2, pp. 32–51, Jun. 2021, doi: 10.46291/icontechvol5iss2pp32-51.

[61] D. A. Zebari, H. Haron, D. M. Sulaiman, Y. Yusoff and M. N. Mohd Othman, "CNN-based Deep Transfer Learning Approach for Detecting Breast Cancer in Mammogram Images," 2022 IEEE 10th Conference on Systems, Process & Control (ICSPC), Malacca, Malaysia, 2022, pp. 256-261, doi: 10.1109/ICSPC55597.2022.10001781.