Solving the Travelling Salesman Problem Using the PSO Optimization

Mehr Ali Qasimi*
Selçuk University Institute of Science and Technology, Bilişim Teknolojileri Mühendisliği, Konya, Türkiye

ARTICLE INFO
Received May 13, 2024
Accepted June 18, 2024
Published June 18, 2024

Keywords:
Salesman Problem, metaheuristics, Particle Swarm Optimization

ABSTRACT
This article examines a potential solution to the well-known Travelling Salesman Problem (TSP), which is classified as an NP-hard problem. We also provide a theoretical synopsis of several approaches that have been employed to tackle this problem. A prominent example of a combinatorial problem is the traveling salesman problem (TSP). To address the fundamental PSO algorithm's premature convergence issue and stagnation behavior on TSP, a scout characteristic-based PSO algorithm is suggested. We address Particle Swarm Optimization (PSO), a member of the evolutionary methods class, and outline the methodology for applying PSO to the TSP. Among population-based metaheuristic optimization methods, Particle Swarm Optimization (PSO) is one of the most widely used. Scientific domains such as engineering, chemistry, medicine, advanced physics, and humanities have all effectively employed PSO. Numerous theoretical and empirical results on the convergence and parameterization of PSO versions have been produced as a result of the method's extensive investigation since its introduction in 1995. Hundreds of PSO versions have been developed. It is well recognized that population size has a significant impact on the effectiveness of metaheuristics; nevertheless, no comprehensive research has been done on the appropriate selection of PSO swarm size to date. Through the application of this approach, we examine the effects of various control settings. The ideal solution and the quality of the solution are contrasted.


This is an open-access article distributed under the Creative Commons 4.0 Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. ©2023 by author.

* Corresponding author: q1.mehrali@gmail.com

INTRODUCTION
The goal of optimization is to identify the most appropriate solution while taking the problem’s requirements and constraints into account. It is possible for an issue to have multiple defined answers. The goal function compares these answers to determine which is the best one. The function that is selected depends on the type of issue at hand. One of the common objectives of transportation network optimization is, for instance, journey time or cost. One of the most crucial optimization tasks, though, is selecting the right objective function. Such optimization issues, which involve several objective functions, are occasionally referred to as multiobjective problems when they require the simultaneous optimization of numerous objectives (Yousefikhoshbakht, 2021)

The simplest method for solving these issues is to create a new objective function that is a line composition of the primary objective functions, with weights applied to each function to determine its effect. The aim of optimization is to ascertain the design variables in a manner that yields a quantitative
or optimal objective function. Each optimization problem has several independent variables, often referred to as design variables, which are represented by the vector $x$ with size $n$. Traveling Salesman Problem (TSP) is one of the most researched optimization issues. The technique of methodically analyzing or resolving a problem by choosing and allocating real or integer values to a function in order to minimize or maximize the function is known as optimization in mathematics (Gulcu & Ornek, 2019).

Many optimization tasks are complicated, time-consuming, and NP-hard. A salesman and nodes with known distances between them are present in TSP. The salesman must visit each node just once, pass all of the points at minimum cost, and then head back to the starting location (Bakhtiar et al., 2011). The traveling salesman problem (TSP) is a highly regarded problem because of its various applications in evaluating the effectiveness of new algorithms and in converting other problems to this approach. A salesman leaves the warehouse in this ostensibly straightforward problem, and he or she must return after seeing each customer once. The goal is to cut the mileage as much as possible.

There are several ways to look at TSP problems. As an illustration, this problem is studied to gauge the effectiveness of new algorithms as, although having a rigorous complexity theory structure, most algorithms can be applied to it with ease, making it possible to turn other difficulties into problems that can be resolved. This method, which uses extremely efficient algorithms to solve the TSP issue, can decrease the steps needed to solve particular problems. A higher quality solution is possible for the primary issue. Comparing an algorithm’s efficiency to that of other algorithms can be done research. To solve TSP, numerous algorithms have been developed (Tao & Michalewicz, 1998). Some of them offer the globally optimal solution (based on branch and bound or dynamic programming techniques).

Heuristic algorithms are faster than other types of algorithms, but they do not ensure the best answers. The Lin-Kerninghan algorithm (variable change), algorithms based on greedy principles (nearest neighbor, spanning tree, etc.), and algorithms based on 2-opt or 3-opt change operators are among the well-known algorithms. Numerous contemporary heuristic techniques, including neural networks, evolutionary algorithms, tabu search, and simulated annealing, were also used to the TSP. The Particle Swarm Optimization (PSO) algorithm is an intelligent technology that was initially introduced by Eberhart and Kennedy in 1995. It was created by drawing inspiration from the behavioral laws of human societies, fish schools, and bird flocks (Clerc & Kennedy, 2002). It is possible that PSO and genetic algorithms are similar in that they are all based on population operation.

However, PSO optimizes the population through information interchange among individuals rather than depending on genetic operators such as selection, crossover, and mutation operators to work individually. PSO starts with a set of random solutions and keeps searching until it finds the best answer. When PSO was first introduced, it raised a lot of questions among academics researching optimization, and in a few of years, it was the subject of extensive research. These domains have produced a lot of scientific advances (Clerc & Kennedy, 2002). Various studies and experiments have demonstrated that PSO is a rather high-efficient optimization algorithm. Birds in a group flock synchronously, but in an unpredictable manner, e.g., scattering suddenly, then regrouping again.

Studies to simulate the underlying rules of bird social behavior led to a new approach for optimizing real-valued functions, “Particle Swarm Optimization” (PSO) (Kennedy and Eberhart, 1997, 1995). This new approach combines notions of evolutionary computation and swarm theory (Kennedy, 1997). PSO provides a population-based search procedure, where each individual is abstracted as a “particle” that flies around in a multidimensional search space. The best positions encountered by a particle and its neighbor determine the particle’s trajectory along with other PSO parameters. In other words, a PSO system attempts to balance exploration and exploitation by combining local and global search methods.

In this aspect, PSO is similar to modern GAs and memetic algorithms. In addition to providing a tool for optimization, another important aspect of the PSO approach is its applicability in analyzing socio-cognition of human and artificial agents, based on principles of social psychology. In proposing the PSO approach, Kennedy (1998, 1997) suggests that knowledge is optimized by social interaction, and that thought processes are not completely separate for different individuals, they involve interpersonal communication. Angeline (1998) performed an empirical comparison of PSO with EP, experimenting with different functions, initialization techniques and dimensionality. Eberhart and Shi
(1998) show that the behavior of a PSO system falls in between genetic algorithms and evolutionary programming. Experimental results demonstrated that the PSO algorithms were competitive. Both Eberhart and Angeline advocated focusing on hybrid implementations PSO may search very wide spaces of potential solutions and makes little or no assumptions about the problem being optimized, making it a meta-heuristic. PSO and other metaheuristics do not, however, ensure that an ideal solution will always be discovered (Ozcan & Mohan, 1998). In other words, PSO does not require that the optimization problem be differentiable, unlike traditional optimization techniques like gradient descent and quasi-Newton methods. This is because PSO does not use the gradient of the problem being optimized. As a result, PSO can also be applied to optimization issues that are noisy, partially irregular, evolve over time, etc.

**RESEARCH METHOD**

In this paper, we have a dataset called cities in csv format that contains 40 or 50 cities route information that a salesman want to travel this cities so we uses PSO optimization to solve the TSP problem and find out the best pathe for salesman to travel, In the Traveling salesman problem, the merchant must visit all the cities (starting and ending with city itself) so that the distance covered is as short as possible. The problem of route planning becomes complicated when we increase the number of cities and their location between each other. If every possible path is calculated, it requires large computational resources. In order to find the optimal route, we will use particle swarms. we will use the most direct way to denote TSP-path presentation. For example, path 4-2-1-3-4 can be denoted as (4, 2, 1, 3) or (2, 1, 3, 4) and it is referred as a chromosome. Every chromosome is regarded as a validate path. (In this paper, all paths should be considered as a ring, or closed path).

Fitness function is the only standard of judging whether an individual is “good” or not. We take the reciprocal of the length of each path as the fitness function. Length the shorter, fitness values the better. The fitness function is defined as following formula:

\[
F(S_i) = \frac{1}{\sum_{i=1}^{n} d[c_{n(i)}, c_{n(i+1)} \mod N]}
\]

Travelling Salesman Problem TSP is an optimization problem. It was first formulated in 1930. The goal in the TSP is to find the shortest tour where a salesman starts in a certain city, visits each city only once and returns to the start city (Dorigo & Gambardella, 1997). An example of a TSP and its solution is shown in Fig. 1. Considering the nodes on a graph and the costs between them, TSP can also be defined as traversing all the nodes in the graph at the most cost effective and each node must be visited only once. As the number of nodes increases, it becomes very difficult to determine the most cost-effective tour in the graph. Therefore, TSP is classified as an NP-hard problem (Gülcü et al., 2018)

![Fig. 1. showing the merchant's route](image)

PSO finds the solution faster and performs better in our data. In just 100 iterations, the distance ranges from 210 to around 250. PSO using the same function as the genetic algorithm also finds the solution faster. However, the second function works slower than the first function and produces a less clear path graph. For the PSO algorithm, the best solution would be the first function. The computational time of
the whole process was measured and reported (Table 1). For example for this experiment the best solution iteration and its number were 210 to 250 in 100 iterations.

<table>
<thead>
<tr>
<th>Number of Iterations</th>
<th>Population size</th>
<th>Time[s]</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>100</td>
<td>1.097</td>
<td>262</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>2.234</td>
<td>229</td>
</tr>
<tr>
<td>200</td>
<td>300</td>
<td>6.589</td>
<td>220</td>
</tr>
<tr>
<td>200</td>
<td>400</td>
<td>8.9</td>
<td>200</td>
</tr>
<tr>
<td>1000</td>
<td>10</td>
<td>1.16</td>
<td>214</td>
</tr>
<tr>
<td>2000</td>
<td>10</td>
<td>2.36</td>
<td>223</td>
</tr>
</tbody>
</table>

RESULT AND DISCUSSION

The Proposed PSO

Despite the superior performance of metaheuristic algorithms, such as PSO, compared to heuristic algorithms, they tend to suffer from premature convergence when applied to complex real-life problems. This phenomenon occurs when particles representing potential solutions become trapped in local optimal points, resulting in a prolonged stagnation without any further improvement. Consequently, the quality of the obtained solutions cannot continuously improve, hindering the achievement of high-quality solutions. To address this issue and mitigate premature convergence, various modifications have been proposed in the literature to enhance the PSO algorithm (Song et al., 2021). In the conventional PSO algorithm, both pbest (personal best) and gbest (global best) are considered.

However, in the proposed algorithm, known as MPSO, an additional consideration is given to the best current answer, referred to as gcbest. The approach involves utilizing two coefficients, $a_1$ and $a_2$, for the solutions gbest and gcbest, respectively. The sum of these two variables always amounts to 100%. As the algorithm progresses, the values of these coefficients undergo changes. Additionally, three local search algorithms are taken into account to enhance the results further when gbest or pbest is updated. The steps of the proposed algorithm are outlined as follows: (Mao et al., 2017)

![Fig.2. steps of the classic version of PSO](Fig2.png)
Fig. 3. Moving particle in the PSO algorithm

1. Initialization

Initialization is the initial step in the Traveling Salesman Problem (TSP) where each particle represents a potential solution and signifies a permutation of the TSP customers. Initially, the algorithm commences by generating m random solutions. Subsequently, the objective function is applied to calculate the objective value for each particle, which is determined by summing the distances traveled by the salesman along all arcs. Additionally, m+2 variables are introduced, with the objective function’s values set to infinite positive. These m variables represent the best known locations for the particles. Furthermore, in each iteration, the other two variables, namely gbest and gcbest, are considered.

2. Diversification and Intensification Mechanisms

Diversification and Intensification Mechanisms play a crucial role in optimizing algorithms. Initially, the PSO algorithm demonstrated superior performance compared to heuristic and metaheuristic algorithms. However, as urbanization expanded and presented more complex challenges in the industry and services sectors, the classic version of the algorithm proved inadequate in solving these problems optimally. Consequently, modifications to the algorithm became necessary to enhance its efficiency. One notable weakness of the PSO algorithm was its tendency towards premature convergence. This limitation confined the algorithm to a specific range within the problem space, disregarding the global search and leading to suboptimal solutions.

In essence, the classical mode lacked a proper balance between exploring the entire problem space globally and conducting an effective local search to identify vulnerable areas. Conversely, metaheuristic algorithms have gained significant recognition in recent times due to several factors. Firstly, the increasing size of everyday problems faced by managers necessitates robust solutions. Secondly, the ability of these algorithms to address problems that customers or users prioritize adds to their importance. Lastly, the stability of the obtained answers across various problem scenarios benefits both managers and customers alike. Therefore, the objective is to develop sustainable algorithms capable of efficiently handling large-scale problems in the industry and services sectors within a reasonable timeframe. In other words, the solutions obtained through different algorithm performances should have minimal tolerance and be solvable with a maximum allowable error.

3. Updating gbest and pbest

The process of updating gbest and pbest occurs after all the particles have completed their movement in the previous step. For each particle, the pbest value is compared to the new answer obtained. If the new answer has a better value than the corresponding pbest, the pbest value is
updated. Furthermore, after updating all pbest values, if any of them surpasses the value of gbest, the gbest value is also updated. It is important to note that at the end of the algorithm, the value of gbest and its objective function are presented as the best answer and value of the problem.

4. Local Search Algorithms

In this section, we present four different local search algorithms, as illustrated in Figure 4. These algorithms are utilized when either the personal best (pbest) or the global best (gbest) is updated. It is worth noting that these algorithms are employed at this stage due to the possibility of finding better solutions in the vicinity of an already improved solution, which can be achieved by the algorithm [1]. The first algorithm, known as the inset move, involves moving a specific customer to a different location along the same path. The second algorithm, called the exchange move, selects two customers and swaps their positions. Additionally, the 2-Opt move operates by removing two edges from the cycle and reconnecting the two arcs in a different manner. Lastly, the 3-inverse move selects three customers and reverses their order in the desired solution. It is important to emphasize that each of these movements is accepted by the algorithm only if it leads to a better solution according to the respective method.

5. The Stop Condition

In this phase, similar to other metaheuristic algorithms, the examination of the stop criterion for terminating the algorithm takes place. Once this condition is met, the best solution and value of the problem will be determined by considering gbest and its objective function. Conversely, if the stop condition of the algorithm is not yet established, the algorithm process will be repeated until it is achieved. The pseudocode of the algorithm can be observed in Figure 5.

6. Computational Complexity of PPSO

To determine the computational complexity of the proposed algorithm, certain assumptions need to be made. Firstly, it is assumed that the number of iterations in the main body of the algorithm is denoted by 'n', and the initial population of particles is represented by 'm'. By focusing on the section of the code where the majority of operations are performed, the remaining parts of the code can be disregarded. This particular section involves the utilization of local search algorithms. The complexity of the main loop is denoted as O(n), indicating that the time required for execution increases linearly with the number of iterations. Additionally, all particles need to be examined to identify any changes in their personal best (pbest), resulting in a complexity of O(m) for this particular section. Furthermore, the 3-inverse algorithm, which is employed in the proposed algorithm, is more intricate compared to other local search algorithms.

This is due to the fact that it needs to select 3 customers from a pool of 'm' individuals. Consequently, the complexity of this part is O(m³). Taking all these factors into consideration, the overall complexity order of the proposed algorithm can be expressed as O(nm⁴). This signifies that the computational time required by the algorithm increases exponentially with the number of iterations and the size of the initial particle population.

![Fig. 4. the insert, exchange, 2-Opt, and 3-inverse moves](image-url)
Experiment Result

In this section, we will present the outcomes of the suggested algorithm for two distinct categories of examples. Furthermore, we will compare the results obtained by the proposed algorithm with those of well-known metaheuristic algorithms in the second set of instances. It is worth mentioning that all the code for the proposed algorithm has been developed using Python 3. The implementation of these programs has been carried out on a laptop equipped with an Intel (R) Core (TM) i3 CPU running at 2.53 GHz and 4 GB of memory. Additionally, this section will delve into the discussion of parameter settings followed by the presentation of our findings.

PARAMETER SETTINGS

The solutions generated by the MPSO, like any other metaheuristic algorithm, are influenced by the seed used for generating pseudorandom numbers and the parameter values of the algorithm. Our proposed algorithm consists of only four parameters, and the values of these parameters have a direct or indirect impact on the quality of the final solution. To achieve the optimal balance between solution quality and computational effort, a parameter setting procedure is necessary. It is important to note that there is no definitive way to determine the most effective parameter values. Therefore, in this study, similar to many others, the parameter values are selected after extensive testing. Through experimentation, it has been determined that the number of populations and the stop condition are the most influential experimental parameters in MPSO that directly affect the quality of the final solution. Various strategies and approaches can be employed to integrate business management with computer science, with adopting digital technologies being a crucial aspect of this integration. This involves leveraging digital tools, platforms, and systems to enhance various aspects of business operations. Below are some key strategies and approaches:

```
Input = n, m, and customer characteristics.
1- Generate m random solutions for the TSP problem and find their objective functions.
2- Find the pbest, gbest and gbest solutions.
3- Repeat
   (i) Move all particles toward to the pbest, gbest or gbest solutions according to the procedure.
   (ii) For each particle, update pbest, then update gbest.
   (iii) If the pbest or gbest is updated, all local search algorithms are used.
4- Until the stop condition is satisfied.
Output = The gbest with its objective function.
```

Fig. 5. Steps of the proposed algorithm

Fig. 6. Parameters For PSO
CONCLUSION

In this paper, the TSP problem was tackled using a modified PSO algorithm known as MPSO. To assess the effectiveness of this method, the worst, average, and best solutions were compared to the classic PSO across a range of standard problems with diverse customer sets. Given that multiagent algorithms typically struggle with the intensification mechanism, several moves such as insert, exchange, 2-Opt, and inverse were incorporated. Furthermore, certain adjustments were made to the PSO algorithm to enhance its search capabilities and avoid local optima. The results demonstrate the algorithm’s efficiency when compared to other methods proposed for standard problems. In future endeavors, this algorithm holds potential for addressing vehicle routing and allocation problems, particularly as its efficiency improves with an increase in the number of customers. However, to ensure stability, it is advisable to employ the tabu search algorithm to enhance the intensification mechanism. Additionally, optimizing the movement of particles towards gbest and pbest can enhance the global search within the problem space. Lastly, various computational intelligence algorithms such as monarch butterfly optimization, earthworm optimization algorithm, elephant herding optimization, moth search algorithm, slime mould algorithm, and harris hawks optimization can be explored to solve the TSP problem. However, the application of these ideas will be deferred to future articles.

REFERENCES


