



Article

# An Improved Particle Swarm Optimization Algorithm for the Urban Transit Routing Problem

Vasileios Kourepinis <sup>1</sup>, Christina Iliopoulou <sup>2</sup> , Ioannis X. Tassopoulos <sup>3</sup>, Chrysanthi Aroniadi <sup>3</sup> and Grigorios N. Beligiannis <sup>3,\*</sup> 

<sup>1</sup> School of Science and Technology, Hellenic Open University, 18 Aristotelous St., 26335 Patras, Greece; std115761@ac.eap.gr or v.kourepinis@petrichor.gr

<sup>2</sup> School of Rural, Surveying and Geoinformatics Engineering, National Technical University of Athens, Zografou, 15780 Athens, Greece; christiliop@central.ntua.gr

<sup>3</sup> Department of Food Science & Technology, Agrinio Campus, University of Patras, G. Seferi 2, 30100 Agrinio, Greece; jtass@upatras.gr (I.X.T.); c.aroniadi@upatras.gr (C.A.)

\* Correspondence: gbeligia@upatras.gr

**Abstract:** The Urban Transit Routing Problem (UTRP) is a challenging discrete problem that revolves around designing efficient routes for public transport systems. It falls under the category of NP-hard problems, characterized by its complexity and numerous constraints. Evaluating potential route sets for feasibility is a demanding and time-consuming task, often resulting in the rejection of many solutions. Given its difficulty, metaheuristic methods, such as swarm intelligence algorithms, are considered highly suitable for addressing the UTRP. However, the effectiveness of these methods depends heavily on appropriately adapting them to discrete problems, as well as employing suitable initialization procedures and solution-evaluation methods. In this study, a new variant of the particle swarm optimization algorithm is proposed as an efficient solution approach for the UTRP. We present an improved initialization function and improved modification operators, along with a post-optimization routine to further improve solutions. The algorithm's performance is then compared to the state of the art using Mandl's widely recognized benchmark, a standard for evaluating UTRP solutions. By comparing the generated solutions to published results from 10 studies on Mandl's benchmark network, we demonstrate that the developed algorithm outperforms existing techniques, providing superior outcomes.

**Keywords:** swarm intelligence; population-based optimization; transit network design; particle swarm optimization; UTRP



**Citation:** Kourepinis, V.; Iliopoulou, C.; Tassopoulos, I.X.; Aroniadi, C.; Beligiannis, G.N. An Improved Particle Swarm Optimization Algorithm for the Urban Transit Routing Problem. *Electronics* **2023**, *12*, 3358. <https://doi.org/10.3390/electronics12153358>

Academic Editors: Vasile-Daniel Pavaloaia, Rodrigo Martin-Rojas and Piotr Sulikowski

Received: 21 July 2023

Revised: 3 August 2023

Accepted: 4 August 2023

Published: 6 August 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

The field of transportation has grown rapidly in recent times, presenting significant challenges for engineers and planners alike. The escalating transportation needs of individuals, both domestically and internationally, have made transportation a critical concern in our era. Amidst high pollution levels in urban areas, environmental considerations have become paramount, putting sustainability at the forefront. As a result, researchers are increasingly emphasizing the integration of environmentally friendly measures into transportation models. Such strategies include the adoption of emission-free buses, public bike stations, and vehicle-sharing programs [1]. Transit networks play a crucial role in sustainable transportation systems, garnering considerable attention from the academic community in terms of both planning and operational aspects.

The Urban Transit Routing Problem (UTRP) is a challenging problem in transportation planning that involves designing and optimizing transit route networks for urban areas [2]. The objective is to find the most efficient routes for public transportation vehicles, considering factors such as travel time, passenger demand, transfer connections, vehicle capacities,

operating costs, and environmental impacts. The problem is usually formulated as an optimization problem that seeks to minimize some form of total system costs while satisfying specific service levels and performance criteria. The UTRP is a complex problem due to the large number of potential routes and the various constraints that must be considered. Finding an optimal solution to the UTRP is known to be computationally difficult, with many heuristic and metaheuristic algorithms developed to address the problem [3]. These include techniques such as genetic algorithms (GAs), simulated annealing, and swarm intelligence methods [4]. Efficient and effective public transportation systems are critical for the mobility and sustainability of urban areas, making the UTRP an important area of research in transportation engineering.

Motivated by the promising performance of swarm intelligent algorithms for the UTRP and the need for efficient algorithms to handle the problem, this study develops a new variant of the PSO algorithm to solve the UTRP. We present an improved initialization function and improved modification operators, along with a post-optimization routine to further improve solutions. We compare our method to 10 earlier studies presenting metaheuristics that utilize the same set of assumptions and parameters as our study, allowing for a fair and straightforward comparison of results. Mandl's [5] benchmark instance is used as a basis for comparison. The computational results corroborate the potential of PSO for the UTRP, outperforming other metaheuristics.

The remainder of the paper is organized as follows: Section 2 reviews related literature on the UTRP; Section 3 describes the problem at hand; Section 4 presents the PSO algorithm developed; and Section 5 presents the experimental results. Conclusions and suggestions are given in Section 6.

## 2. Related Work

In the field of transportation, addressing the UTRP has been an ongoing challenge for researchers. Early attempts to tackle the problem relied on heuristic approaches, which proved to be limited in handling large networks and delivering accurate solutions [6]. However, they did lay the groundwork for the development of future methodologies. Analytical methods were then employed, aiming to estimate network structure based on the physical characteristics of transit networks. Yet, as noted by Chakroborty and Wivedi [7], these methods optimized only parameters such as route spacing and length, rather than determining the actual routes themselves. Mathematical programming formulations were also investigated [8,9] but proved inadequate for realistically representing transit routes. Such formulations constructed routes to achieve desirable features, which were reflected by the objective function and constraints rather than determining them directly through the mathematical program.

Early work on metaheuristics included single-solution approaches and Gas. Various single-solution metaheuristics have been proposed, often in combination with other metaheuristics. Simulated annealing frameworks were presented by Zhao and Zeng [10], Fan and Machemehl [11], and Fan and Mumford [3]. Other approaches using Tabu Search were developed by Fan and Machemehl [12], Pacheco et al. [13], and Roca-Riu et al. [14]. Most relevant studies, however, proposed different evolutionary optimization approaches, including GAs, multi-objective evolutionary algorithms, and memetic algorithms for solving the UTRP. Among the early such studies, Chakroborty and Wivedi [7] presented a novel stochastic initialization procedure and a multi-criteria evaluation method using a GA, while Chew and Lee [15] developed an efficient GA solution scheme. Nayeem et al. [16] used a GA featuring the concept of elitism and an increasing population approach. More recently, Jha et al. [17] used a GA for determining routes combined with a multi-objective PSO framework for generating corresponding frequencies. Additionally, differential evolution strategies and memetic evolutionary algorithms have also been proposed by Buba and Lee [18,19], Zhao et al. [20], and Duran-Micco et al. [21].

In a departure from earlier GAs and single-solution metaheuristics, swarm intelligence methods, such as Bee Colony Optimization (BCO), PSO, and Ant Colony Algorithms

(ACOs), have been receiving growing attention for solving the UTRP, showcasing their potential in addressing the problem. Early efforts using swarm intelligence for transit network optimization were based on ACO. Hu et al. [22] proposed two methods to maximize passenger flow and optimize headways, while Yu and Yang [23] presented an iterative approach to obtain a Pareto-optimal solution for bus network design and transit assignment. Yang et al. [24] proposed a coarse-grain parallel ant colony algorithm to optimize a bus network, while Blum and Mathew [25] presented an intelligent agent system based on ant colonies to determine routes and frequencies for a given transit network. Yu et al. [26] extended the direct traveler density model to maximize transit trip density with respect to total demand and route length using ACO. A subsequent stream of studies investigated the potential of bee colonies for the UTRP. In this line of work, Szeto and Jiang [27] and Jiang et al. [28] used a hybrid enhanced artificial bee colony algorithm to determine the route structure for a bus network. Nikolić and Teodorović [29] proposed a BCO model for the UTRP, which outperformed other metaheuristics on Mandl's network. They later extended their work [30] to simultaneously determine both routes and frequencies.

In terms of PSO, the standard version of the algorithm is well-suited for continuous optimization problems since it employs equations to calculate the velocity and position of any individual. However, for discrete decision variables such as those in the UTRP, appropriate modifications must be made to represent the search process. In this respect, Kechagiopoulos and Beligiannis [31] developed a discrete version of the PSO algorithm, demonstrating competitive performance compared to other metaheuristics. In a later effort, Gunby and Gustavsen [32] proposed a hybrid swarm intelligence method that combines ACO with additional attributes inspired by BCO and PSO, resulting in better performance than the basic ACO implementation. Recently, Katsaragakis et al. [33] introduced a modified version of the Cat Swarm Optimization (CSO) algorithm for the UTRP that achieved better results than previous implementations.

For further reading, the interested reader is referred to the thorough reviews of related methodological developments provided in [4,6,34,35]. This study further explores the potential of swarm intelligence for the problem and presents a new variant of the PSO algorithm for the UTRP.

### 3. Problem Description

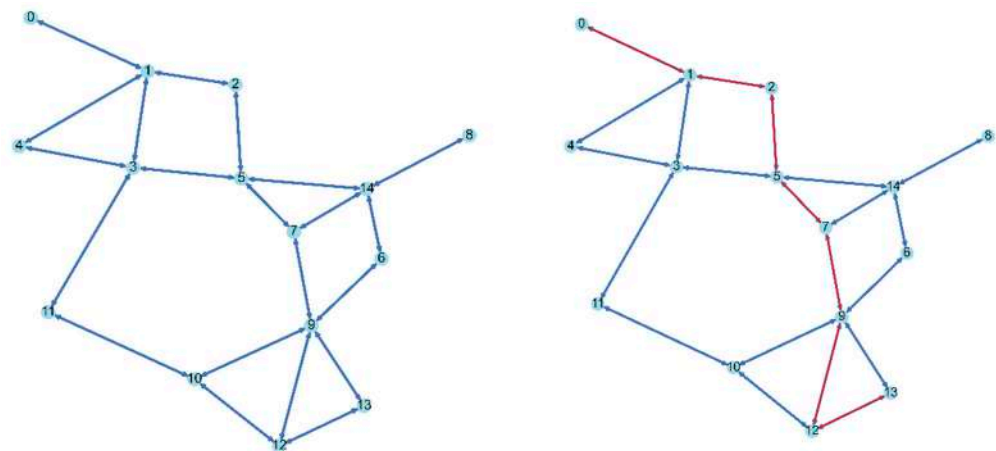
#### 3.1. The UTRP Problem

The planning process for any transit operator involves the following five steps, in this order: (1) route design, (2) frequency determination, (3) timetabling, (4) vehicle scheduling, and (5) the crew scheduling and rostering [34]. The UTRP consists of the construction of routes for a fleet of vehicles subject to operational and other constraints. The problem belongs to a broader class of optimization problems known as the Vehicle Routing Problem (VRP), which has been at the core of designing routing operations for decades; a set of routes is determined so that several customers are served by a fleet of vehicles at a minimum cost. Similarly, the UTRP aims to satisfactorily meet passenger demand while minimizing travel costs, comprising a multi-objective problem.

The main focus of the UTRP is to optimize the efficiency of public transportation networks while considering operational and resource constraints, such as the number and length of routes, allowable service frequencies, and available buses. The transport network is represented as an undirected graph  $G = (N, A)$  with nodes,  $N$ , representing access points and edges,  $A$ , representing direct transport links. A route  $R$  is represented by a path in the transport network, and the UTRP's solution is specified by a route set  $RS$ . The route network associated with a route set is defined as a subgraph of the transport network that contains only those edges that appear in at least one route of the route set.

To solve the UTRP, the optimal value of the objective function must be determined given a road network, a transit demand matrix, and a set of constraints. Various design objectives have been proposed in the literature, including minimizing user cost (such as average travel time, average waiting time, and number of transfers), minimizing operator

cost (such as number of routes, fleet size, length of routes, and hours of operation), or both [35]. Figure 1 shows a road network and a transit route on this network.



**Figure 1.** A transportation network and an example route (in red). Nodes (access points) of the network are named using different numbers.

One of the inherent difficulties of designing a transit network is the need to offer a desirable level of service for passengers, while ensuring a profitable operation for the transit agency. From the passengers' standpoint, the service should be fast, offering quicker travel compared to other available transportation modes. Additionally, the service should primarily facilitate direct trips, allowing, at most, one, or for longer trips, a maximum of two, transfers between vehicles, which is generally acceptable to passengers. Efficient route sets are characterized by low unsatisfied demand, a high percentage of demand satisfied through direct trips, and a low average travel time per user. On the other hand, the transit company aims to minimize its operational cost while meeting the passengers' standards. Operational costs usually depend on the number of routes, the fleet size, the length of the routes, and the hours of operation. Real-world constraints must also be considered, such as the length and number of routes offered, to incorporate practical considerations such as scheduling and shift duration. Finally, routes cannot have cycles, and the road network must be a connected graph, i.e., offer the possibility to all passengers to travel between their origin and destination.

### 3.2. Problem Difficulty

The complexity of the UTRP stems from several factors that contribute to its intricacy. Firstly, the discrete nature of the decision variables involved in the problem is a primary source of complexity. These decision variables refer to the routes and their constituent links, and even in a small network, the sheer number of these variables significantly amplifies the computational cost [8]. Additionally, the presence of logical conditions, such as route connectivity and cycle existence, further burdens the computational process as they necessitate modeling and verification.

In terms of complexity, the UTRP problem is known to be NP-hard in its general form, implying that finding an acceptable solution within a polynomial time frame using a deterministic algorithm is infeasible [36]. Another noteworthy characteristic of the problem is the lack of a universally adopted solution method in the existing literature, despite the abundance of available approaches. Newell [36] highlighted the difficulty of designing a route structure for a realistic-sized network as a non-convex optimization problem, requiring a direct comparison of multiple local optima. The author emphasized that the problem's discrete, NP-hard, and combinatorial nature makes it challenging to obtain a solution through traditional optimization techniques, primarily due to the intricacies involved in calculating the objective function.

This complexity is one of the key reasons why conventional methods are deemed inefficient for addressing the UTRP, as external procedures are necessary to compute input quantities for the optimization process [37]. Indeed, evaluating the quality of a given route set necessitates a complex procedure, as passengers must be assigned to routes, requiring the evaluation of all routes within the final route set as an integrated entity. Chakroborty [37] attributes the failure of traditional solution approaches to the representation mechanisms employed in mathematical programming formulations, which struggle to incorporate concepts such as transfers and route continuity.

Another notable characteristic of the UTRP problem is the presence of numerous conflicting objectives that need to be fulfilled, as previously explained. This inherently gives rise to a multi-objective problem, wherein multiple targets compete with one another. These competing objectives include reducing service costs, maximizing the percentage of passengers benefiting from direct service, and minimizing average travel time, among others.

#### 4. Solution Algorithm

Motivated by the promising performance of the discrete PSO on the UTRP for Mandl's benchmark [1,6,31], we seek to further improve upon the former, by introducing appropriate modifications. The proposed PSO algorithm was developed in Java. A comprehensive explanation of the algorithm's elements can be found in the subsequent sub-sections. The initial solution generation, the route-set-modification procedure, the objective function, and the termination criterion are addressed next.

##### 4.1. Solution Representation

A key element in solving the problem is the representation of the solution. Herein, we chose a popular solution representation based on matrices [37,38]. Table 1 shows a solution with a set of four routes.

**Table 1.** Solution representation.

Routes								
1	11	1	3	10	12	13	9	12
2	7	9	12	5	0	1	2	10
3	14	8	7	6	13	12		
4	9	6	14	5	1	4	3	1

##### 4.2. Initialization Process

Although optimization algorithms can converge regardless of the quality of initial solutions, employing a set of "good" initial solutions can expedite the convergence process. Various approaches have been proposed to generate an efficient initial route set for the UTRP. One such approach, presented by Chakroborty et al. [39], involves probabilistically selecting the first node of a path based on its activity level, giving higher-activity nodes a greater likelihood of selection. Similarly, subsequent nodes are chosen using the same methodology based on adjacency. Kechagiopoulos and Beligiannis [31] modified this process by incorporating the Make-Small-Change procedure developed by Fan et al. [2] and Fan and Mumford [3], which ensures that each route reaches the maximum allowable number of nodes. Conversely, Fan et al. [2] argued that employing a non-random method for individual route generation does not enhance algorithmic convergence, as the best quality of individual routes does not guarantee the best quality of the route set.

The initialization process proposed herein extends the one used by Fan et al. [2]. Modifications entail the use of a random variable to introduce randomness in the route length constraint, as well as changes in the process of selecting the first node of each route and the remaining ones. We also use a priority heap to select the best  $k$  initial solutions. The steps of the method are as follows:

1. The desired length of a route is given by an integer number within the acceptable range, as defined by the problem inputs. The value of this number depends on the random variable; if its value is set to true, a number within the feasible range is chosen at random; otherwise the maximum value is adopted. This process is repeated for all routes in the set.
2. Selection of the first node: The selection is performed randomly for the first node of the first route. Otherwise, a node is chosen at random from the nodes already present in the previous routes to increase connections between routes.
3. Selection of successive nodes: Successive nodes are selected from the adjacency matrices of their predecessors on the condition that they are not already part of the route. If such a node does not exist, then the route is reversed and the process is repeated. If extending the route is still not possible, these two steps are repeated, and the process randomly selects a node out of all possible nodes and attempts to add it to the route. If this addition leads to a cycle in the route, it is abandoned and the process starts from step 2.
4. In the case of successful route creation with the length specified in step 1, the route is added to the route set and the process continues until the maximum number of routes are created.

#### 4.2.1. Route Modification Procedure to Achieve Feasibility

The above procedure produces route sets without circular paths, yet it does not ensure that the resulting network is connected or that all nodes of the initial network are included. To ensure the satisfaction of these constraints, we use the Feasibility Check and Make-Small-Change methods by Fan et al. [2]. The former method checks whether all nodes are included in a route set. Subsequently, the Make-Small-Change method modifies one route at a time by either adding or removing a node. The method randomly selects a route from the route set and modifies it depending on the length of the route, with three possible cases discerned (minimum, maximum route length, or in-between).

#### 4.2.2. Swarm Creation Process

The initial generation of solutions using the methods mentioned above creates the specified number of total solutions. The steps of the process are as follows:

1. The initialization method is first executed to generate a set of routes with the value of the random parameter selected with 50% probability, so that 50% of routes have the maximum route length.
2. The feasibility of the route set generated in the previous step is checked using the Feasibility Check method. If the route set is feasible, it is retained. Otherwise, the Make-Small-Change procedure is executed until the route set becomes feasible or until a predefined number of iterations is exhausted. If no feasible set of routes is obtained, the process is repeated from step 1. Note that duplicate route sets are discarded, thus ensuring a greater diversity of solutions.
3. Once a predetermined number of iterations has been completed and if the minimum number of unique solutions has been generated, the solution-generation process is terminated.
4. The desired number of solutions is retained based on their objective function value.

The purpose of Step 1 is to prioritize solutions featuring routes that have the maximum number of nodes, since these are usually better, while also including solutions with shorter routes, which nonetheless, may be of good quality. Finally, Step 4, although not necessary, significantly improves the convergence speed, since it initializes the algorithm with a higher-quality and more diverse solution set without duplicate solutions (Step 2).

#### 4.3. Route Set Evaluation

The evaluation of solutions in the UTRP is inherently more intricate compared to continuous mathematical problems. This assessment relies on multiple criteria, each carrying

varying degrees of significance, which may also reflect different planning perspectives. Indeed, a route set must be evaluated with respect to the level of service offered to passengers. The latter is captured through the following five indicators, in line with Chakroorty [37], Kechagiopoulos and Beligiannis [31], and Fan et al. [2]:

- Average travel time (*ATT*): This criterion measures the average time required for passengers to travel from their origin node to their destination node.
- Direct route percentage ( $d_0$ ): This criterion assesses the percentage of passengers who can reach their destination by choosing the shortest route without any transfers.
- One-transfer route percentage ( $d_1$ ): This criterion quantifies the percentage of passengers who can reach their destination by taking the shortest route with a single transfer.
- Two-transfer route percentage ( $d_2$ ): This criterion measures the percentage of passengers who can reach their destination by selecting the shortest route with two transfers.
- Unsatisfied demand ( $d_{un}$ ): This criterion evaluates the percentage of passengers who can reach their destination by utilizing the shortest route with more than two transfers.

These criteria are used to compute the following objective function:

$$FIT(r) = \omega_1 \cdot F_1(r) + \omega_2 \cdot F_2(r) + \omega_3 \cdot F_3(r) \tag{1}$$

where

$r$ : set of routes to evaluate;

$F_1(r)$ : The component associated with *ATT*;

$F_2(r)$ : The component associated with  $d_0$ ,  $d_1$  and  $d_2$ ;

$F_3(r)$ : The component associated with  $d_{un}$ ;

$\omega_1, \omega_2, \omega_3$ : user-specified weights.

The goal in this case is to maximize the objective function (1). The computation of each of these components is addressed next.

#### 4.3.1. $F_1(r)$ Calculation

The value of  $F_1(r)$  reflects the average detour cost per passenger, increasing as the average travel time decreases. To compute  $F_1(r)$ , for each pair of nodes ( $i, j$ ) of a set of routes, the absolute minimum travel time between nodes  $i$  and  $j$ ,  $T_{i,j}^{min}$ , is calculated, based solely on the road network, without considering the routes or possible transfer delays. Subsequently, the corresponding in-vehicle travel time  $IVT_{i,j}(r)$  for a passenger travelling between the corresponding nodes using the route network is computed. The computation of average travel time per passenger requires certain assumptions regarding passenger behavior. Therefore, it is assumed that passengers possess complete information about the transportation network and plan their route(s) ahead of time. To ensure comparability with other studies, the travel time of each passenger is increased by 5 min for each transfer required. The method assumes that the burden of transferring is considered beforehand by passengers, enabling them to avoid transfers that lead to longer travel times, even if the alternative path is not the shortest based on the road network [3].

Finally,  $T_{i,j}^{min}$  is compared to the corresponding  $IVT_{i,j}(r)$  to calculate indicator  $f_{i,j}(r)$ , which accounts for the divergence from the shortest path:

$$f_{i,j}(r) = \begin{cases} \beta_1 x + K_1, & 0 \leq x \leq x_m \\ 0, & x > x_m \end{cases} \tag{2}$$

where  $x = IVT_{i,j}(r) - T_{i,j}^{min}$ ,

$x_m$ : Largest acceptable value of  $x$ ;

$K_1$ : A positive user-defined constant which indicates the upper bound of  $f_{i,j}(r)$ ;

$\beta_1$ : A constant in the range  $-K_1/x_m \leq \beta_1 \leq 0$ .

Finally,  $F_1(r)$  is calculated as follows:

$$F_1(r) = \frac{\sum_{\forall(i,j) \in S(r)} d_{i,j} f_{i,j}}{\sum_{\forall(i,j) \in S(r)} d_{i,j}} \quad (3)$$

where

$S(r)$ : The set of origin-destination node pairs  $(i, j)$  for which passengers  $d_{i,j}$  are served by the route set  $r$ .

#### 4.3.2. $F_2(r)$ Calculation

The second component  $F_2(r)$  is computed using the share of total demand traveling with zero, one or two transfers, respectively. It is computed as follows:

$$F_2(r) = \beta_2 d_T(r) \quad (4)$$

where

$K_2$ : A user-defined positive constant denoting the upper bound of  $F_2(r)$ ;

$\beta_2$ : A constant in the range  $K_2/\alpha_2 \leq \beta_2 \leq 2K_2/\alpha_2$ ;

$\alpha_2$ : The upper bound of  $d_T(r)$ ;

$d_T(r)$ : The weighted sum of transfer shares, calculated as follows:

$$d_T(r) = a \cdot d_0(r) + b \cdot d_1(r) + c \cdot d_2(r) \quad (5)$$

$d_0(r)$ : Share of demand satisfied with no transfers using route set  $r$ ;

$d_1(r)$ : Share of demand satisfied with one transfer using route set  $r$ ;

$d_2(r)$ : Share of demand satisfied with two transfers using route set  $r$ ;

$a, b, c$ : User-defined constants which attribute importance to each component, with  $a \geq b \geq c$ .

#### 4.3.3. $F_3(r)$ Calculation

The third component  $F_3(r)$  penalizes unsatisfied demand and is calculated as follows:

$$F_3(r) = \beta_3 d_{un}(r) + K_3 \quad (6)$$

where

$K_3$ : A user-defined positive constant denoting the upper bound of  $F_3(r)$ ;

$\beta_3$ : A constant in the range  $-K_3 \leq \beta_3 \leq 0$ ;

$d_{un}(r)$ : The share of unsatisfied passengers, i.e., those that cannot reach their destination with two or fewer transfers.

#### 4.4. Computational Cost Savings Using Dynamic Programming

At each iteration, multiple solution evaluations are conducted, adding to the complexity of the algorithm. To avoid repeating this computationally demanding process for solutions that have already been explored, we store the objective function value, as well as the evaluation results ( $d_0, d_1, d_2, d_{un}, ATT$ ) for each set of routes encountered during the search in a Hash table, using the set of paths as the key and the evaluation results as the corresponding value. For the implementation of the table, the Java data structure, HashMap, which has a data-retrieval speed of  $O(1)$ , was used. Hence, during the evaluation of a solution, a check is performed to determine if its corresponding results already exist in the HashMap. If they do, the results are fetched from the Hash table rather than recomputing them. Indicatively, it is reported that evaluation results using the array were retrieved over 90% of the time. The percentage coincides with the computational cost savings, as it is these calculations that influence the total execution time.



#### 4.5. Modification Procedure

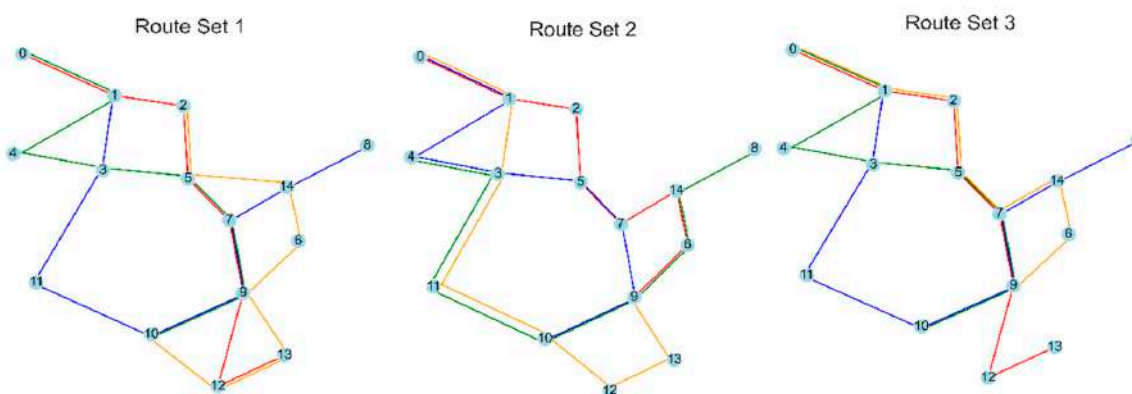
Solutions are modified by either replacing or introducing modifications to individual routes, similar to [7]. The corresponding procedures are described next.

##### 4.5.1. Entire Route Replacement

Throughout the entire process of replacing routes, two sets of routes are initially selected: the first set includes the route that will be replaced, while the second set includes the route that will serve as the replacement. The steps are as follows:

1. A route from the first set of routes and a route from the second set of routes are randomly selected.
2. The first route is replaced by the second route.
3. The feasibility of the route set is checked using the Feasibility Check procedure. If the new set of routes is feasible, then the procedure terminates by returning true.
4. If no feasible set of routes is obtained, then all possible replacements from the second route set are investigated until a feasible solution is obtained. If the routes of the second set are exhausted, then the procedure is repeated from the first step, excluding already-explored routes. The procedure terminates by returning “true” if a feasible set of routes is obtained and “false” otherwise.

In the final step of the algorithm, a modified version of the aforementioned method is employed. Instead of terminating upon finding a feasible set of routes, this variant explores all possible combinations of swaps and stores the outcomes in a priority queue. By doing so, the process retains the best-performing swap as the final result. Figure 2 provides an illustrative example of the procedure’s implementation, demonstrating the replacement of the yellow route from the first route set with the red route from the second set, resulting in the formation of route set 3.



**Figure 2.** Entire route replacement. Each route is presented with a differently-coloured line.

##### 4.5.2. Partial Route Modification

This process entails introducing modifications to the current particles by incorporating route segments from other members of the population. Similar to the approach explained earlier, two sets of routes are initially chosen: the first set encompasses the segment that will be replaced, while the second set contains the route part that will serve as the replacement. The procedure for this process is outlined below:

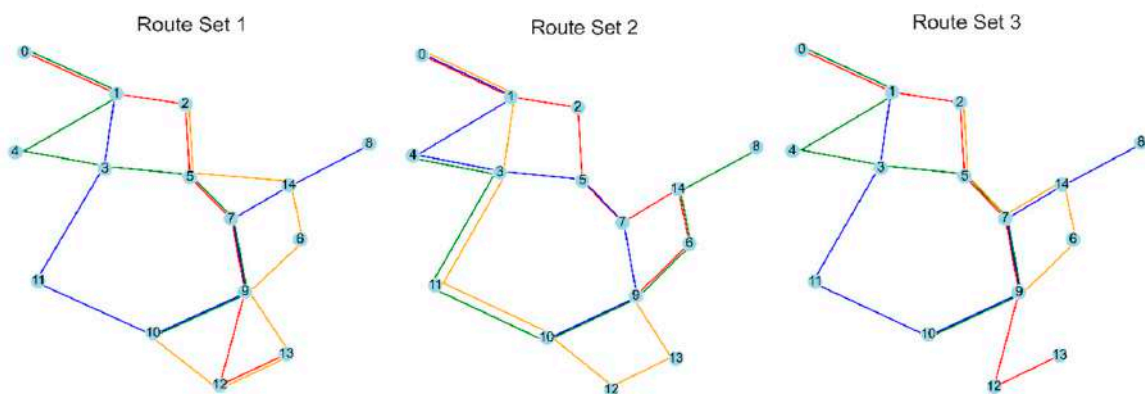
1. One route from the first solution and one route from the second are randomly selected.
2. The process searches for common nodes between the two routes. If there are multiple shared nodes, one of them is randomly selected. If no common nodes exist, then another route from the second route set is selected. If no common nodes are detected, a new route is chosen from the first set of routes, and the procedure is repeated until a common node is identified.

3. The route segment following the shared node is replaced in the first route, on the condition that a circular path is not created. All possible combinations are tested in random order, i.e.,
  - (i) Case 1: The two routes in their initial state;
  - (ii) Case 2: Reversal of the first route;
  - (iii) Case 3: Reversal of the second route;
  - (iv) Case 4: Reversal of both routes.

Furthermore, it is verified that the number of nodes falls within the predefined minimum and maximum limits. If this criterion is not satisfied, another route selection is executed, as described in the second step.

4. The feasibility of the route is checked using the Feasibility Check procedure. If the new set of routes is feasible, the procedure terminates. If no feasible set of routes is found, another route is selected, as described in Step 2.

In the last step of the algorithm, a variant of the method is used, where all possible combinations are considered and only the best result is retained. An example implementation of the procedure is shown in Figure 3. The swap is performed between the yellow route of set 1 and set 2, at the common node 5, where the part after node 5 in the former (5-14-6-9-13-12-10) is replaced by the part after node 5 in the latter (5-7-14-6-9).



**Figure 3.** Partial route replacement. Each route is presented with a differently-coloured line.

#### 4.6. Description of Proposed PSO Algorithm

The PSO algorithm is a stochastic optimization technique based on the behavior of swarms proposed by Eberhart and Kennedy [40]. It simulates the social behavior of various animal groups, such as insect colonies, animal herds, fish schools, and bird flocks. All these living organisms collaborate with the aim of finding food, and each member of the swarm adjusts its behavior based on the experiences gained from itself or other members of the swarm. The design of the PSO algorithm is built upon two main ideas. The first idea draws inspiration from evolutionary algorithms, which utilize a swarm to explore a large region of the solution space in order to optimize a given objective function. The second idea is derived from the behavior of living organisms and how they employ collective intelligence and communication behaviors among themselves to organize and perform functions such as searching for food and protecting themselves from predators.

The PSO algorithm involves creating an initial swarm of particles, where each particle represents a potential solution. The selection of initial solutions is random and does not affect convergence. Each particle has a position vector and a velocity vector, which evolve over discrete time steps corresponding to algorithm iterations. At each time step, a particle stores its position, previous velocity, and personal best position (*pbest*), with the best objective function value achieved so far and the global best position (*gbest*) among all particles in the swarm. The particle's position corresponds to a solution, and its velocity determines its next position in the solution space. The cognitive component is calculated

based on the particle’s previous velocity and *pbest* position, while the social component is calculated using the velocity and *gbest* position. After particle movement, *gbest* and *pbest* are updated based on the new positions of the particle itself and other particles in the swarm. This iterative process continues until a convergence criterion is met.

This implementation simulates the movement towards the personal and the global best, while the velocity component was not implemented. The reason for this is that the velocity component does not hold much significance in this particular problem. The problem is discrete, and the nature of its solutions is such that it cannot be assumed that a particle has a fixed direction towards a specific point. The following procedures comprise the PSO algorithm.

- **Best\_of\_Self:** The particle moves towards the best solution it has found so far for itself, approaching the *pbest* particle.
- **Best\_of\_Swarm:** The particle moves towards the best value achieved thus far, which is the global optimum, thus approximating the *gbest* particle, which represents the global maximum obtained so far. In the last iteration, this process is also performed using *routeSwapBest* and *routeSegmentSwapBest* and, in that case, is called *Best\_of\_Swarm\_all*.

The algorithm’s flowchart is depicted in Figure 4, highlighting two distinct loops: the outer loop, which runs for a maximum of *max\_iter* iterations, and the inner loop, which iterates *max\_particle* times, equivalent to the fish population size. During the final iteration, the exhaustive procedures *Best\_of\_Swarm\_all* and *Best\_of\_Self\_all* are executed to further refine the solutions. This step ensures convergence with a reduced number of iterations, and, based on the conducted tests, it has been observed to enhance the quality of solutions.

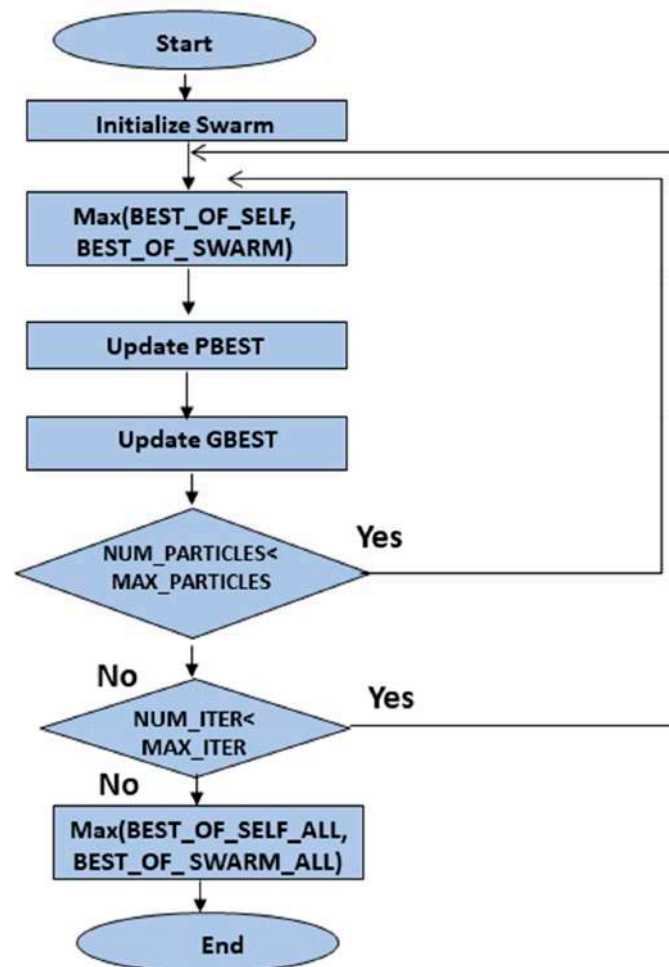


Figure 4. Flowchart of the proposed PSO.

### 5. Computational Results

#### 5.1. Benchmark Network and Parameter Settings

The proposed PSO algorithm utilizes a road network as input, which was initially introduced by Mandl [5] and is based on an actual Swiss road network. This particular road network has been extensively employed for evaluating optimization approaches and is widely recognized as the standard benchmark for the UTRP problem. Mandl’s network consists of 15 nodes and 21 links, with a total of 15,570 passenger trips. The demand matrix exhibits symmetry, and the routes operate in both directions. In line with customary practices in the literature, our investigation focuses on route configurations involving four, six, seven, and eight routes for this specific network. All experiments were performed on a 3.7 GHz Intel® i7-8700 k with 32 GB DDR3 3000 MHz memory.

The following functions and parameter values were adopted. In term  $F_1(r)$ ,  $x_m = 20$  min,  $K_1 = 10$ , and  $\beta_1 = -0.5$ . Further, in term  $F_2(r)$ ,  $\beta_2 = 10$ ,  $a = 0.8$ ,  $b = 0.15$ ,  $c = 0.05$ . Last, in term  $F_3(r)$ ,  $K_3 = 10$ , and  $\beta_3 = -1$ . After experimentation, the objective function coefficients were set as  $\omega_2 = 1.0$ ,  $\omega_3 = 1.0$ , with  $\omega_1 = 2.0$ . Moreover, a population of 200 solutions was used, with 100 iterations for the four-route case and 150 for the six-, seven-, and eight-route cases.

#### 5.2. Network Configurations and Comparison to State-of-the-Art

This section presents computational results and resulting network configurations for all route-number scenarios investigated. Table 2 shows the best solution resulting from PSO, while the corresponding routes are also shown in Figure 5.

Table 2. Best PSO routes for the 4-route case.

Route 1:	0	1	2	5	7	14	6	9
Route 2:	0	1	4	3	5	7	9	10
Route 3:	8	14	6	9	10	11	3	4
Route 4:	0	1	3	11	10	12	13	9

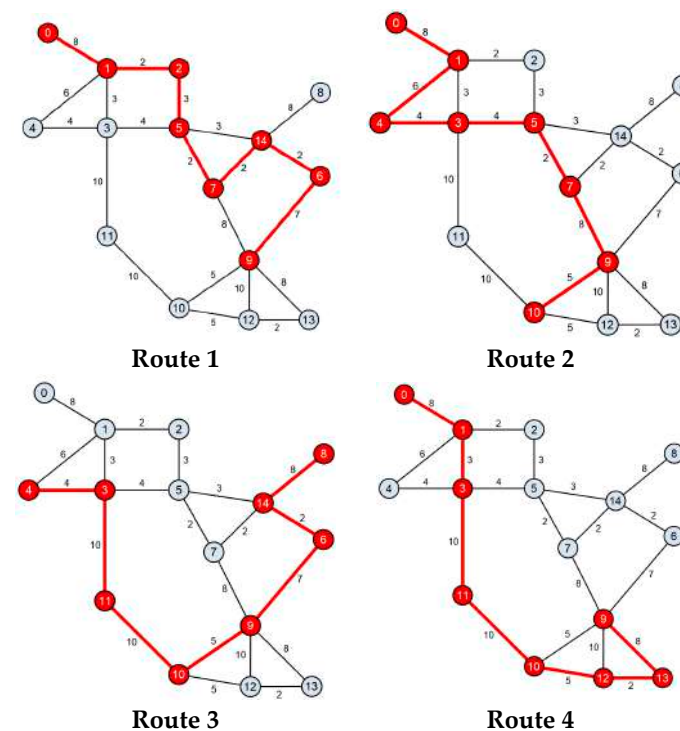


Figure 5. Routes obtained for the 4-route case.

In Table 3, we present a comparison with previous studies. To assess the consistency of the algorithm, we conducted 40 experiments and calculated the mean and standard deviation. In addition to presenting the solution with the best objective function value, we also include the solution with the lowest average trip time (*ATT*) obtained, as we deem it to be a solution of exceptional quality.

**Table 3.** Comparison between methods for 4-route case.

Study	[5]	[41]	[37]	[2]	[15]	[31]	[29]	[32]	[17]	[33]	PSO Best	PSO Best <i>ATT</i>	PSO Mean	PSO Std
<i>d</i> <sub>0</sub> (%)	69.94	72.95	86.86	93.26	93.7	91.84	88.76	89.15	85.28	91.52	<b>94.41</b>	<b>93.51</b>	94.2	0.49
<i>d</i> <sub>1</sub> (%)	29.93	26.91	12	6.74	6.29	7.64	10.15	9.76	14.72	7.77	<b>5.27</b>	<b>6.1</b>	5.48	0.5
<i>d</i> <sub>2</sub> (%)	0.13	0.13	1.14	0	0	0.51	1.09	1.09	0	0.71	0.32	0.39	0.31	0.11
<i>d</i> <sub>un</sub> (%)	0	0	0	0	0	0	0	0	0	0	0	0	0	0
<i>ATT</i> (mpu)	12.9	12.72	11.9	11.37	10.82	10.64	10.79	10.64	10.9	10.54	10.56	<b>10.53</b>	10.56	0.02

The results displayed in Table 3 reveal that PSO outperforms other population-based algorithms, encompassing various GA implementations and swarm intelligence methods. Particularly noteworthy is its ability to achieve a remarkably high percentage of direct demand coverage, reaching nearly 94.5%, while maintaining a significantly low average trip time. The mean execution time for PSO was 27 s.

Next, we present the results obtained for a six-route network. Table 4 and Figure 6 illustrate the best solution achieved for this particular case.

**Table 4.** Best PSO routes for the 6-route case.

Route 1:	4	3	1	2	5	7	14	6
Route 2:	0	1	4	3	5	7	9	13
Route 3:	0	1	3	5	14	6	9	12
Route 4:	8	14	6	9	10	11	3	4
Route 5:	13	12	10	11	3	5	14	8
Route 6:	0	1	2	5	7	9	10	12

In Table 5, we provide a comparison with previous studies. The results demonstrate that the solution generated using PSO exhibits a lower average travel time and a higher percentage of direct demand coverage compared to other solutions, as indicated in the table. The solution with the lowest *ATT* is shown as well.

**Table 5.** Comparison between methods for the 6-route case.

Study	[38]	[41]	[37]	[2]	[15]	[31]	[29]	[32]	[17]	[33]	PSO Best	PSO Best <i>ATT</i>	PSO Mean	PSO Std
<i>d</i> <sub>0</sub> (%)	78.61	77.92	82.34	91.52	95.57	95.63	96.21	88.57	92.61	96.21	<b>97.94</b>	<b>97.81</b>	98.48	0.23
<i>d</i> <sub>1</sub> (%)	21.39	19.62	15.86	8.48	4.43	4.37	3.66	10.34	7.39	3.66	<b>2.06</b>	<b>2.18</b>	1.52	0.23
<i>d</i> <sub>2</sub> (%)	0	2.4	1.8	0	0	0	0.13	1.09	0	0.13	0	0	0	0
<i>d</i> <sub>un</sub> (%)	0	0	0	0	0	0	0	0	0	0	0	0	0	0
<i>ATT</i> (mpu)	11.86	11.87	11.41	10.48	10.28	10.23	10.22	10.68	10.4	10.22	<b>10.19</b>	<b>10.18</b>	10.29	0.04

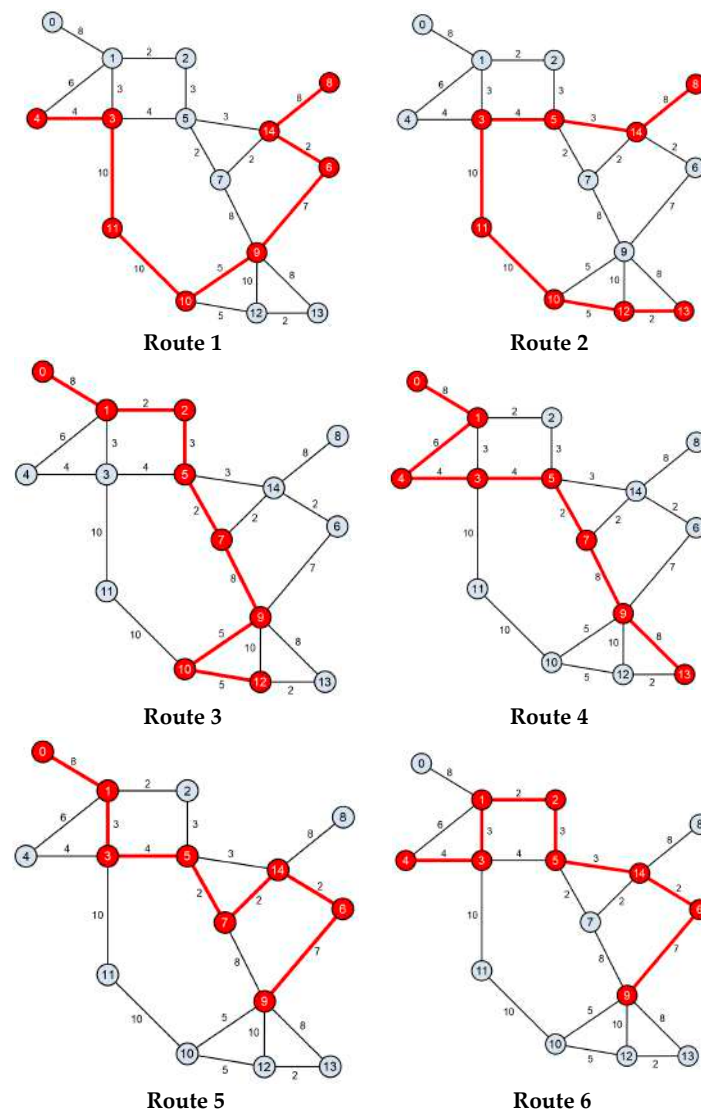


Figure 6. Routes obtained for the 6-route case.

It is worth noting that for this benchmark, the share of direct trips reaches approximately 98%, surpassing the highest value achieved by other methods, which was 96.21%. In this particular case, with a population size of 200 and 150 iterations, the average computational time was 437 s.

We next present the results obtained for a seven-route network. Table 6 displays the best solution achieved for this case, while Table 7 provides a comparison with earlier studies. Additionally, Figure 7 showcases the best route set obtained.

Table 6. Best PSO routes for the 7-route case.

Route 1:	13	9	7	5	3	4	1	2
Route 2:	4	3	11	10	9	6	14	8
Route 3:	11	3	1	2	5	7	14	8
Route 4:	0	1	2	5	14	6	9	13
Route 5:	9	13	12	10	11	3	1	0
Route 6:	6	14	7	5	3	4	1	0
Route 7:	0	1	2	5	7	9	10	12

Table 7. Comparison between methods for 7-route case.

Study	[38]	[41]	[37]	[2]	[15]	[31]	[29]	[32]	[17]	[33]	PSO Best	PSO Mean	PSO Std
$d_0$ (%)	80.99	93.91	89.15	93.32	95.57	98.52	97.17	89.98	93.83	97.94	<b>99.29</b>	99.19	0.09
$d_1$ (%)	19.01	6.09	10.85	6.36	4.43	1.48	2.83	8.54	6.17	2.06	<b>0.71</b>	0.81	0.09
$d_2$ (%)	0	0	0	0.32	0	0	0	1.48	0	0	0	0	0
$d_{um}$ (%)	0	0	0	0	0	0	0	0	0	0	0	0	0
ATT (mpu)	12.5	10.7	10.15	10.42	10.27	10.13	10.16	10.61	10.17	10.12	<b>10.1</b>	10.11	0.01

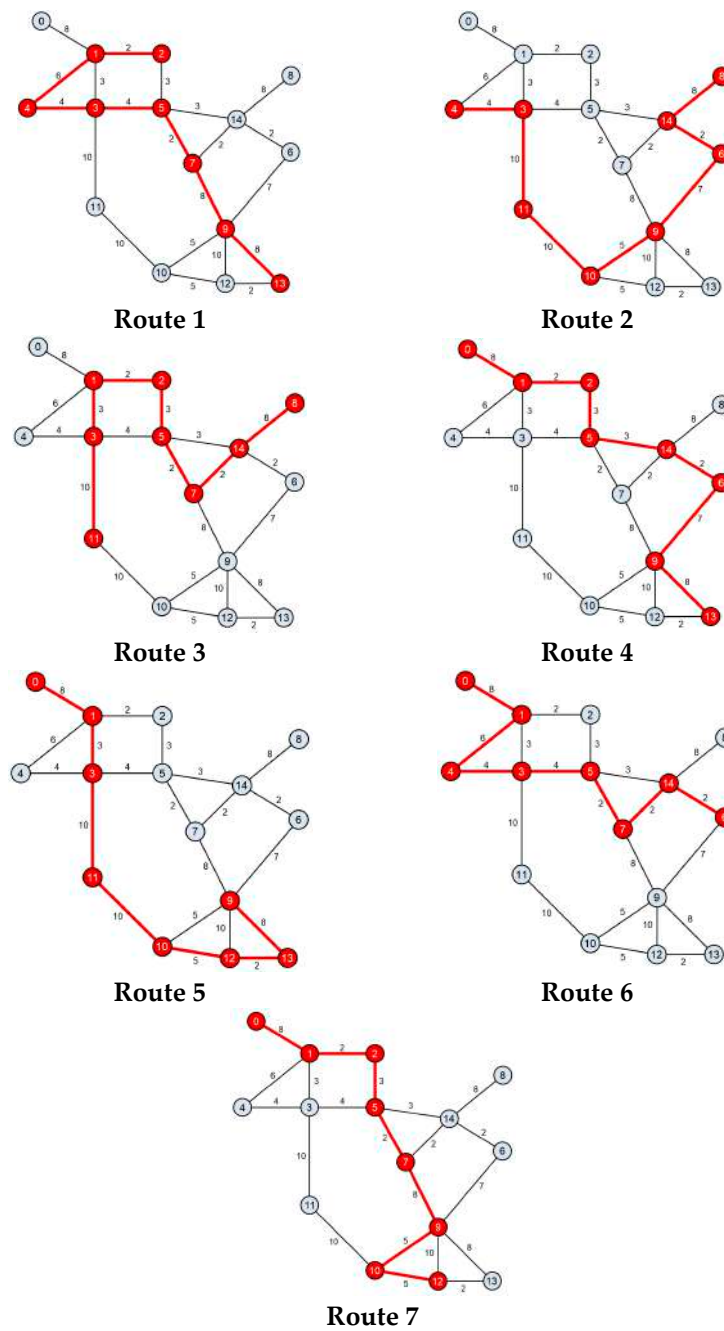


Figure 7. Route set obtained for 7-route case.

As depicted in Table 7, the solution generated by PSO exhibits a lower average travel time and a higher percentage of direct demand compared to existing solutions.

In this case, as well, it is noteworthy that the share of direct trips is significantly higher compared to existing approaches, reaching 99.29%. Importantly, no passenger has to transfer twice in the obtained solution. The average run time for 150 iterations in this case was 854 s.

Moving on, we also investigated route network configurations with eight routes. The results for the transit network with eight routes are presented in Table 8, while Figure 8 displays the routes. Interestingly, in this particular case, not all routes reach the maximum node number per route.

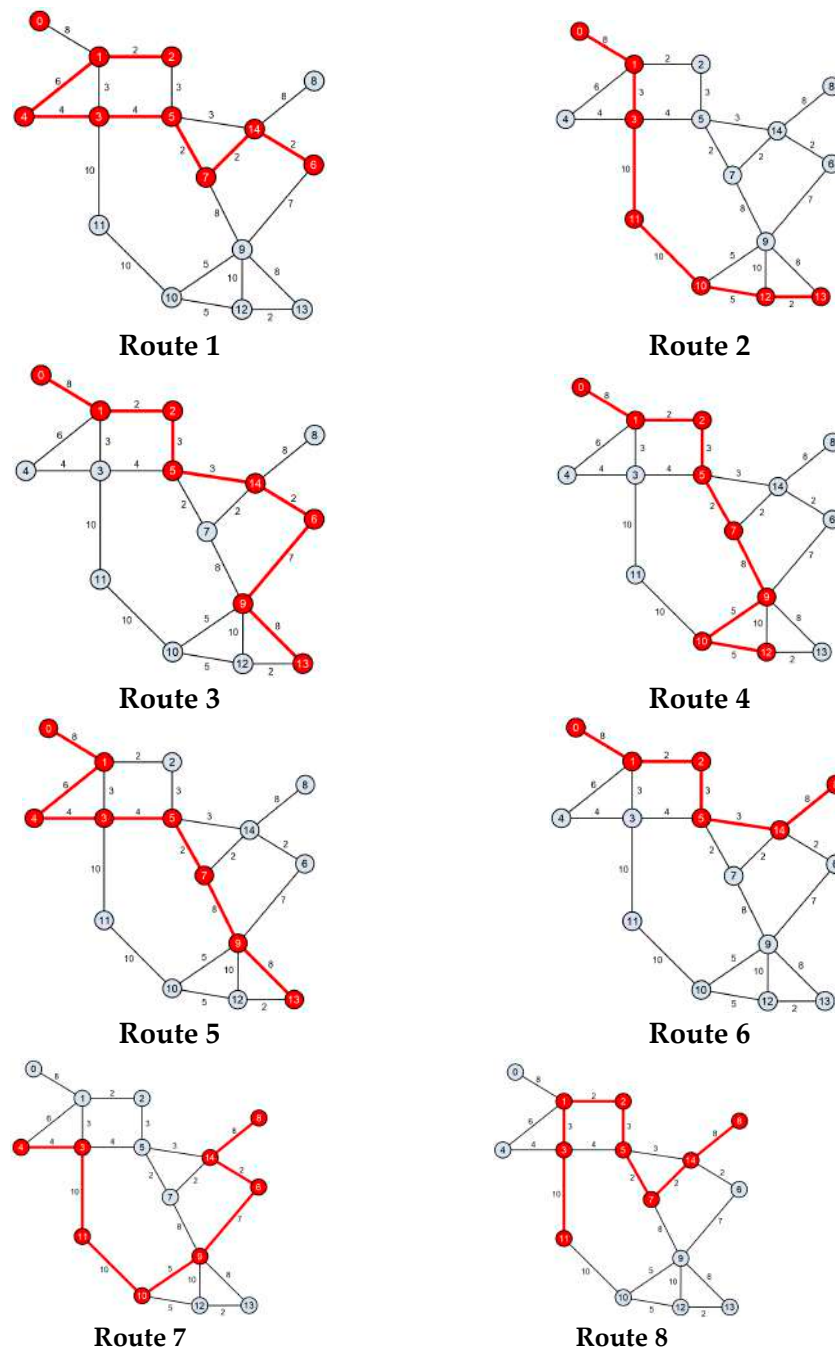


Figure 8. Route set obtained for the 8-route case.



**Table 8.** Best PSO routes for the 8-route case.

Route 1:	6	14	7	5	3	4	1	2
Route 2:	13	12	10	11	3	1	0	
Route 3:	13	9	6	14	5	2	1	0
Route 4:	12	10	9	7	5	2	1	0
Route 5:	0	1	4	3	5	7	9	13
Route 6:	0	1	2	5	14	8		
Route 7:	8	14	6	9	10	11	3	4
Route 8:	8	14	7	5	2	1	3	11

Table 9 provides a comparative analysis of the results obtained for the eight-route case with other relevant studies conducted on the UTRP.

**Table 9.** Comparison between methods for the 8-route case.

Study	[38]	[41]	[37]	[2]	[15]	[31]	[29]	[32]	[17]	[33]	PSO Best	PSO Mean	PSO Std
$d_0(\%)$	79.96	84.73	90.38	94.54	97.81	98.97	97.75	92.87	93.12	98.97	<b>99.68</b>	99.63	0.1
$d_1(\%)$	20.04	15.27	9.62	5.46	2.18	1.03	2.25	6.42	6.87	1.03	<b>0.32</b>	0.37	0.1
$d_2(\%)$	0	0	0	0	0	0	0	0.71	0	0	0	0	0
$d_{un}(\%)$	0	0	0	0	0	0	0	0	0	0	0	0	0
<b>ATT (mpu)</b>	11.86	11.22	10.46	10.36	10.18	10.09	10.13	10.46	10.24	10.08	<b>10.07</b>	10.08	0.01

As indicated in Table 9, the results obtained for the eight-route network demonstrate that PSO outperforms all previous approaches. It achieves a high percentage of direct demand satisfaction and the lowest average travel time, specifically, 10.07 min. It is noteworthy that as the number of routes increases in the transit network, the average travel time decreases and the percentage of direct trips increases, which aligns with our intuition. However, it is worth mentioning that the improvement in solutions for the eight-route case is relatively small compared to the seven-route case. The average run time for this case was 1314 s over 150 iterations.

### 6. Conclusions and Future Work

In this study, an optimization algorithm based on PSO was developed and analyzed to effectively solve the UTRP. The effectiveness of the algorithm was validated by comparing the results obtained with relevant literature based on a set of experimental data for Mandl’s bus network in Switzerland. The study focused on various aspects of the problem, such as initialization, objective function, and computational cost reduction, leading to improved solutions for the UTRP. More specifically,

- An initialization method was developed, ensuring high-quality initial solutions and a diverse range of solutions.
- Significant emphasis was placed on evaluating route sets, selecting and shaping the objective function, and selecting its parameters through multiple trials to identify the parameters leading to the best possible solutions.
- A method was devised to reduce the computational cost of the algorithm by 90% using dynamic programming.
- Comparisons with recent metaheuristics on the UTRP demonstrated the effectiveness of the proposed algorithm, across all scenarios. More specifically, the algorithm achieved higher direct coverage shared under lower or similar average trip times compared to existing implementations.

Future work can consider hybrid algorithmic designs and new metaheuristics. Evidently, there are several new metaheuristics that have been applied in other fields, e.g.,

for reliability-based design optimization [42–44], and have not been explored so far for the UTRP. Such methods include the dragonfly optimizer, the grasshopper optimization algorithm, the salp swarm algorithm, the ant lion optimizer, and the water cycle algorithm [43,44], among others. Similarly, parallel implementations of PSO can be investigated as well, to cut down on computational costs, while hybridization with single-solution metaheuristics such as neighborhood search [45] is also considered promising. Moreover, the use of reinforcement learning techniques to train agents or transit vehicles to make routing decisions based on interactions with the environment can be exploited within the PSO framework. Future research can consider a multi-objective perspective for the UTRP, aiming to determine a set of non-dominated solutions for user and operator costs. In a similar context, environmental considerations can be incorporated in the problem and modelled as a separate objective. A new research direction also includes the introduction of socially aware objectives [46] or transfer efficiency [47] in the UTRP to increase rider comfort within multi-objective settings.

**Author Contributions:** Conceptualization, C.A. and G.N.B.; methodology, V.K. and G.N.B.; software, V.K.; validation, V.K.; formal analysis, V.K. and C.I.; investigation, V.K. and C.I.; data curation, V.K. and C.I.; writing—original draft preparation, C.I. and I.X.T.; writing—review and editing, C.I., I.X.T., C.A. and G.N.B.; visualization, V.K. and C.I.; supervision, I.X.T., C.A. and G.N.B. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** Publicly available datasets were analyzed in this study. These data can be found here: Research on the Urban Transit Routing Problem at <https://users.cs.cf.ac.uk/C.L.Mumford/Research%20Topics/UTRP/Outline.html>, accessed on 21 July 2023.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Iliopoulou, C.; Tassopoulos, I.; Kepaptsoglou, K.; Beligiannis, G. Electric Transit Route Network Design Problem: Model and Application. *Transp. Res. Rec.* **2019**, *2673*, 264–274. [CrossRef]
2. Fan, L.; Mumford, C.L.; Evans, D. A Simple Multi-Objective Optimization Algorithm for the Urban Transit Routing Problem. In Proceedings of the 2009 IEEE Congress on Evolutionary Computation, Trondheim, Norway, 18–21 May 2009; pp. 1–7.
3. Fan, L.; Mumford, C. A Metaheuristic Approach to the Urban Transit Routing Problem. *J. Heuristics* **2010**, *16*, 353–372. [CrossRef]
4. Durán-Micco, J.; Vansteenwegen, P. A Survey on the Transit Network Design and Frequency Setting Problem. *Public Transp.* **2022**, *14*, 155–190. [CrossRef]
5. Mandl, C.E. Evaluation and Optimization of Urban Public Transportation Networks. *Eur. J. Oper. Res.* **1980**, *5*, 396–404. [CrossRef]
6. Iliopoulou, C.; Kepaptsoglou, K.; Vlahogianni, E. Metaheuristics for the Transit Route Network Design Problem: A Review and Comparative Analysis. *Public Transp.* **2019**, *11*, 487–521. [CrossRef]
7. Chakroborty, P.; Wivedi, T. Optimal Route Network Design for Transit Systems Using Genetic Algorithms. *Eng. Optim.* **2002**, *34*, 83–100. [CrossRef]
8. Baaj, M.; Mahmassani, H.S. An AI-Based Approach for Transit Route System Planning and Design. *J. Adv. Transp.* **1991**, *25*, 187–210. [CrossRef]
9. Ceder, A.; Israeli, Y. User and Operator Perspectives in Transit Network Design. *Transp. Res. Rec.* **1998**, *1623*, 3–7. [CrossRef]
10. Zhao, F.; Zeng, X. Optimization of Transit Route Network, Vehicle Headways and Timetables for Large-Scale Transit Networks. *Eur. J. Oper. Res.* **2008**, *186*, 841–855. [CrossRef]
11. Fan, W.; Machemehl, R.B. Using a Simulated Annealing Algorithm to Solve the Transit Route Network Design Problem. *J. Transp. Eng.* **2006**, *132*, 122–132. [CrossRef]
12. Fan, W.; Machemehl, R.B. *Tabu Search Strategies for the Public Transportation Network Optimizations with Variable Transit Demand*; Wiley: Hoboken, NJ, USA, 2008; Volume 23.
13. Pacheco, J.; Alvarez, A.; Casado, S.; González-Velarde, J.L. A Tabu Search Approach to an Urban Transport Problem in Northern Spain. *Comput. Oper. Res.* **2009**, *36*, 967–979. [CrossRef]
14. Roca-Riu, M.; Estrada, M.; Trapote, C. The Design of Interurban Bus Networks in City Centers. *Transp. Res. Policy Pract.* **2012**, *46*, 1153–1165. [CrossRef]
15. Chew, J.S.C.; Lee, L.S. A Genetic Algorithm for Urban Transit Routing Problem. *Int. J. Mod. Phys. Conf. Ser.* **2012**, *9*, 411–421. [CrossRef]

16. Nayeem, M.A.; Rahman, M.K.; Rahman, M.S. Transit Network Design by Genetic Algorithm with Elitism. *Transp. Res. Emerg. Technol.* **2014**, *46*, 30–45. [[CrossRef](#)]
17. Jha, S.B.; Jha, J.K.; Tiwari, M.K. A Multi-Objective Meta-Heuristic Approach for Transit Network Design and Frequency Setting Problem in a Bus Transit System. *Comput. Ind. Eng.* **2019**, *130*, 166–186. [[CrossRef](#)]
18. Buba, A.T.; Lee, L.S. A Differential Evolution for Simultaneous Transit Network Design and Frequency Setting Problem. *Expert. Syst. Appl.* **2018**, *106*, 277–289. [[CrossRef](#)]
19. Buba, A.T.; Lee, L.S. Differential Evolution for Urban Transit Routing Problem. *J. Comput. Commun.* **2016**, *4*, 11–25. [[CrossRef](#)]
20. Zhao, H.; Xu, W.; Jiang, R. The Memetic Algorithm for the Optimization of Urban Transit Network. *Expert. Syst. Appl.* **2015**, *42*, 3760–3773. [[CrossRef](#)]
21. Duran-Micco, J.; Vermeir, E.; Vansteenwegen, P. Considering Emissions in the Transit Network Design and Frequency Setting Problem with a Heterogeneous Fleet. *Eur. J. Oper. Res.* **2020**, *282*, 580–592. [[CrossRef](#)]
22. Hu, J.; Shi, X.; Song, J.; Xu, Y. *LNCS 3611—Optimal Design for Urban Mass Transit Network Based on Evolutionary Algorithms*; Springer: Berlin, Germany, 2005; Volume 3611.
23. Yu, B.; Yang, Z.; Cheng, C.; Liu, C. Optimizing bus transit network with parallel ant colony algorithm. In *Proceedings of the Eastern Asia Society for Transportation Studies*; Semantic Scholar: Seattle, WA, USA, 2005.
24. Yang, Z.; Yu, B.; Cheng, C. A Parallel Ant Colony Algorithm for Bus Network Optimization. *Comput. Aided Civ. Infrastruct. Eng.* **2007**, *22*, 44–55. [[CrossRef](#)]
25. Blum, J.J.; Mathew, T.V. Intelligent Agent Optimization of Urban Bus Transit System Design. *J. Comput. Civ. Eng.* **2011**, *25*, 357–369. [[CrossRef](#)]
26. Yu, B.; Yang, Z.Z.; Jin, P.H.; Wu, S.H.; Yao, B.Z. Transit Route Network Design-Maximizing Direct and Transfer Demand Density. *Transp. Res. Emerg. Technol.* **2012**, *22*, 58–75. [[CrossRef](#)]
27. Szeto, W.; Jiang, Y. Hybrid Artificial Bee Colony Algorithm for Transit Network Design. *Transp. Res. Rec.* **2012**, *2284*, 47–56. [[CrossRef](#)]
28. Jiang, Y.; Szeto, W.; Ng, T. Transit Network Design: A Hybrid Enhanced Artificial Bee Colony Approach and a Case Study. *Int. J. Transp. Sci. Technol.* **2013**, *2*, 243–260. [[CrossRef](#)]
29. Nikolić, M.; Teodorović, D. Transit Network Design by Bee Colony Optimization. *Expert. Syst. Appl.* **2013**, *40*, 5945–5955. [[CrossRef](#)]
30. Nikolić, M.; Teodorović, D. A Simultaneous Transit Network Design and Frequency Setting: Computing with Bees. *Expert. Syst. Appl.* **2014**, *41*, 7200–7209. [[CrossRef](#)]
31. Kechagiopoulos, P.N.; Beligiannis, G.N. Solving the Urban Transit Routing Problem Using a Particle Swarm Optimization Based Algorithm. *Appl. Soft Comput. J.* **2014**, *21*, 654–676. [[CrossRef](#)]
32. Gunby, H.; Gustavsen, S. *A Combined Swarm System for the Urban Transit Routing Problem*; NTNU: Trondheim, Norway, 2015.
33. Katsaragakis, I.V.; Tassopoulos, I.X.; Beligiannis, G.N. Solving the Urban Transit Routing Problem Using a Cat Swarm Optimization-Based Algorithm. *Algorithms* **2020**, *13*, 223. [[CrossRef](#)]
34. Ibarra-Rojas, O.J.; Delgado, F.; Giesen, R.; Muñoz, J.C. Planning, Operation, and Control of Bus Transport Systems: A Literature Review. *Transp. Res. Methodol.* **2015**, *77*, 38–75. [[CrossRef](#)]
35. Kepaptsoglou, K.; Karlaftis, M. Transit Route Network Design Problem: Review. *J. Transp. Eng.* **2009**, *135*, 491–505. [[CrossRef](#)]
36. Newell, G.F. Some Issues Relating to the Optimal Design of Bus Routes. *Transp. Sci.* **1979**, *13*, 20–35. [[CrossRef](#)]
37. Chakroborty, P. Genetic Algorithms for Optimal Urban Transit Network Design. *Comput. Aided Civ. Infrastruct. Eng.* **2003**, *18*, 184–200. [[CrossRef](#)]
38. Hadi Baaj, M.; Mahmassani, H.S. Hybrid Route Generation Heuristic Algorithm for the Design of Transit Networks. *Transp. Res. Part C Emerg. Technol.* **1995**, *3*, 31–50. [[CrossRef](#)]
39. Chakroborty, P.; Deb, K.; Subrahmanyam, P.S. Optimal Scheduling of Urban Transit Systems Using Genetic Algorithms. *J. Transp. Eng.* **1995**, *121*, 544–553. [[CrossRef](#)]
40. Eberhart, R.; Kennedy, J. Particle Swarm Optimization. In *Proceedings of the IEEE International Conference on Neural Networks*, Perth, WA, Australia, 27 November–1 December 1995; pp. 1942–1948.
41. Kidwai, F.A. *Optimal Design of Bus Transit Network: A Genetic Algorithm Based Approach, India*; Indian Institute of Technology: Kanpur, India, 1998.
42. Meng, Z.; Yıldız, A.R.; Mirjalili, S. Efficient decoupling-assisted evolutionary/metaheuristic framework for expensive reliability-based design optimization problems. *Expert. Syst. Appl.* **2022**, *205*, 117640. [[CrossRef](#)]
43. Meng, Z.; Li, G.; Wang, X.; Sait, S.M.; Yıldız, A.R. A comparative study of metaheuristic algorithms for reliability-based design optimization problems. *Arch. Comput. Methods Eng.* **2021**, *28*, 1853–1869. [[CrossRef](#)]
44. Meng, Z.; Li, H.; Zeng, R.; Mirjalili, S.; Yıldız, A.R. An efficient two-stage water cycle algorithm for complex reliability-based design optimization problems. *Neural Comput. Appl.* **2022**, *34*, 20993–21013. [[CrossRef](#)]
45. Iliopoulou, C.; Tassopoulos, I.; Beligiannis, G. A Variable Neighbourhood Search-Based Algorithm for the Transit Route Network Design Problem. *Appl. Sci.* **2022**, *12*, 10232. [[CrossRef](#)]

46. Hoover, S.; Porter, J.D. Building a socially-aware solution to the urban transit routing problem. *Int. J. Transp. Sci. Technol.* **2023**, *12*, 187–200. [[CrossRef](#)]
47. Jing, D.; Yao, E.; Chen, R.; Sun, X. Optimal design method of public transit network considering transfer efficiency. *IET Intell. Transp. Syst.* **2023**, *17*, 1118–1136. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.