Designing Urban Transit Network using Memetic Algorithm

***Abstract*—Urban transit network design (UTND) problem represents a challenge in designing routes, with a trade-off between serving passengers and operators benefits. In this study, a Memetic Algorithm (MA) is proposed to solve the UTND problem. The algorithm uses the hill climbing local search (HCLS) algorithm as an additional operator for Genetic Algorithm (GA) to improve routes construction during the global search. The proposed method consists of two phases. In the first phase, a set of solutions (transit network designs) is generated as an initial population for MA, where each solution consists of a set of routes. The predefined set of solutions satisfies the constraints such as route length or number of routes, and requirements like lack of loops, and that all nodes are covered by at least one route. In the second phase, the suggested Memetic Algorithm (MA) is used to generate all possible solutions from the predefined set. The MA tries to find the best structured solution that represents the flawless transit network. The proposed MA is applied on the widely examined benchmark problems: Mandl and Mumford networks. The experiment results show that the suggested MA provides significant improvements in terms of the direct trip percentage and average travel time compared to the previous studies.**

Keywords—Transit Network Design, Local Search Algorithms, Evolutionary Algorithms.

I. INTRODUCTION

The population in cities is increasing year by year and so are the mobility needs within the urban areas. Therefore, the transportation systems play the main role in the social and economic life at each urban region. The urban transit network design (UTND) is a multi-objective problem that includes objectives related to several factors, such as: total time of journey, walking time, journey cost, number of transits [1]. There are many constraints that influence the transit network design, such as: route length, number of bus routes, lack of loops, fleet size and that each node should be covered by at least one route. UTND is NP-hard, so the optimal design is difficult to find, as shown in [2].

Several authors examined the UTND and proposed different heuristic methods to design a set of routes that involve objectives of minimizing the average travel time and maximizing the trip directness. For instance, a multi-objective model was developed to solve the TND problem using the Bee Colony Optimization (BCO) algorithm with the aim of minimizing the number of transfers and passenger trip time [2]. Also, a hybrid AI-based heuristic method was proposed to solve the TND problem, which consists of the route generation design algorithm (RGA) for constructing a set of routes, the transit route analysis procedure (TRUST) for evaluating the efficiency of the generated routes, and route improvement algorithm (RIA) for enhancing the performance of the designed network [3, 4]. This work was later extended to four-stage framework, including the route generation, network analysis, transit selection, and network improvement procedures [5]. Furthermore, a Memetic algorithm, based on the classical genetic algorithm (GA) and Variable Neighborhood Search (VNS) was used to optimize urban transit routing problem with the aim of minimizing the travel passengers time. The GA-VNS enhanced the quality of final solutions in comparison with using pure metaheuristic algorithms such as GA [6]. Many researches considered not only the quality of service offered to passengers, but also the cost of the operators like fleet size required to cover the set of routes [2, 7-11].

In this study we focus on the transit network design problem, with the aim of minimizing the total in-vehicle time of all served passengers, reducing the total number of transfers in the network and enhancing the passenger satisfaction level. Memetic Algorithm (MA) based on Genetic Algorithm (GA) is used to find the best possible structure that represents the flawless transit network. The transit network found by the algorithm must satisfy the required objectives and must follow the predefined constraints. The suggested algorithm uses hill climbing local search (HCLS) algorithm as an extra operator in GA to enhance the routes structure during the global search.

The rest of this paper is organized as follows. In Section 2 we describe the UTND problem and we introduce the proposed Memetic Algorithm, together with its representation, operators, and fitness function. Section 3 shows the experiment results followed by a discussion. We also compare our solution with the results from the previous studies [2, 3, 7-12]. Finally, in Section 4, we present the conclusions.

II. METHODOLOGY

A. Urban Transit Network Design Problem

The UTND problem is one of the most important problems in transportation planning. The problem is to design the transportation routes that satisfy many criteria, for instance serving users' travel demand represented by an origin–

978-1-7281-8393-0/21/$31.00 ©2021 IEEE

destination matrix with the minimum total average travel time, excluding transfer penalties [10]. We used the well-known Mandl's Swiss road network [12] as shown in Fig. 1 for describing the UTND problem.

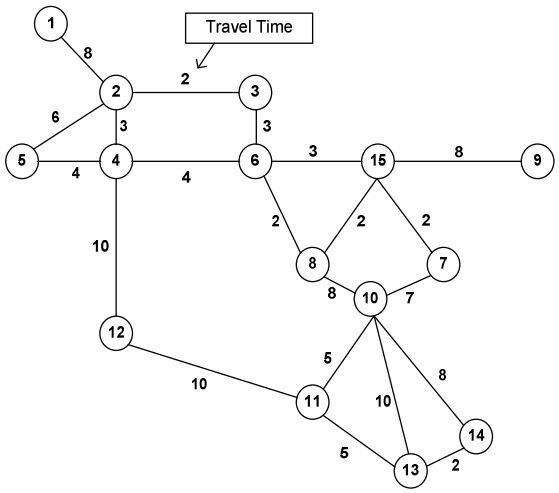


Fig. 1. Mandl's network.

The transit network is represented by an undirected graph $G = (N, A)$, where N is a set of nodes representing bus stops and A is a set of edges representing connections between the stops. In case of Mandl's network we have $|N| = 15$, $|A| = 21$. We also introduce an edge labeling $t: A \rightarrow \mathbb{N}$ representing the specific travel time between the stops, and a demand function $d: N \times N \rightarrow \mathbb{N}$ representing the number of passangers willing to transfer between two given stops. The total demand is defined as $\sum_{i \in N} \sum_{j \in N} d(i,j)$. The original total demand for the Mandl's network is 15,570 passengers, which is demonstrated in the OD matrix presented in [13].

In this study, the solution of the UTND problem is represented as a set of transit routes (that is, a set of paths in G) determined under specific objectives and constraints. From the theoretical point of view, UTND is a variant of the path coverage problem with some additional constraints. The solution can be designed for several scenarios based on the route sizes with distinct transfer penalties where the higher the number of transfers, the higher is the transfer penalty.

B. The Proposed Memetic Algorithm (MA)

In this paper, the UTND problem is solved using Memetic Algorithm, which is an extension of the traditional genetic algorithm. At the beginning, a set of sets of routes is generated randomly to represent several possible urban transit network designs (known as solutions). The predefined network designs must satisfy the constraints and the required objectives. After that, MA is used to reconstruct all the possible routes from the predefined solutions and to find the optimal solution of the UTND problem. The suggested algorithm uses hill climbing local search algorithm to enhance the results of the traditional GA. Algorithm 1 presents the pseudo-code for the proposed approach.

Algorithm 1: Memetic Algorithm

Input: $ProblemSize, PopSize, MemeSize$
Output: S_{best} – the best solution found

$Pop \leftarrow Init.Pop(ProblemSize, PopSize)$
While (**not** StopCondition()) **do**
  foreach $(S_i \in Pop)$
   $S_i.fitness \leftarrow fitness\_fun(S_i)$
  End for
  $Pop_i \leftarrow Selection\ (Pop_{i-1})$
  $Pop_i \leftarrow Intra\_string\ Crossover\ (Pop_i)$
  $Pop_i \leftarrow Rand\_Path\_Mutation\ (Pop_i)$
  $Meme\_Pop \leftarrow Select\ Meme\_Pop(Pop_i, MemeSize)$
  While (**not** StopCondition()) **do**
   $Pop_i \leftarrow HillClimbingLocalSearch$(Meme_Pop)
  End while
End while
$S_{best} \leftarrow GetBestSolutionFrom(Pop_i)$

Algorithm 1 shows how the new solutions (called population) are reconstructed from the old or previous solutions using selection, crossover, mutation and hill climbing local search algorithm.

Route set representation. Memetic algorithm consists of a specific number of iterations (generations) set as a parameter before the algorithm begins its work. In each iteration a set of solutions is generated, initiated randomly before the first iteration. The initiated solutions represent a set of potential solutions for the UTND. Each solution consists of number of routes that represent the transit network design. Each route contains string codes representing the nodes of that route in the designed network, as shown in Fig. 2, where (n_m^i), denotes the m-th node in the i-th route.

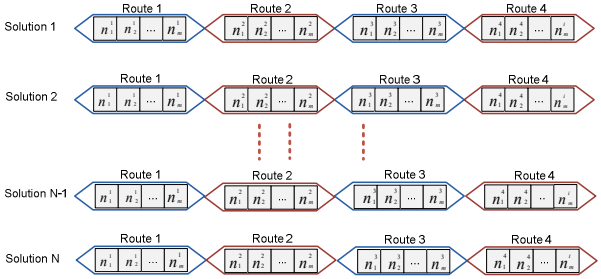
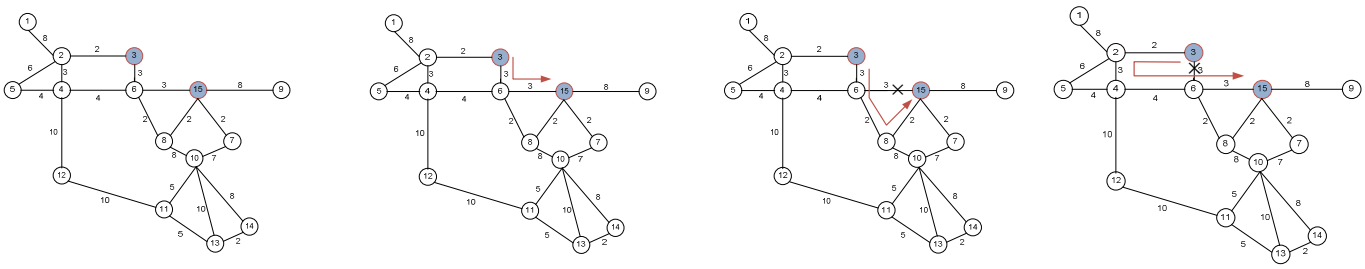


Fig. 2. The example of the initiated transit network solutions.

Fig. 2 shows an example of the initial transit network solutions ($solution_N$), where N here is the number of the transit network solutions. Each solution is a set of a fixed number of routes (here: 4) and each route is a sequence of nodes (n_m), where m here is the number of nodes at each route taking into account that m may varies from one route to another.

Initialization procedure. The idea of the initialization procedure is to form a set of transit routes for each origin–destination pair. Next, the initialized routes are used to form the initial transit network solutions, which are used as an input for the MA operators, such as selection, crossover, mutation and hill climbing local search. After that, the initial solutions are pa-



A. Origin–destination pair in our case. B. 1^{st} shortest path (K=1). C. 2^{nd} shortest path (K=2). D. 3^{rd} shortest path (K=3).

Fig. 3. An example for the k = 3 shortest path.

ssed through the MA operators, where the solutions quality is enhanced based on minimizing the travel time and the number of transits among the routes. In this study, we used the algorithm developed by Yen [14] to create a set of transit routes for each origin–destination pair. Yen's algorithm divides the problem of finding the K shortest path into sub-problems. For each sub-problem a standard shortest path algorithm (Dijkstra's algorithm) is used to find the K shortest path. In Fig. 3A, an example shows how Yen's algorithm works to find the 3^{rd} shortest path in the graph of the Mandl's road network, where the origin and destination are nodes 3 and 15 respectively. Yen's algorithm divides this problem into three sub-problems. First, Yen's algorithm is looking for the 1^{st} shortest path between nodes 3 and 15 as shown in Fig. 3B. In Fig. 3B, Yen's algorithm just runs Dijkstra's algorithm [15] to find the shortest path from 3 to 15. The result is the path (3, 6, 15). Next, Yen's algorithm is looking for the 2^{nd} shortest path between the same nodes 3 and 15 as shown in Fig. 3C. In Fig. 3C, Yen's algorithm uses the 1^{st} shortest path to find the 2^{nd} shortest path forcing Dijkstra's algorithm along a different, less optimal route by removing one of the edges, which is a part of the 1^{st} shortest path. Therefore, Yen's algorithm tries to remove edge (3,6), which is the first edge in 1^{st} shortest path, and then runs Dijkstra's algorithm. After that, Yen's algorithm tries to remove the edge (6,15), which is the second edge in the 1^{st} shortest path, and then runs Dijkstra's algorithm. At the end, the shortest of the alternative new paths, which are generated by the forced Dijkstra's algorithm, is chosen as the 2^{nd} shortest path. Next, Yen's algorithm is looking for the 3^{rd} shortest path between the same nodes 3 and 15 as shown in Fig. 3D. In Fig 3D, as the 2^{nd} shortest path was found by removing edges from the 1^{st} shortest path, the 3^{rd} shortest path is found by removing edges from the 2^{nd} shortest path with avoiding getting the 1^{st} or the 2^{nd} shortest path as a repeated result.

In this study, Yen's algorithm is used to generate the routes, which are used to build the initial population (solutions) of MA in three stages. First, Yen's algorithm creates the routes that serve the 'isolated nodes' (i.e. the nodes with only one incident edge such as node 1 and 9 in case of Mandl's road network) in the transit network. Hence, the created routes must start or end at these nodes. Second, Yen's algorithm generates routes that serve each OD pair with a nonzero travel demand except the pairs that are covered in the first stage. In the two previous stages, Yen's algorithm creates more than one route for each OD pair to ensure diversity in the generated solutions. Third, a set of solutions is generated and each solution consists of a set of routes randomly selected from the initial routes, taking into account the predefined constraints shown in Table 1.

Modification procedure. Modification procedure plays an important role of the optimization process, using the traditional GA operators and hill climbing local search (HCLS) to discover a better set of solutions. There are three standard operators: selection, crossover and mutation, related to the traditional GA. The HCLS is considered as the fourth operator of the suggested MA. The four operators are defined in the following subsections, together with examples.

Selection operator. An elitist population replacement strategy is used for selecting and transferring the route sets (solutions) to the next generation based on the 'survival of the fittest' concept. Elitism can very rapidly increase the performance of MA, because it prevents losing the best found route sets (solutions).

Crossover operator. The intra-string crossover is used to make smooth changes to the initial route set, where the individual solutions are modified at the node level maintaining linkage. Two routes (parents) are selected randomly in the same solution, and two sets of nodes in the selected routes are interchanged to generate two offsprings. In this study, the crossover probability is set to 0.5 and the results are passed through the next evolutionary approach mutation operator. The intra-string crossover operation is illustrated in Fig. 4.

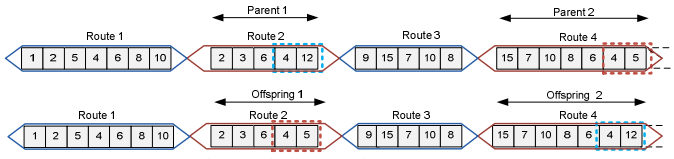


Fig. 4. The intra-string crossover.

Fig. 4 shows an example of the intra-string crossover, where a sequence of nodes is swapped between two routes, named as Parent 1 and Parent 2. The intra-string crossover may happen only when both parents have at least one node in common. In this case, the sequence of nodes after the intersection node can be swapped between the two parents. Otherwise, the intra-string

crossover cannot be applied. In Fig. 4 the common node is 6, so the fragments denoted by the dashed rectangles can be swapped to get two offspring.

Mutation operator. Mutation is an important operator in the evolutionary approach, as it introduces the diversity, allowing us to explore better the solution space and not get stuck in a local minimum. We designed our mutation to create slight modification with allowing big jump only occasionally to avoid the local optima. Mutation is accomplished by selecting randomly two different nodes within a route. Then the path between them is modified using the k-shortest paths to ensure smooth change in the route [14] as shown in Fig. 5. In this study, the probability of mutation is set to 0.3.

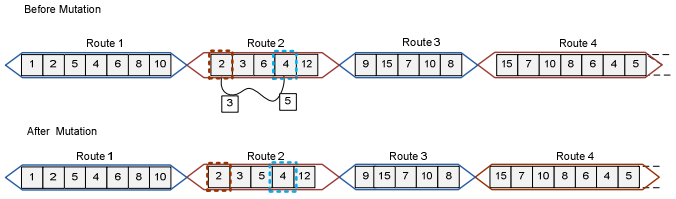


Fig. 5. Mutation on the route.

Fig. 5 shows the sample mutation process. First, we randomly select two nodes (in this case nodes 2 and 4) in the route 2. Then, we find k paths between them using the k-shortest paths technique, where k value is selected within the range [3,5] arbitrarily. After that, we replace one of them (2, 3, 5, 4) with the original path (2, 3, 6, 4) in the route 2. The important part of the mutation process is not only the random change of the path between two nodes in a certain route, but also making sure that the generated solution is feasible.

Hill climbing local search (HCLS). Apart from mutation, selection and crossover, HCLS is used as the fourth genetic operator in the proposed MA to refine the solutions. HCLS tries to find the best possible solution within each iteration of GA. At each iteration of the traditional GA a group of solutions, called memetic population, is randomly selected from the population. Next, the memetic population is passed to the HCLS algorithm, where route locations are exchanged among the solutions at each iteration of the HCLS (the number of the HCLS iterations are adjusted based on the number of the routes in the solution of each experiment to give the best result). At each HCLS iteration, the fitness values of the updated solutions are calculated and only if the fitness values of the solutions are better than the ones in the previous HCLS iteration, the new solutions are added to the next HCLS iteration. Otherwise, the updated solutions are neglected and the process starts over in the next iteration of HCLS. The number of HCLS iterations can be considered as stopping condition for the local search. As well as, the HCLS results will be ignored, if HCLS doesn't provide any improvement during its iterations to avoid any misleading information caused by getting stuck in local minima. Fig. 6. shows routes exchanging among the solutions at each iteration of the HCLS: the memetic population is passed to the HCLS algorithm, where the route locations are exchanged among the solutions within a specific number of iterations to create better solutions. For example, routes 4 and 2 in solutions 1 and 3 are exchanged with routes 1 and 3 in solutions 2 and 4 respectively. If the fitness values of the updated solutions are better than the previous ones, the updated solutions are passed to the next HCLS iteration. Otherwise, the updated solutions are neglected and the solutions from the previous HCLS iteration are passed to the next HCLS iteration. At the end of the HCLS algorithm, the enhanced memetic population will be received back as a part of the population in the current iteration of GA.

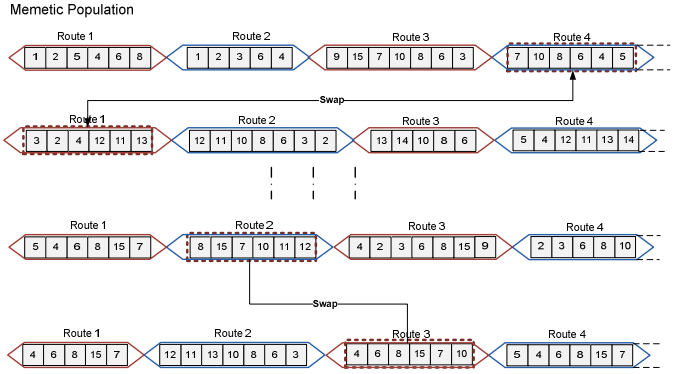


Fig. 6. Hill climbing local search (HCLS).

Routes set evaluation. The evaluation process comprises all nodes of a routes set to measure the quality of each routes set. The evaluating process is based on the following criteria: (1) The average in-vehicle travel time for all passengers of the transit network; (2) The percentage of passengers traveling directly, with a single transfer, or with a double or more than two transfers from their origin to their destination.

In the present research, the total fitness value $TF(R_s)$ of a given routes set R_s is a weighted sum of the following terms [10]:

$$TF(R_s) = w_1 ATT(R_s) + w_2 PT(R_s) + w_3 PUD(R_s), \quad (1)$$

with

$$ATT(R_s) = [TIVT(R_s) + P_1 * T_1(R_s) + P_2 * T_2(R_s)]/ \text{total passenger demand}, \quad (2)$$

$$PT(R_s) = \omega_1[-PT_0(R_s)] + \omega_2 PT_1(R_s) + \omega_3 PT_2(R_s), \quad (3)$$

where:

$TF(R_s)$ is the total fitness of the designed routes set;

$ATT(R_s)$ is the average travel time incurred by the passengers in the transit network, consisting of in-vehicle travel time and transfer penalty time for one-transfer and two-transfers;

$PT(R_s)$ is the weighted sum of percentages of direct trips (without any transfer), one-transfer trips and two-transfer trips experienced by all passengers in the transit network;

$PUD(R_s)$ is the percentage of passengers who need more than two transfers in the transit network, and are considered as the unsatisfied demand.

w_1, w_2, w_3 are the non-negative weights and their values are selected based on the importance of the term that coupled with, $w_1 + w_2 + w_3 = 1$;

$TIVT(R_s)$ is the total in-vehicle travel time of passengers in the

transit network;
$T_1(R_s)$ is the total number of passengers in the transit network who have travelled with one-transfer;
$T_2(R_s)$ is the total number of passengers in the transit network who have travelled with two-transfers;
P_1: is the penalty time for each one-transfer trip, $P_1 > 0$;
P_2: is the penalty time for each two-transfer trip, $P_2 > 0$;
$PT_0(R_s)$ is the percentage of direct trips;
$PT_1(R_s)$ is the percentage of one-transfer trips;
$PT_2(R_s)$ is the percentage of two-transfer trips;
$\omega_1, \omega_2, \omega_3$ are the non-negative weights and their values are selected based on the importance of the term that coupled with, $\omega_1 + \omega_2 + \omega_3 = 1$, and $\omega_1 \geq \omega_2 \geq \omega_3$.
From Eq. (3) we see that the percentage of passengers with direct trips should be maximized, while other terms should be minimized. The higher the percentage of direct trips, the lower the penalty of the transfer time.
After the evaluating process, the fitness value of each routes set is calculated to select solutions for the next generation.
Termination. The termination criterion is defined by reaching the maximum number of iterations. When the termination criterion is satisfied, the MA algorithm stops and returns the solution of the UTND, being the solution with the smallest fitness value within the population from the last iteration.

III. Experimental Results

In this section, the performance of the proposed MA is evaluated through the extensive experiments. In these experiments, the proposed MA is applied to the commonly used benchmark problems: Mandl and Mumford networks. This approach allowed us to compare the experimental results of the MA with the previous studies and investigate the reliability of the performance of our algorithm.

A. Datasets and Experimental Setup

We have conducted experiments on Mandl's Swiss Road Network [12] and three large datasets proposed in [16], known as, Mumford1, Mumford2 and Mumford3. Mandl and Mumford datasets are widely examined as the benchmark networks. The properties of these datasets and the number of runs performed on each dataset are shown in Table 1. The proposed algorithm parameters used for our experiments are shown in Table 2. Three scenarios are used for the transfer penalty, as presented in Table 3 [10].

B. Computational Results

In this section, the experimental results of the proposed MA are presented. In order to determine the effectiveness of the suggested algorithm, the experimental results of the previous researches are compared with the results generated by our MA. In Table 5 we show the comparison for Mandl dataset with four different route set sizes and in Table 6 – the comparison for Mumford datasets. Regarding the previous researches, the best solutions achieved by the proposed MA are compared using four performance metrics presented in Table 4 [10]. Moreover, the average and standard deviation values presented are the ones calculated over all the runs performed for each dataset as referred in Table 1 [10]. In bold are the values which determine the best achieved results.
For each scenario of the Mandl's dataset, the relevant experimental results of MA are compared to the previous works [2, 3, 7-12]. As shown in Table 5, the percentage of the total transfer demands satisfied directly (without any transfers), obtained by the proposed MA, is fully satisfied – it is equal to 100%, and the ATT is better than the previous works in the case of eight and seven routes. The percentages of total transfer demands satisfied directly, found by the proposed MA, are equal to 98.97% and 94.16% in the case of six and four routes respectively. The obtained results are significantly better than other approaches and the ATT is near to the values of the previous researches as well. Therefore, we can conclude that MA can achieve better results compared to other approaches [2, 3, 7-12] in terms of the transfer demand and the ATT where the results enhance as the route size increases in the examined scenarios.

TABLE 1. DATA SET CONSTRAINTS USED IN THE EXPERIMENTS.

Data set	Nodes	Links	Route Length	No. of Routes	Runs Performed
Mandl	15	20	2-8	4,6,7,8	100
Mumford1	70	210	10-30	15	20
Mumford2	110	385	10-22	56	20
Mumford3	127	425	12-25	60	20

TABLE 2. PARAMETER VALUES FOR THE UTND.

Parameters	values
Population size, N	20
Maximal number of generations, G	100
HCLS iterations	4 -10
Elite Size	4
Route crossover probability (intra-string crossover), p_{rc}	0.5
Mutation probability	0.3
Weights w_1, w_2, w_3 associated with $TF(R_s)$, $\sum_p w_p = 1$, where $p = 1,2,3$	0.4, 0.5, 0.1
Weights $\omega_1, \omega_2, \omega_3$ associated with $PT(R_s)$, $\sum_p \omega_p = 1$, where $p = 1, 2, 3$ and $\omega_1 \geq \omega_2 \geq \omega_3$	0.6, 0.3, 0.1

TABLE 3. TRANSFER PENALTIES SCENARIOS [10].

Transfer penalty (in minutes)		
Scenario	First transfer	Second transfer
1	5	17
2	30	40
3	15	20

TABLE 4. PERFORMANCE METRICS.

Metrics	Definitions
d_0	The percentage of demand satisfied with direct trips
d_1	The percentage of demand satisfied with one transfer
d_2	The percentage of demand satisfied with two transfers
d_{un}	The percentage of demand unsatisfied
ATT	Average travel time of all served passengers expressed in minutes per transit user (mpu). This travel time includes transfer penalty (i.e. 5 min per passenger)

TABLE 5. COMPARISON BETWEEN MA AND PREVIOUS METHODS ON MANDL DATASET.

Route size	Source		d_0	d_1	d_2	ATT-1	ATT-2	ATT-3
8	Baaj (1991)		79.96	20.04	0.00	11.86	16.87	13.86
	Chakroborty (2002)		90.38	9.58	0.00	10.46	12.86	11.42
	Chew (2013)		99.04	0.96	0.00	10.19	10.43	10.29
	Nikolić (2013)		98.97	1.03	0.00	10.09	10.35	10.19
	Kechagiopoulos (2014)		97.75	2.25	0.00	10.13	10.69	10.36
	Arbex (2015)		98.65	1.35	0.00	11.31	11.65	11.44
	Jha & Tiwari (2019)		93.12	6.87	0.00	10.24	11.96	10.93
	The proposed MA	Best	**100**	**0.00**	0.00	**10.05**	**10.05**	**10.05**
		Worst	99.15	0.85	0.00	10.08	10.17	10.11
		Mean	99.62	0.32	0.00	10.063	10.09	10.07
		Std.	0.397	0.397	0.00	0.0307	0.0302	0.0309
7	Chew (2013)		98.01	1.99	0.00	10.27	10.77	10.47
	Nikolić (2013)		98.84	1.16	0.00	10.15	10.44	10.27
	Kechagiopoulos (2014)		7.17	2.83	0.00	10.16	10.87	10.44
	Arbex (2015)		98.52	1.48	0.00	11.98	12.35	12.13
	Jha & Tiwari (2019)		99.42	0.58	0.00	11.85	12.00	11.91
	The proposed MA	Best	**100**	**0.00**	0.00	**10.13**	**10.13**	**10.13**
		Worst	99.59	0.41	0.00	10.17	10.41	10.20
		Mean	99.75	0.19	0.00	10.149	10.22	10.15
		Std.	0.705	0.705	0.00	0.0355	0.0350	0.0352
6	Baaj (1991)		78.61	21.39	0.00	11.86	17.21	14.00
	Chakroborty (2002)		86.04	13.96	0.00	10.30	13.79	11.70
	Chew (2013)		98.14	1.86	0.00	10.28	**10.75**	**10.47**
	Nikolić (2013)		97.24	2.76	0.00	**10.23**	10.92	10.51
	Kechagiopoulos (2014)		96.21	3.47	0.32	**10.23**	11.17	10.59
	Arbex (2015)		98.20	1.80	0.00	11.64	12.09	11.82
	Jha & Tiwari (2019)		98.46	1.54	0.00	12.61	12.99	12.76
	The proposed MA	Best	**98.97**	**1.03**	0.00	11.22	11.47	11.31
		Worst	98.53	1.47	0.00	11.27	11.53	11.39
		Mean	98.78	1.23	0.00	11.23	11.40	11.339
		Std.	0.584	0.562	0.00	0.028	0.069	0.028
4	Mandl (1980)		69.64	29.93	0.13	12.90	20.41	15.90
	Chakroborty (2002)		89.98	10.02	0.00	11.90	14.41	12.90
	Chew (2013)		92.74	7.26	0.00	11.16	12.98	11.89
	Nikolić (2013)		91.91	8.09	0.00	**10.51**	12.53	11.32
	Kechagiopoulos (2014)		91.84	7.64	0.51	10.64	**11.35**	**10.92**
	Arbex (2015)		**98.27**	**1.73**	0.00	11.22	11.65	11.39
	Jha & Tiwari (2019)		97.17	2.83	0.00	13.13	13.84	13.41
	The proposed MA	Best	94.16	5.48	0.00	12.90	14.36	13.48
		Worst	93.98	6.02	0.00	13.11	14.41	13.52
		Mean	94.058	5.75	0.00	13.001	14.218	13.29
		Std.	0.097	0.21	0.00	0.098	0.0195	0.0197

As mentioned before, we performed the additional experiments using the proposed MA on the Mumford datasets and a comparison took place to prove the efficiency of the suggested MA compared to the other methods [6, 16-20]. Table 6 illustrates that MA can achieve better results in terms of trip directness and ATT compared to the previous approaches [6, 16-20]. In terms of trip directness, the suggested MA made a noticeable improvement by 3.37%, 3.78%, 7.03% compared with the best previous work results in Mumford1, Mumford2 and Mumford3 respectively. The ATT was competitive with a difference ranges as 0.02, 0.07, 0.13 compared to the finest ATT of the previous work [20].

One may argue that the performance of the GA can be affected by using HCLS algorithm. Therefore, we computed the fitness values of the pure GA and MA with mutation operator disabled (just HCLS) in Mandl_4 and Mumford3 to evaluate the efficiency of our suggested algorithm (MA).

The experiments on Mandl and Mumford networks show that the suggested MA provides significant enhancements with respect to the values of the fitness function, compared with the slow pure GA and MA with mutation operator disabled which falls in local minima as shown in Figs 7 and 8.

Moreover, we computed the average execution time (in seconds) for MA, pure GA and MA with mutation operator disabled (HCLS) over all the experiments to estimate the effect of using HCLS with GA in our proposed MA. A comparison is taken place between the execution time of particle swarm optimization (PSO) algorithm from [11], pure GA, MA with mutation operator disabled and our suggested approach in order to prove the effectiveness of the MA.

TABLE 6. COMPARISON BETWEEN MA AND PREVIOUS METHODS ON MUMFORD DATASETS.

Dataset	No. of Routes	Metric	Mumford (2013) [16]	Fatih (2014) [17]	GAWE (2014) [18]	Rahman (2015) [19]	Islam, K.A., et al (2019)[20]	Proposed MA			
								Best	Worst	Mean	Std.
Mumford1	15	d_0	35.53	44.66	41.39	44.34	55.10	**55.23**	48.68	51.571	2.897
		d_1	51.74	49.41	53.14	52.10	43.74	43.66	51.18	47.926	2.698
		d_2	12.35	5.89	4.13	3.56	1.16	1.11	0.14	0.863	0.098
		d_{un}	0.39	0.04	0.02	0.00	0.00	0.00	0.00	0.00	0.00
		ATT	24.71	23.31	23.64	23.38	22.11	**22.09**	23.13	22.48	0.37
Mumford2	56	d_0	29.32	34.60	34.09		51.86	**58.89**	51.89	54.874	2.635
		d_1	50.40	58.87	60.98		47.98	41.26	48.29	43.453	3.687
		d_2	18.30	0.00	4.92		0.16	0.15	0.18	0.104	0.00978
		d_{un}	1.98	0.02	0.01		0.00	0.00	0.00	0.00	0.00
		ATT	28.17	26.76	26.58		25.89	**25.82**	26.12	23.68	**0.098**
Mumford3	60	d_0	26.35	28.39	30.19	31.09	50.76	**57.79**	50.7	53.245	2.455
		d_1	50.16	57.06	61.23	61.74	48.87	42.02	49.57	440795	2.775
		d_2	20.04	13.84	8.56	7.14	0.38	0.19	0.27	0.189	0.037
		d_{un}	3.45	0.70	0.01	0.03	0.00	0.00	0.00	0.00	0.00
		ATT	30.49	30.12	29.57	29.40	27.64	**27.51**	28.24	26.342	0.265

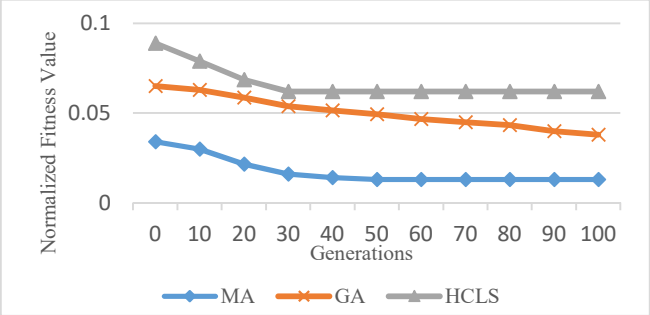


Fig.7. Fitness function value in case of MA, GA and HCLS in Mandl_4 through 100 iterations.

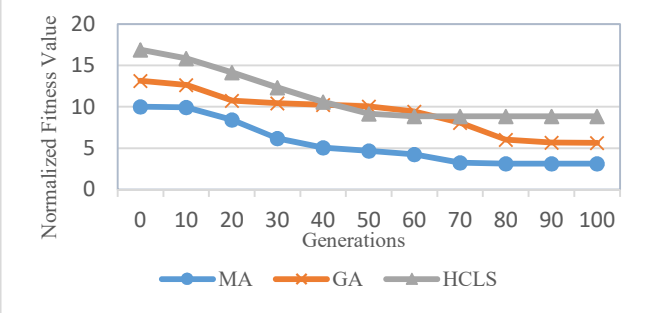


Fig. 8. Fitness function value in case of MA, GA and HCLS in Mumford3 through 100 iterations.

TABLE 7. COMPARISON BETWEEN MA, GA, HCLS AND PSO ON MANDL DATASET.

Network	NO. of Node	Route size	PSO	Avg time GA	Avg Time HCLS	Avg Time MA
Mandl	15	4	280.76	88.36	98.23	186.59
		6	300.23	111.85	89.14	200.99
		7	321.62	138.31	81.93	220.24
		8	344.3	174.37	74.59	248.96
Mumford1	70	15		189.99	41.39	231.38
Mumford2	110	56		211.32	54.89	266.21
Mumford3	127	60		235.41	59.95	295.36

In Table 7 the average execution time (in seconds) is presented for the performed experiments whose results are presented in Tables 5 and 6 for both datasets Mandl and Mumford. All the experiments were applied on a laptop of 2.20 GHz Intel Core i7 processor with 6 GB RAM. In the case of Mandl network, the proposed algorithm proved its efficiency compared with [11] which used the particle swarm optimization algorithm to solve only Mandl transit routing problem. However, we were not able to compare the algorithms regarding time on Mumford dataset, due to lack of data.

In conclusion, applying the hill climbing search on the solutions generated by the GA makes a significant improvement. Also, when the size of the routes increases, the proposed algorithm performs increasingly better compared to the previous approaches in both Mandle and Mumford network problems. The experiments prove that MA can effectively explore the huge search space of the transit network design problem.

IV. CONCLUSIONS

In this study we proposed the Memetic Algorithm to solve the urban transit network design (UTND) problem. The main issue in this problem is to determine an efficient set of routes that provides the best trips for passengers with minimum travel time and number of transits between the routes. We performed experiments based on the well-known benchmarks, the Mandl's Swiss network [12] and three large datasets known as Mumford networks [16]. The results show that hill climbing local search significantly improves the solutions quality in comparison to the traditional GA.

In the case of Mandl's network, the proposed algorithm wins in terms of the direct trip percentage in case of eight, seven and six routes. In the two first scenarios it was able to achieve 100% of the total transfer demands satisfied directly. In case of six routes it is still the best one among others, but achieves 98.97% (the rest, 1.03%, is a 1-transfer case). Only in case of four routes our solution is worse than two other studies, but better than the remaining studies. In general, the results are quite promising, when compared with the previous works. Although the ATT

values (in all three scenarios) are not the best ones, they are comparable with other approaches.

In the case of Mumford networks, the suggested algorithm provides the best results in terms of the direct trips with percentage difference of 3.37%, 3.78%, 7.03% compared with [20] in Mumford1, Mumford2 and Mumford3 respectively. Despite the good results of MA, the results of ATT are nearly close to the best ATT of the previous work as a trade-off of seeking for the direct trip.

In conclusion, applying hill climbing search on the solutions generated by GA gives us a significant improvement comparing to the previous works on both Mandle and Mumford datasets.

REFERENCES

[1] W. Fan and R. B. Machemehl, "Optimal transit route network design problem with variable transit demand: genetic algorithm approach," *Transp. Eng. J. ASCE*, vol. 132, pp. 40-51, 2006.

[2] M. Nikolić and D. Teodorović, "Transit network design by bee colony optimization," *Expert. Syst. Appl*., vol. 40, pp. 5945-5955, 2013.

[3] M. H. Baaj and H. S. Mahmassani, "An AI-based approach for transit route system planning and design," *J. Adv. Transp.*, vol. 25, pp. 187-209, 1991.

[4] M. H. Baaj and H. S. Mahmassani, "Hybrid route generation heuristic algorithm for the design of transit networks," *Transp. Res. Part C Emerg. Technol*., vol. 3, pp. 31-50, 1995.

[5] M.-C. Shih and H. S. Mahmassani, "A design methodology for bus transit networks with coordinated operations," *Transp. Res. Rec*., vol. 162. pp. 16-32, 1994.

[6] J. Koszelew and K. Ostrowski, "A memetic algorithm for routing in urban public transportation networks," in *ICAIS*, 2011, pp. 368-380.

[7] R. O. Arbex and C. B. da Cunha, "Efficient transit network design and frequencies setting multi-objective optimization by alternating objective genetic algorithm," Tr*ansport. Res. B-Meth*., vol. 81, pp. 355-376, 2015.

[8] P. Chakroborty and T. Wivedi, "Optimal route network design for transit systems using genetic algorithms," *Eng. Optim.*, vol. 34, pp. 83-100, 2002.

[9] J. S. C. Chew and L. S. Lee, "A genetic algorithm for urban transit routing problem," in *Int. J. Mod. Phys. Conf. Ser*., 2012, pp. 411-421.

[10] S. B. Jha, J. K. Jha, and M. K. Tiwari, "A multi-objective meta-heuristic approach for transit network design and frequency setting problem in a bus transit system," *CAIE*, vol. 130, pp. 166-186, 2019.

[11] P. N. Kechagiopoulos and G. N. Beligiannis, "Solving the urban transit routing problem using a particle swarm optimization based algorithm," *Appl. Soft Comput*., vol. 21, pp. 654-676, 2014.

[12] C. E. Mandl, "Evaluation and optimization of urban public transportation networks," *Eur. J. Oper. Res.*, vol. 5, pp. 396-404, 1980.

[13] S. Chai and Q. Liang, "An improved nsga-ii algorithm for transit network design and frequency setting problem," *J. Adv. Transp*., vol. 2020, pp. 298–313, 2020.

[14] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Manage. Sci*., vol. 17, pp. 712-716, 1971.

[15] O. Khaing, D. Wai, and D. Myat, "Using Dijkstra's algorithm for public transportation system in yangon based on GIS," *Int. J. Sci. Eng. Appl*., vol. 7, pp. 442-447, 2018.

[16] C. L. Mumford, "New heuristic and evolutionary operators for the multi-objective urban transit routing problem," in *CEC*, 2013, pp. 939-946.

[17] F. Kılıç and M. Gök, "A demand based route generation algorithm for public transit network design," *Comput. Oper. Res*., vol. 51, pp. 21-29, 2014.

[18] M. A. Nayeem, M. K. Rahman, and M. S. Rahman, "Transit network design by genetic algorithm with elitism," *Transp. Res. Part C Emerg. Technol*., vol. 46, pp. 30-45, 2014.

[19] M. K. Rahman, M. A. Nayeem, and M. S. Rahman, "Transit network design by hybrid guided genetic algorithm with elitism," in *CASPT*, 2015, pp.30-45.

[20] K. A. Islam, I. M. Moosa, J. Mobin, M. A. Nayeem, and M. S. Rahman, "A heuristic aided Stochastic Beam Search algorithm for solving the transit network design problem," *Swarm Evol. Comput*., vol. 46, pp. 154-170, 2019.