**JTAM (Jurnal Teori dan Aplikasi Matematika)**
http://journal.ummat.ac.id/index.php/jtam
p-ISSN 2597-7512 | e-ISSN 2614-1175
Vol. 9, No. 4, October 2025, pp. 1082-1093

# Enhancing Adaptive Particle Swarm Optimization Based on Human Social Learning with Human Learning Strategies for the Traveling Salesman Problem

**Yusti Qomah[1*], Bib Paruhum Silalahi[1], Toni Bakhtiar[1]**
[1]School of Data Science, Mathematics, and Informatics, IPB University, Indonesia
qomahyq25yusti@apps.ipb.ac.id

**Article History:**
Received : 14-05-2025
Revised : 17-07-2025
Accepted : 18-07-2025
Online : 01-10-2025

**Keyword:**
Particle Swarm Optimization;
Human Learning Optimization;
Traveling Salesman Problem;
Operations Research.

**ABSTRACT**

Particle Swarm Optimization (PSO) is a widely used metaheuristic approach for solving optimization problems. Recent developments in this field involve the adaptation of human learning behaviors to enhance algorithmic performance. One such adaptation is the Adaptive Particle Swarm Optimization based on Human Social Learning (APSO-HSL), a variant of PSO that incorporates human inspired learning strategies. This study aims to enhance the performance of APSO-HSL on the Traveling Salesman Problem (TSP) by incorporating additional human learning strategies. The proposed algorithm, named Modified Adaptive Particle Swarm Optimization–Human Learning Strategies (MAPSO-HLS), integrates learning mechanisms from Human Learning Optimization (HLO), including individual, random, and social learning. This research is classified as applied research and algorithmic experimentation, focusing on the development and modification of a metaheuristic algorithm to solve a well-known combinatorial optimization problem. Benchmark datasets from the Traveling Salesman Problem Library (TSPLIB) are used for evaluation, and all computations and experiments are implemented in Python. The performance of MAPSO-HLS is compared with the exact method in terms of shortest distance and computation time. The results of the study indicate that the MAPSO-HLS algorithm is capable of producing TSP solutions with low total distance deviation, below 10%, compared to exact solutions across all tested datasets. This reflects a high level of solution accuracy. In addition, MAPSO-HLS demonstrates better time efficiency than the exact ILP method, particularly for datasets with a large number of cities. The integration of human learning strategies within the adaptive PSO framework provides significant advantages in terms of both efficiency and effectiveness in solving TSP.

https://doi.org/10.31764/jtam.v9i4.31466

**This is an open access article under the CC–BY-SA license**

## A. INTRODUCTION

Particle Swarm Optimization (PSO) is a meta-heuristic algorithm introduced by Kennedy and Eberhart and is widely used to solve optimization problems. The PSO algorithm is based on natural phenomena, namely the movement of a group of living things, such as birds and fish in finding food (Houssein et al., 2021). Computationally, PSO has advantages in terms of memory usage and speed (Jiyue et al., 2023; Lynn & Suganthan, 2015; Punyakum et al., 2022). The simplicity and efficiency factors have caused PSO to be widely used in various fields and it is considered the most effective method for solving optimization problems (Bangyal et al., 2023). PSO implementations are widely found in various fields, such as health, environment, industry, and commerce (Al-Maamari & Omara, 2015; Ramdhani, 2016).

Despite its strengths, standard PSO often suffers from premature convergence, particularly when parameters are not properly tuned (Guo et al., 2025; Larsen et al., 2016; Ashraf et al., 2022). This leads to limited global exploration, insufficient local exploitation, and reduced solution diversity. To address these limitations, several PSO variants have been proposed, including heterogeneus comprehensive learning PSO (Lynn & Suganthan, 2015), discrete PSO (Zhong et al., 2018), and improved PSO with adaptive initialization techniques (Ashraf et al., 2022), each aiming to enhance convergence and avoid local optima.

A promising direction to overcome these challenges is the integration of human learning behavior into algorithmic design (Roberts-Mahoney et al., 2016; Jarecki et al., 2018; Wang et al., 2017; Du et al. 2022). Human Learning Optimization (HLO) introduces learning strategies inspired by how humans solve problems, through random trial, individual experience, and social imitation. These principles have been shown to improve convergence and solution diversity in complex search spaces (Wang et al., 2014). Building on this, the Adaptive Particle Swarm Optimization based on Human Social Learning (APSO-HSL) algorithm incorporates human social learning into the PSO framework, resulting in improved accuracy, stability, and global search performance (Jiyue et al., 2023). Nonetheless, current implementations of APSO-HSL primarily focus on the social learning component, leaving the potential of individual and random learning strategies underexplored.

Solving combinatorial problems such as the Traveling Salesman Problem (TSP), a classic NP-hard problem requiring the optimal traversal of cities, is a compelling application for metaheuristics (Shaj et al., 2016; Chen et al., 2025). TSP not only represents real-world complexity but also demands a balance between exploration and exploitation due to its exponentially growing solution space (Jedrzejowicz et al., 2024). Although many PSO variants have been applied to TSP, further exploration into the role of adaptive human-inspired learning remains limited.

This study aims to enhance the APSO-HSL algorithm by integrating comprehensive human learning strategies from HLO, namely random, individual, and social learning, to improve the effectiveness of PSO in solving the TSP. Based on (Vahdat et al., 2016), the combination of meta-heuristic algorithms with adaptive learning mechanism has many advantages in solving optimization problems compared to algorithms inspired by natural phenomena. By extending APSO-HSL with a more complete representation of human learning behavior, the proposed approach addresses the gap in existing PSO-based methods that insufficiently balance exploration and exploitation.

## B. METHODS

This research is categorized as applied research and algorithmic experimentation, focusing on the development and modification of a metaheuristic optimization to solve a well-known combinatorial optimization problem (TSP). The dataset utilized in this research comprises benchmark instances sourced from the Traveling Salesman Problem Library (TSPLIB). Seven case examples with varying numbers of cities were selected, namely burma14, eil51, berlin 52, st70, eil76, and pr76 (*TSPLIB*, n.d.). All experiments were conducted using an Intel PC (core i5 @3.09GHz CPU, 4GB RAM). The performance of the modified algorithm was evaluated based on the total distance traveled and computation time. A comparative analysis was conducted

against the exact method using benchmark TSP instances using the percentage deviation $PD$ shown by Equation (1). The exact method used in this study is Integer Linear Programming (ILP).

$$PD = \frac{(\text{approximation} - \text{exact})}{\text{exact}}\ 100\% \tag{1}$$

## 1. Particle Swarm Optimization (PSO)

The PSO algorithm begins with the particle initialization and initial velocity assignment. The optimal function value and location can be found by using the initial velocity to assess the objective function at each particle location. The particle velocity at that moment, each particle's optimal location, and each particle's optimal neighboring location are then used to determine a new velocity. The particle's location, velocity, and neighbors are updated iteratively until the algorithm reaches the stopping criterion. The new location was determined by adding the old location and modified velocity to maintain the particle within the boundary (Ab Wahab et al., 2015; Jain et al., 2018; Gad, 2022).

To obtain the best solution, each particle moves based on its personal best position ($p_{best}$) and the global best position ($g_{best}$) in the swarm. Each particle $i$ will update its velocity $v$ and position $x$ in each iteration $t+1$ using the following equation:

$$v_i^{t+1} = \omega v_i^t + c_1 r_1\left(p_{best_i}^t - x_i^t\right) + c_2 r_2 (g_{best}^t - x_i^t), \tag{2}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1}, \tag{3}$$

where $i$ denotes the particle index, $t$ is the current iteration order, $\omega$ is the inertia used to balance local exploitation and global exploration, $r_1$ and $r_2$ is any number uniformly distributed on [0,1], $c_1$ and $c_2$ are the acceleration coefficients, which are two positive constants. Equation (2) is used to calculate the new velocity of the particle based on the distance and last velocity of a position from the particle's best personal experience and the group's best experience. Then, the particle moves towards the new position based on Equation (3) (Piotrowski et al., 2020; Zhong et al., 2018).

## 2. Adaptive Particle Swarm Optimization based on Social Learning Intelligence (APSO-HSL)

The APSO-HSL algorithm developed by Jiyue et al. (2023) uses a learning strategy based on the multiswarm technique which the diversity of each particle is determined by the division of the swarm and the size of each swarm. Human social learning intelligence is used to adaptively divide the swarm and determine the size of each subswarm. The multiswarm technique in APSO-HSL has the following mathematical definition:

$$\begin{aligned} f_1 &= f_{max}, \\ f_2 &= f_{\min} + \alpha_1 (f_{max} - f_{\min}), \\ f_3 &= f_{\min} + \alpha_2 (f_{max} - f_{\min}), \\ f_4 &= f_{\min} + \alpha_3 (f_{max} - f_{\min}), \\ f_5 &= f_{\min}, \end{aligned} \tag{4}$$

$$\Omega_j^{\mathrm{t}} = \{x_i^t \mid f_j \leq \text{fitness}\,(x_i^t) < f_{j+1}, i = 1, 2, \ldots, N; j = 1, 2, 3, 4\}.$$

$f_{\min}$ and $f_{max}$ are the minimum and maximum values of the fitness function. $\{f_k, k = 1, 2, \ldots, 5\}$ denotes the subswarm boundaries. $\Omega_j^{\mathrm{t}}$ denotes the $j$th subswarm at the $t$th iteration. $x_i^t$ is the position of the $i$th particle at the $t$th iteration. The upper and lower bounds for each particle's fitness are determined using $\alpha_k$ with $k = 1, 2, 3$, where $\alpha_1 = 0.25, \alpha_2 = 0.50, \alpha_3 = 0.75$. The determination of the upper and lower bounds were determined to avoid a large size imbalance of the subswarms.

Every particle was split into three groups according to their learning capacity, each of which had a distinct purpose based on how well they evolved with each iteration. The first group is made up of the swarm's best particles overall ($Gbest$). The second is each subswarm's best particle ($Sbest$). The remaining particles with average fitness, referred to as ordinary particles, make up the third group. The following is a definition of the inertia load operators for the three groups:

$$\omega_i^t = \begin{cases} \dfrac{\sum_{j=1}^{N(t)-1} fitness\left(Sbest_j^t\right)}{(N(t)-1)} \Big/ fitness(x_i^t), & \text{if } x_i^t = Gbest^t \\ fitness\left(Sbest_{j+1}^t\right) \Big/ fitness(x_i^t), & \text{if } x_i^t = Sbest_j^t \\ fitness\left(Sbest_{S_{(i)}}^t\right) \Big/ fitness(x_i^t), & \text{else,} \end{cases} \tag{5}$$

where $N(t)$ represents the number of subswarms, $Sbest_j^t$ shows the best particle of the $j$th subswarm at the $t$th iteration, $S_{(i)}^t$ shows the subswarm with the position of the $i$th particle located at the $t$th iteration. Groups with different learning abilities also have different ways of updating their velocity. Mathematically, the particle velocity in the group with the first learning ability can be defined using the following equation:

$$v_i^{t+1} = \omega_i^t v_i^t + c_{11} r_1 (Pbest_i^t - x_i^t) + c_{21} r_2 \left(AVGSbest^t - x_i^t\right), \tag{6}$$

where $c_{11} = 2$ and $c_{21} = 1$ denote positive acceleration coeeficients, $r_1$and $r_2$ are two arbitrary numbers distributed uniformly on [0,1]. $Pbest_i^t$ denotes the best position of each particle and $AVGSbest^t$ is the average of the best positions of the particles in each subswarm which is defined as:

$$\begin{cases} AVGSbest_i^t = \sum_{j=1}^{N(t)-1} r_i^t \, x_i^t \\ r_i^t = \dfrac{\left(1 - \dfrac{fitness(Sbest_i^t)}{\sum_{j=1}^{N(t)-1} fitness\left(Sbest_j^t\right)}\right)}{N(t) - 2} \end{cases} \tag{7}$$

Furthermore, it was found:

$$\sum_{j=1}^{N(t)-1} r_i^t = 1 \tag{8}$$

The particle velocity in the group with the second learning ability is defined by the following equation:

$$v_i^{t+1} = \omega_i^t v_i^t + c_{12} r_1 (Pbest_i^t - x_i^t) + c_{22} r_2 \left(Sbest_{S(i)+1}^t - x_i^t\right), \tag{9}$$

where $c_{12} = 1$ and $c_{22} = 2$ are the acceleration coefficients and $Sbest_{S(i)+1}^t$ denote the learning characteristics of the best particle in the subswarm with a lower fitness level. Ordinary particles update their velocities using:

$$v_i^{t+1} = \omega_i^t v_i^t + c_{13} r_1 (Pbest_i^t - x_i^t) + c_{23} r_2 \left(Sbest_{S(i)+1}^t - x_i^t\right), \tag{10}$$

where $c_{13} = c_{23} = 1.5$ (Jiyue et al., 2023).

## 3. Human Learning Optimization (HLO)

The adaptive algorithm known as Human Learning Optimization (HLO) is based on activities of human learning. To master new things, humans repeatedly learn and practice them. After determining an appropriate learning method, humans can evaluate the recognition of a new object. The learning model used in HLO includes three learning stategy, random learning strategy, individual learning strategy, and social learning strategy (Wang, Ni, et al., 2015; Ding dan Gu 2020). At the initial stage of a novel task, individuals typically lack prior knowledge, resulting in behavior characterized by trial and error. Within the framework of Human Learning Optimization (HLO), this phase is conceptualized as random learning, wherein actions are guided by uninformed guesses. As individuals accumulate experience through repeated attempts, they begin to discern effective strategies from ineffective ones based on personal outcomes. This process aligns with what is referred to in HLO as individual learning. Beyond self-experience, learning also occurs through observation and interaction with others. When individuals are exposed to peers with superior performance or greater experience, they are inclined to adopt or imitate those observed strategies. This behavior is captured in HLO as social learning, reflecting the influence of social context on cognitive adaptation (Wang, Ni, et al., 2015; Wang et al., 2018). The three learning processes are determined based on the probability of random learning $p_r$ and the probability of individual learning $p_{in}$. Once a random number $rand \in (0,1)$ is generated, the selection of the learning process is governed according to Equation (11) (Wang, Yang, et al., 2015).

$$\begin{cases} \text{individual learning}, & \text{if } rand \in (0, p_r) \\ \text{random learning}, & \text{if } rand \in [p_r, p_{in}] \\ \text{social learning}, & \text{else} \end{cases} \tag{11}$$

## 4. Modified Adaptive Particle Swarm Optimization-Human Learning Strategies (MAPSO-HLS)

The proposed algorithm is a combination of APSO-HSL and HLO called Modified Adaptive Particle Swarm Optimization-Human Learning Strategies (MAPSO-HLS). Every particle with a different role is given three types of learning. The learning process begins with individual learning, where each particle evaluates its own historical experience to refine movement strategies. This self-guided adaptation helps particles identify locally promising areas based on their own performance. The next learning process is random learning, which introduces stochastic exploration, directs particle to study from $K$ neighbors best experience, based on their previous experience. $K$ neighbors used as learning references were randomly selected. Finally, social learning enables particles to adopt strategies from particles showing better performance, promoting convergence toward globally optimal regions.

The velocity and position particle-$i$ at the $t+1$th iteration represented by $v_i^{t+1}$ and $x_i^{t+1}$ for the best particle in the entire swarm is given by Equation (12), for the best particle in each subswarm can be updated using Equation (13), and for ordinary particles can be updated using Equation (14). $\omega$ is the moment of inertia to balance local exploitation and global exploration calculated using Equation (5), $r_1$ and $r_2$ is any number uniformly distributed on [0,1]. $c_{11}, c_{12}, c_{13}$ are the individual acceleration coefficients, and $c_{21}, c_{22}, c_{23}$ are the social acceleration coefficients. $Pbest_i^t$ shows the best position of each particle and $AVGSbest^t$ is the average of the best positions of particles in each subswarm, $g_K$ shows the best position from any $K$ particle.

$$v_i^{t+1} = \begin{cases} \omega_i^t v_i^t + c_{11} r_1 (Pbest_i^t - x_i^t) + c_{21} r_2 \left(AVGSbest^t - x_i^t\right) p_{rn}; \ p_{rn} \in (0, p_{in}) \\ \omega_i^t v_i^t + c_{11} r_1 (Pbest_i^t - x_i^t) + c_{21} r_2 (g_K - x_i^t); \ p_{rn} \in [p_{in}, p_{sc}] \\ \omega_i^t v_i^t + c_{11} r_1 (Pbest_i^t - x_i^t)(1 - p_{sc}) + c_{21} r_2 \left(AVGSbest^t - x_i^t\right); \ p_{rn} \in (p_{sc}, 1) \end{cases} \quad (12)$$

$$v_i^{t+1} = \begin{cases} \omega_i^t v_i^t + c_{12} r_1 (Pbest_i^t - x_i^t) + c_{22} r_2 \left(Sbest_{S(i)+1}^t - x_i^t\right) p_{rn}; \ p_{rn} \in (0, p_{in}) \\ \omega_i^t v_i^t + c_{12} r_1 (Pbest_i^t - x_i^t) + c_{22} r_2 (g_K - x_i^t); \ p_{rn} \in [p_{in}, p_{sc}] \\ \omega_i^t v_i^t + c_{12} r_1 (Pbest_i^t - x_i^t)(1 - p_{sc}) + c_{22} r_2 \left(Sbest_{S(i)+1}^t - x_i^t\right); \ p_{rn} \in (p_{sc}, 1) \end{cases} \quad (13)$$

$$v_i^{t+1} = \begin{cases} \omega_i^t v_i^t + c_{13} r_1 (Pbest_i^t - x_i^t) + c_{23} r_2 \left(Sbest_{S(i)}^t - x_i^t\right) p_{rn}; \ p_{rn} \in (0, p_{in}) \\ \omega_i^t v_i^t + c_{13} r_1 (Pbest_i^t - x_i^t) + c_{23} r_2 (g_K - x_i^t); \ p_{rn} \in [p_{in}, p_{sc}] \\ \omega_i^t v_i^t + c_{13} r_1 (Pbest_i^t - x_i^t)(1 - p_{sc}) + c_{23} r_2 \left(Sbest_{S(i)}^t - x_i^t\right); \ p_{rn} \in (p_{sc}, 1) \end{cases} \quad (14)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1}, \quad (15)$$

Similar to HLO, the three learning strategies are independent of one another. Determination of the learning process based on the $p_{in}$, $p_{sc}$, and $p_{rn}$. The values $(p_{in} - 0)$ indicate the occurrence of individual learning processes, $(p_{sc} - p_{in})$ for random learning, and $(1 - p_{sc})$ for social learning, respectively. Parameter $p_{rn}$ will be determined randomly at the interval (0,1). In the context of TSP, velocity updates must be redefined as swap operations due to the discrete and permutation-based nature of the problem (Clerc, 2004). If the fitness value from the updated position is better than before, the position becomes the particle's $Pbest$. The 2-opt algorithm is then applied to further refine the solution. In TSP, 2-opt enhances solution quality by reducing inefficient or intersecting paths often found in PSO results. Its low complexity and

deterministic nature make it ideal for improving convergence in global search algorithms like PSO and its variants (Sathyan et al., 2015).

## C. RESULT AND DISCUSSION

The chosen correlation coefficients were $c_{11} = 2$and $c_{21} = 1$ for the best particle in the entire swarm, $c_{12} = 1$ and $c_{22} = 2$ for the best particle in each subswarm, and $c_{13} = c_{23} = 1.5$ for ordinary particle. The learning behavior of each particle is determined by the probabilities $p_i = 0.4$ and $p_s = 0.8$. Each particle with free learning strategy learns from the best experience among $K = 5$ randomly selected neighbors. The number of particles used in this study was 100, with a total of 1000 iterations. The effectiveness of MAPSO-HLS was assessed through experiments on six TSP instances with different city counts, with performance evaluation conducted based on total route length and computation time. The approximation results of the shortest distances generated by MAPSO-HLS were evaluated against the exact method through percentage deviation, as shown in Table 1. A smaller percentage deviation indicates a higher accuracy of the algorithm in finding the optimal solution.

**Table 1.** Comparison of Total Distance

| Instances | Solution | | Percentage Deviation ($PD$) |
|---|---|---|---|
| | Exact | MAPSO-HLS | |
| burma14 | 30.88 | 30.88 | 0.00% |
| eil51 | 429.98 | 445.03 | 3.50% |
| berlin52 | 7544.37 | 7811.31 | 3.54% |
| st70 | 678.60 | 694.09 | 2.28% |
| eil76 | 544.37 | 573.44 | 5.34% |
| pr76 | 108159.44 | 113925.86 | 5.33% |

The percentage deviations presented in Table 1 indicate that MAPSO-HLS demonstrates good performance. The deviations across all six TSP datasets are below 10%. For the dataset with a small number of cities (burma14), the deviation is 0.00%. Even for datasets with a larger number of cities (more than 50), the deviations remain low, ranging from 2.28% to 5.34%. These results suggest that MAPSO-HLS is capable of producing accurate solutions with only slight differences compared to the exact solutions.

Computation time is also a key factor in evaluating algorithm performance. The computation time ratio presented in Table 2 shows that MAPSO-HLS is significantly faster than the exact method, particularly for datasets with more than 50 cities. For the berlin52 dataset, MAPSO-HLS produced a solution in 92.63 seconds, whereas the exact method required 1178.57 seconds. This indicates that MAPSO-HLS is 12.72 times faster. Similarly, on the eil51, eil76, pr76, and st70 datasets, MAPSO-HLS also achieved shorter computation times, being 8.97, 1390.69, 2164.18, and 3888.11 times faster, respectively. The percentage deviation also highlights that computation time for datasets with more than 50 cities is significantly faster using MAPSO-HLS compared to the exact method. This is evident in the fact that the computation time deviation for the eil51, eil76, pr76, and st70 datasets, the deviation in computation time is extremely high, approaching 100%.

**Table 2.** Comparison of Computation Time

| Instances | Time (seconds) | | Ratio | Percentage Deviation (*PD*) |
|---|---|---|---|---|
| | Exact | MAPSO-HLS | | |
| burma14 | 1.22 | 45.33 | 0.03 | 3615.57% |
| eil51 | 711.09 | 79.29 | 8.97 | 88.85% |
| berlin52 | 1178.57 | 92.63 | 12.72 | 92.14% |
| st70 | 589243.03 | 151.55 | 3888.11 | 99.97% |
| eil76 | 206754.48 | 148.67 | 1390.69 | 99.93% |
| pr76 | 368235.35 | 170.15 | 2164.18 | 99.95% |

Based on the percentage deviation in total distance and the computation time ratio, MAPSO-HLS demonstrates an advantage over exact methods in terms of time efficiency while maintaining solution accuracy. Therefore, MAPSO-HLS can be considered a viable alternative for solving discrete permutation problems such as the TSP, particularly for datasets involving a large number of cities. Figures 1 and Figure 2 illustrate the shortest route for eil51 obtained using the exact method and MAPSO-HLS, respectively.

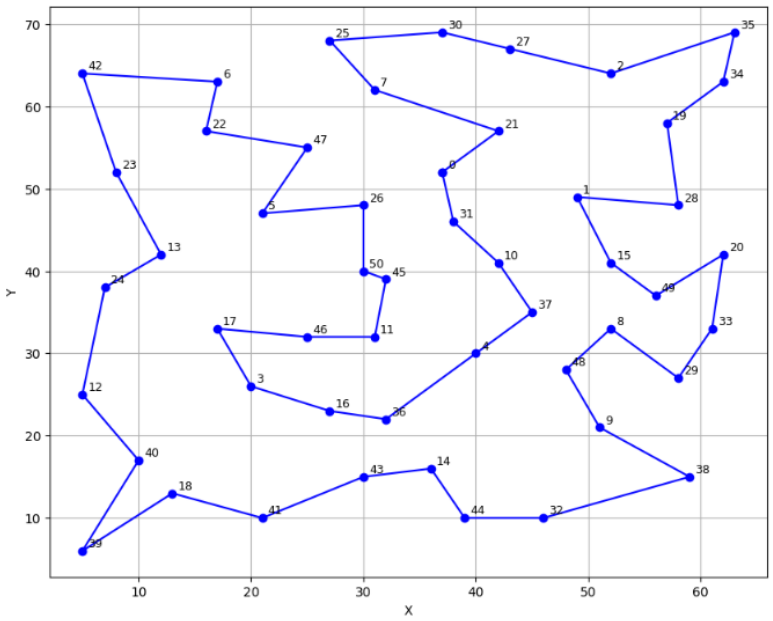



**Figure 1.** The Shortest Route for eil51 Based on the Exact Method

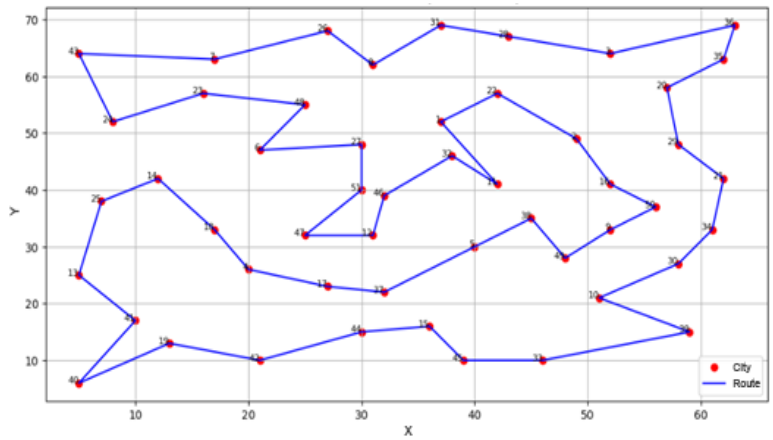



**Figure 2.** The Shortest Route for eil51 Based on the MAPSO-HLS Algorithm

This study continues by evaluating the parameters $p_{in}$ and $p_{sc}$ to observe their impact on the total distance traveled and computation time. The parameters $p_{in}$ and $p_{sc}$ represent the probabilities of individual learning and social learning, respectively. These probabilities control the balance between exploration and exploitation. Exploration refers to the condition where particles are still searching for new areas in the solution space, while exploitation refers to the condition in which the swarm refines solutions that are already reasonably good. All combinations of $p_{in}$ and $p_{sc}$ ranging from 0.1 to 0.9 were applied to six TSP datasets, with the constraint that $p_{in} < p_{sc}$. The relationship between these parameters and the average shortest distances obtained across the six datasets is visualized in Figure 3.

As shown in Figure 3, the parameter $p_i$ corresponds to $p_{in}$ and $p_s$ corresponds to $p_{Sc}$. It can be observed that when the difference between $p_{in}$ and $p_{sc}$ is either too small or too large, the resulting average distance tends to be higher. For example, the combinations $p_{in} = 0.1$ and $p_{sc} = 0.2$, $p_{in}$=0.1 and $p_{sc}$=0.9, $p_{in}$=0.8 and $p_{sc}$=0.9 produce larger average distances compared to other combinations. The best performance is observed at $p_{in}$=0.1 and $p_{sc}$=0.8, while the worst performance is found at $p_{in}$ =0.1 and $p_{sc}$ =0.2. Lower average distances were also obtained with $p_{in}$=0.1 and $p_{sc}$=0.5, and $p_{in}$=0.2 and $p_{sc}$=0.5. These results suggest that when $p_{in}$ is low (0.1 or 0.2), particles do not overly rely on their own learning, helping to avoid local optima. Meanwhile, when $p_{Sc}$ is set to moderate values (0.5 or 0.6), particles engage in social learning at a balanced rate, promoting convergence while preserving diversity. This combination leads to a more balanced process between exploration and exploitation.

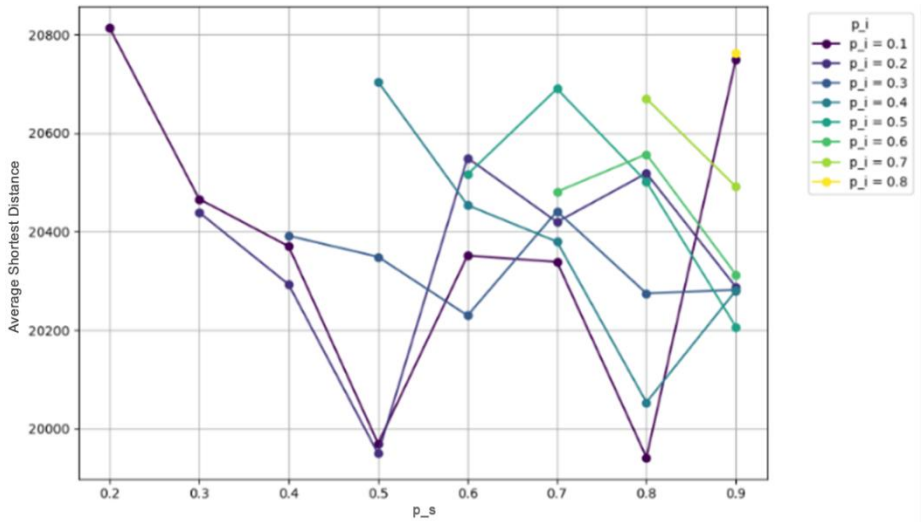



**Figure 3.** Effect of $p_{in}$ and $p_{sc}$ on Average Total Distance

The relationship between the parameters $p_{in,}$ $p_{sc}$ and the average computation time is visualized in Figure 4. The parameter $p_i$ corresponds to $p_{in}$, and $p_s$ corresponds to $p_{sc}$. Combinations of $p_{in}$ and $p_{sc}$ with small differences tend to result in shorter computation times. For example, when $pin$=0.1, the combination with $psc$=0.2 yields faster computation compared to combinations with other $p_{sc}$ values. A similar pattern is observed for $p_{in}$ =0.2, where combining it with $p_{sc}$=0.3 results in shorter computation time. The larger the gap between $p_{in}$ and $p_{sc}$, the more time is required to reach a solution. The best-performing combination in

terms of computation time is $p_{in}$=0.3 and $p_{sc}$=0.4, while the worst-performing combination is $p_{in}$=0.1and $p_{Sc}$=0.9.

The increasing computation time associated with higher $p_{sc}$ values is caused by the greater frequency of social learning, which leads to more frequent information exchange among particles. When $p_{in}$=0.6, 0.7, or 0.8, the resulting computation time is lower. However, as shown in Figure 3, the quality of the solutions obtained is less optimal. This indicates that when particles tend to rely more on their own experience, convergence is faster and computation time is shorter, but the solutions are suboptimal. Based on the average total distance and computation time, the combinations $p_{in}$=0.1 with $p_{sc}$=0.5 or 0.6 are preferable when accuracy is the main objective. If time efficiency is prioritized, combinations such as $p_{in} > 0.6$ with $p_{sc}$=0.6 or 0.7 can be selected. To achieve a balance between optimal solution quality and shorter computation time, the combinations $p_{in}$ =0.3 or 0.4 with $p_{sc}$ =0.5 or 0.7 are recommended.

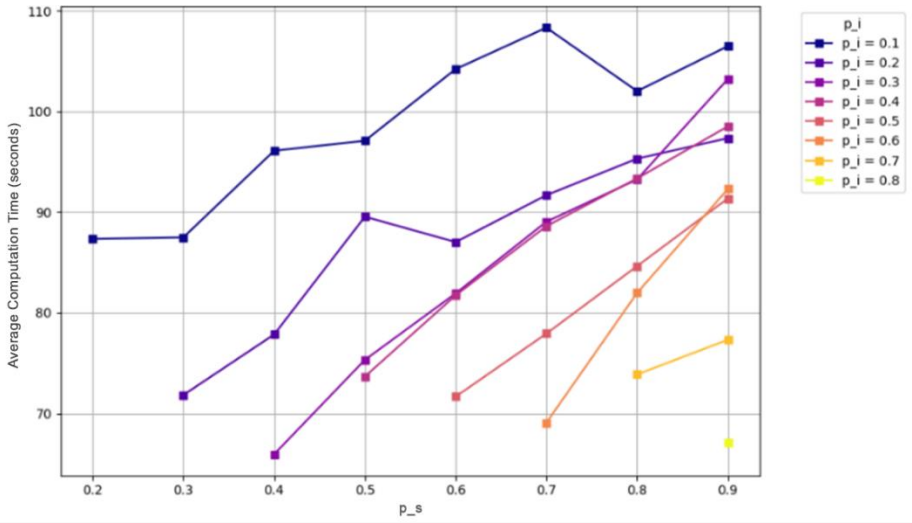



**Figure 4.** Effect of $p_{in}$ and $p_{sc}$ on Average Computation Time

## D. CONCLUSION AND SUGGESTIONS

This study introduces MAPSO-HLS, a modified APSO-HSL algorithm that integrates human learning strategies, individual, random, and social learning, adapted from the HLO framework. The integration of these three learning mechanisms enhances the algorithm's ability to balance exploration and exploitation. This balanced dynamic allows MAPSO-HLS to avoid premature convergence and more effectively navigate complex solution spaces. The percentage deviation of total distance from the six TSP datasets, when compared to the exact solutions, indicates that MAPSO-HLS produces relatively low deviations, under 10%. This demonstrates the accuracy of the solutions obtained using MAPSO-HLS. In terms of computation time, MAPSO-HLS outperforms the exact ILP method by requiring significantly less time, especially for datasets with a large number of cities.

A proper balance between exploration and exploitation can be achieved by selecting appropriate combinations of the parameters $p_{in}$ and $p_{sc}$. The parameter analysis shows that choosing more adaptive values enhances the trade-off between solution quality and computational efficiency. A smaller $p_{in}$ value can reduce the risk of getting trapped in local

optima, while a larger $p_{sc}$ value can reduce computation time. Further research is recommended to incorporate statistical testing in evaluating the performance of the modified algorithm, and to compare it with both the original version and other PSO variants. The algorithm can also be implemented in more complex and practical problem domains, using datasets derived from real-world scenarios. Additionally, other factors that influence human learning behavior, such as motivation level, age, and experience, may be explored and adapted into the algorithm to further enhance its performance.

## REFERENCES

Ab Wahab, M. N., Nefti-Meziani, S., & Atyabi, A. (2015). A comprehensive review of swarm optimization algorithms. *PLoS ONE*, *10*(5), 1-36. https://doi.org/10.1371/journal.pone.0122827

Al-Maamari, A., & Omara, F. A. (2015). Task scheduling using PSO algorithm in cloud computing environments. *International Journal of Grid and Distributed Computing*, *8*(5), 245–256. https://doi.org/10.14257/ijgdc.2015.8.5.24

Ashraf, A., Almazroi, A. A., Bangyal, W. H., & Alqarni, M. A. (2022). Particle swarm optimization with new initializing technique to solve global optimization problems. *Intelligent Automation and Soft Computing*, *31*(1), 191–206. https://doi.org/10.32604/IASC.2022.015810

Bangyal, W. H., Nisar, K., Soomro, T. R., Ag Ibrahim, A. A., Mallah, G. A., Hassan, N. U., & Rehman, N. U. (2023). An improved particle swarm optimization algorithm for data classification. *Applied Sciences (Switzerland)*, *13*(1), 1-18. https://doi.org/10.3390/app13010283

Chen, D., Imdahl, C., Lai, D., & Van Woensel, T. (2025). The Dynamic Traveling Salesman Problem with Time-Dependent and Stochastic travel times: A deep reinforcement learning approach. *Transportation Research Part C: Emerging Technologies*, *172*, 105022. https://doi.org/10.1016/J.TRC.2025.105022

Clerc, M. (2004). Discrete particle swarm optimization, illustrated by the traveling salesman problem. In *Studies in Fuzziness and Soft Computing* (pp. 219–239). https://doi.org/10.1007/978-3-662-07814-8_10

Ding, H., & Gu, X. (2020). Hybrid of human learning optimization algorithm and particle swarm optimization algorithm with scheduling strategies for the flexible job-shop scheduling problem. *Neurocomputing*, *414*, 313–332. https://doi.org/10.1016/j.neucom.2020.07.004

Du, J., Wang, L., Fei, M., & Ilyas, M. (2022). A human learning optimization algorithm with competitive and cooperative learning. *Complex & Intelligent Systems*, *9*(1), 797-823. https://doi.org/10.1007/s40747-022-00808-4

Gad, A. G. (2022). Particle swarm optimization algorithm and its applications: a systematic review. *Archives of Computational Methods in Engineering*, *29*(5), 2531–2561. https://doi.org/10.1007/s11831-021-09694-4

Guo, W., Li, L., Chen, M., Ni, W., Wang, L., & Li, D. (2025). Balancing convergence and diversity preservation in dual search space for large scale particle swarm optimization. *Applied Soft Computing*, *169*, 112617. https://doi.org/10.1016/J.ASOC.2024.112617

Houssein, E. H., Gad, A. G., Hussain, K., & Suganthan, P. N. (2021). Major Advances in Particle Swarm Optimization: Theory, Analysis, and Application. *Swarm and Evolutionary Computation*, *63*, 100868. https://doi.org/10.1016/J.SWEVO.2021.100868

Jain, N. K., Nangia, U., & Jain, J. (2018). A review of particle swarm optimization. In *Journal of The Institution of Engineers (India): Series B* (Vol. 99, Issue 4, pp. 407–411). Springer. https://doi.org/10.1007/s40031-018-0323-y

Jarecki, J. B., Meder, B., & Nelson, J. D. (2018). Naïve and robust: class-conditional independence in human classification learning. *Cognitive Science*, *42*(1), 4–42. https://doi.org/10.1111/cogs.12496

Jedrzejowicz, P., Keller, K., Skakovski, A. (2024). An efficient hybrid evolutionary algorithm for solving the traveling salesman problem. *Procedia Computer Science*, *246(C)*, 3566-3574. https://doi.org/10.1016/j.procs.2024.09.201

Jiyue, E., Liu, J., & Wan, Z. (2023). A novel adaptive algorithm of particle swarm optimization based on the human social learning intelligence. *Swarm and Evolutionary Computation*, *80*, 101336. https://doi.org/10.1016/j.swevo.2023.101336

Larsen, R. B., Jouffroy, J., & Lassen, B. (2016). On the premature convergence of particle swarm optimization. *2016 European Control Conference, ECC 2016*, 1922–1927. https://doi.org/10.1109/ECC.2016.7810572

Lynn, N., & Suganthan, P. N. (2015). Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation. *Swarm and Evolutionary Computation*, *24*, 11–24. https://doi.org/10.1016/J.SWEVO.2015.05.002

Piotrowski, A. P., Napiorkowski, J. J., & Piotrowska, A. E. (2020). Population size in Particle Swarm Optimization. *Swarm and Evolutionary Computation*, *58*(April), 100718. https://doi.org/10.1016/j.swevo.2020.100718

Punyakum, V., Sethanan, K., Nitisiri, K., Pitakaso, R., & Gen, M. (2022). Hybrid differential evolution and particle swarm optimization for multi-visit and multi-period workforce scheduling and routing problems. *Computers and Electronics in Agriculture*, *197*(2), 106929. https://doi.org/10.1016/j.compag.2022.106929

Ramdhani, L. S. (2016). Penerapan Particle Swarm Optimization (PSO) untuk seleksi atribut dalam meningkatkan akurasi prediksi diagnosis penyakit hepatitis dengan metode algoritma C4.5. *Swabumi*, *4*(1), 1–15. https://doi.org/10.31294/swabumi.v4i1.1011

Roberts-Mahoney, H., Means, A. J., & Garrison, M. J. (2016). Netflixing human capital development: personalized learning technology and the corporatization of K-12 education. *Journal of Education Policy*, *31*(4), 405–420. https://doi.org/10.1080/02680939.2015.1132774

Sathyan, A., Boone, N., & Cohen, K. (2015). Comparison of approximate approaches to solving the travelling salesman problem and its application to UAV swarming. *International Journal of Unmanned Systems Engineering*, *3*(1), 1–16. https://doi.org/10.14323/ijuseng.2015.1

Shaj, V., M, A. P., & S, A. (2016). Edge PSO: a recombination operator based PSO algorithm for solving TSP. In *Proceedings of the 2016 Inernational Conference on Advances in Computing, Communication and Informatics* (pp. 35–41). https://doi.org/10.1109/ICACCI.2016.7732022

*TSPLIB*. (n.d.). Retrieved November 3, 2024, from http://www.iwr.uniheidelberg.de/groups/comopt/software/TSPLIB95/tsp/

Vahdat, M., Oneto, L., Anguita, D., Funk, M., & Rauterberg, M. (2016). Can machine learning explain human learning? *Neurocomputing*, *192*, 14–28. https://doi.org/10.1016/j.neucom.2015.11.100

Wang, L., Ni, H., Yang, R., Pardalos, P. M., Du, X., & Fei, M. (2015). An adaptive simplified human learning optimization algorithm. *Information Sciences*, *320*(6), 126–139. https://doi.org/10.1016/J.INS.2015.05.022

Wang, L., Pei, J., Menhas, M. I., Pi, J., Fei, M., & Pardalos, P. M. (2017). A Hybrid-coded Human Learning Optimization for mixed-variable optimization problems. *Knowledge-Based Systems*, *127*(C), 114–125. https://doi.org/10.1016/j.knosys.2017.04.015

Wang, L., Pei, J., Wen, Y., Pi, J., Fei, M., & Pardalos, P. M. (2018). An improved adaptive human learning algorithm for engineering optimization. *Applied Soft Computing Journal*, *71*, 894–904. https://doi.org/10.1016/j.asoc.2018.07.051

Wang, L., Yang, R., Ni, H., Ye, W., Fei, M., & Pardalos, P. M. (2015). A human learning optimization algorithm and its application to multi-dimensional knapsack problems. *Applied Soft Computing Journal*, *34*, 736-743. https://doi.org/10.1016/j.asoc.2015.06.004

Zhong, Y., Lin, J., Wang, L., & Zhang, H. (2018). Discrete comprehensive learning particle swarm optimization algorithm with metropolis acceptance criterion for traveling salesman problem. *Swarm and Evolutionary Computation*, *42*(C), 77–88. https://doi.org/10.1016/j.swevo.2018.02.017