Transit network design by genetic algorithm with elitism

Article *in* Transportation Research Part C Emerging Technologies · September 2014

DOI: 10.1016/j.trc.2014.05.002

3 authors:

Muhammad Ali Nayeem
Bangladesh University of Engineering and Technology

Khaled Rahman
Meta

Mohammad Sohel Rahman
Bangladesh University of Engineering and Technology

Transportation Research Part C 46 (2014) 30–45

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Transit network design by genetic algorithm with elitism

Muhammad Ali Nayeem, Md. Khaledur Rahman, M. Sohel Rahman *

AℓEDA group, Department of CSE, BUET, Dhaka 1000, Bangladesh

ARTICLE INFO

Article history:
Received 30 November 2013
Received in revised form 11 April 2014
Accepted 2 May 2014

Keywords:
Transit network
Genetic algorithm
Elitism

ABSTRACT

The transit network design problem is concerned with the finding of a set of routes with corresponding schedules for a public transport system. This problem belongs to the class of NP-Hard problem because of the vast search space and multiple constraints whose optimal solution is really difficult to find out. The paper develops a Population based model for the transit network design problem. While designing the transit network, we give preference to maximize the number of satisfied passengers, to minimize the total number of transfers, and to minimize the total travel time of all served passengers. Our approach to the transit network design problem is based on the Genetic Algorithm (GA) optimization. The Genetic Algorithm is similar to evolution strategy which iterates through fitness assessment, selection and breeding, and population reassembly. In this paper, we will show two different experimental results performed on known benchmark problems. We clearly show that results obtained by Genetic Algorithm with increasing population is better than so far best technique which is really difficult for future researchers to beat.

1. Introduction

With the increase in population and urbanization, efficient public transport systems are urgently needed. But finding an efficient public transport system is very difficult. This is why extensive research work has been conducted on relevant topics in the literature. Although achieving a major increase in public transit usage is an extremely complex issue, frequent and reliable cost-effective services are center of focus.

One important issue in transit network design is the vast search space. There are many factors that are mutually in conflict with each other while designing transit networks. For instance, the shorter the passengers' waiting times, the higher the number of vehicles necessary. When designing the transit network, the interests of both the operator and the passenger must be taken into account. Because of these conflicting nature of interests, we treat the transit network design problem as a multicriteria decision-making problem. In this paper, we investigate the problem from a different perspective. While designing the transit network, we aim to maximize the total number of satisfied passengers, to minimize the total number of transfers, and to minimize the total travel time of all served passengers.

Basically, route designers have relied much on historical experience, simple guidelines, local knowledge and ad hoc procedures. However, in recent years, several major studies have revealed that computer based tools should be employed more for designing and evaluating public transit networks. In the present paper, we are mainly concerned with route planning which involves the following objectives (Nikoli and Teodorovi, 2013): (a) to maximize the number of satisfied passengers, (b) to minimize the total number of transfers and (c) to minimize the total travel time of all the served passengers.

* Corresponding author. Tel.: +880 1552389480; fax: +880 448712475276.
E-mail address: sohel.kcl@gmail.com (M.S. Rahman).

http://dx.doi.org/10.1016/j.trc.2014.05.002
0968-090X/

The transit network design problem can be subdivided into two major components, namely, the transit routing problem and the transit scheduling problem (Chakroborty, 2003). Generally, the transit routing problem involves the development of efficient transits routes (e.g., bus routes) on an existing road network, with predefined pickup/dropoff points (e.g., bus stops). On the other hand, the transit scheduling problem is charged with assigning the schedules for the passenger carrying vehicles. In practice, the two phases are usually implemented sequentially (or iteratively), with the routes determined in advance of the schedules.

In this paper, we discuss the transit routing problem and present a metaheuristics framework for solving it. We introduce two different versions of Genetic Algorithm (GA) for the transit network design problem, that allow us to concentrate on the key issues of minimizing the travel time and the number of transfers simultaneously. We show the effectiveness of our schemes, by comparing our results with previously published results on a benchmark instance. Furthermore, we explore the scalability of our approach by testing it on some larger instances, generated by Nikoli and Teodorovi (2013).

The rest of this paper is organized as follows. Literature review is given in Section 2. Section 3 formally defines the problem. Proposed solutions to the problem are given in Section 4. Experimental results and analyses are provided in Section 5. Finally, we briefly conclude in Section 6 with some future research directions.

2. Literature review

In this section, we briefly review the relevant literature for the transit network design problem. Lampkin and Saalmans proposed the first heuristic algorithm to design a transit route network (Lampkin, 1967). In the first step, the proposed algorithm produced an initial skeleton route. In the next steps, the other nodes were inserted one by one into the skeleton route. The case study of a small town in the North of England was also presented in the paper.

Simman, Barzily, and Passy proposed a two-staged approach for the problem (Simman et al., 1974). They first generated a set of route-candidates through several iterations. The authors determined the optimal vehicle frequencies in the second stage. They tried to minimize passengers travel time, while simultaneously taking care of the total number of passengers who cannot find seats. Byrne considered the case when the region served by the public transit is a segment of a circle and may be defined in the polar coordinates (Byrne, 1975). He proposed the model of a transit system that is built in the polar coordinates with radial transit lines.

In his pioneering work, Mandl proposed a heuristic algorithm to find the set of the best transit routes (Mandl, 1979). He developed a solution in two stages: first a feasible set of routes was generated, and then heuristics were applied to improve the quality of the initial route set. The route generation phase involved first computing the shortest paths between all pairs of vertices by Dijkstra's algorithm or Floyd–Warshall algorithm (Cormen et al., 1990), and then seeding the route set with those shortest paths that contained the most nodes, respecting the positions of any node designated as terminals. Unserved nodes were then iteratively incorporated into routes in the most favorable way, or new routes created with unserved nodes as route terminals.

In Newell (1979), Newell performed a theoretical analysis of the bus route network design problem. He discussed various aspects of the problem and reached to the following conclusion: " in essence, our conclusion is that it would require a large computer and a vast amount of data to determine even a nearly optimal route geometry".

Ceder and Wilson published a model (Ceder and Wilson, 1986) that focused on two routines for generating and testing candidate route sets: Level 1 considered only the passengers' viewpoint, and was aimed at minimizing the total travel time, while Level 2 considered both passengers' and operators' viewpoint and balanced travel time and waiting time with the number of vehicles required. Vehicle frequencies and timetables were also set at Level 2. The general idea of the route construction algorithms was to start from the terminal nodes having the largest demand and expand the routes incrementally by including more nodes.

Baaj and Mahmassani described and implemented a heuristic route generation algorithm for the route network design (Baaj and Mahmassani, 1995). Generally it determined an initial set of skeletons and expanded them to form transit routes, which heavily depended on the travel demand matrix. In this algorithm, the designers knowledge and experience were also used to reduce the search space.

Ceder and Israeli defined an objective function that takes into account the interests of both passengers and operators (Ceder and Israeli, 1998). The proposed model for the transit network design problem combines mathematical programming, and decision-making techniques.

The last two decades have seen a rapid growth in computing power and, as computers have become faster, metaheuristic techniques have become ever more popular for solving hard combinatorial problems. Methods such as genetic algorithms, tabu search and simulated annealing have all played important roles in recent research on transit network design.

When solving the bus route network design problem, Pattnaik, Mohan, and Tom proposed a two step procedure (Pattnaik et al., 1998). They generated the set of the route candidates in the first step. In the second step, the authors decided about the final set of routes by using the genetic algorithms. They used a binary encoding scheme to identify candidate routes. In general, their initial candidate route sets were produced using heuristic procedures, applying shortest path calculations moderated by user-defined guidelines. The genetic operators, mutation and crossover, produced new route set variations for selection, giving the population scope to improve over time, provided selection is biased towards saving the better solutions over the poorer ones.

On the other hand, Chakroborty and Dwivedi took a different approach for encoding-by listing the nodes explicitly, rather than binary coding a route as an entity in their genetic algorithm (Charkroborty and Dwivedi, 2002). This work was enhanced further by Chakroborty to cover scheduling as well as routing (Chakroborty, 2003).

Bielli, Caramia, and Carotenuto applied genetic algorithm approach when considering bus network optimization problem in Bielli et al. (2002). They tested their approach for the city of Parma, Italy. Lee and Vuchic considered the transit network design problem in the case of a variable transit demand, under a given fixed total demand (Lee and Vuchic, 2005). The authors offered an iterative approach that takes care about the relationship between the variable transit trip demand and the transit network design. The proposed approach was tested on a relatively small transit network.

Guan, Yang, and Wirasinghe proposed the model for simultaneous optimization of transit line configuration and passenger line assignment (Guan et al., 2003). The proposed model was solved by the branch and bound method. Fan and Machemehl used the simulated annealing technique to solve the optimal bus transit route network design problem (Fan and Machemehl, 2006). The proposed concept was tested on three experimental networks. Zhao and Zeng combined genetic algorithm and simulated annealing while searching for the optimal route structures and headways (Zhao and Zeng, 2006). The authors tried to minimize transfers and total user cost, and to maximize service coverage. Subsequently, Zhao and Zeng developed the model for route network design, vehicle headways, and timetable assignment. The proposed approach combined simulated annealing, and tabu search (Zhao and Zeng, 2007).

Fan and Machemehl considered the design of public transportation networks in the case of variable demand (Fan and Machemehl, 2008). The authors developed a multi-objective model. Their solution methodology was based on the tabu search method.

Guihaire and Hao classified 69 various approaches dealing with the transit network design and frequency setting (Guihaire and Hao, 2008). When solving route design and bus assignment problem, Pacheco et al. developed an algorithm based on a local search strategy, as well as an algorithm based on a tabu search strategy (Pacheco et al., 2009). The authors showed the robustness of their approach with respect to variations in demand. The case study of the city of Burgos, Spain was presented in the paper.

Mauttone and Urquhart developed Pair Insertion Algorithm (PIA) that can be used to generate initial solutions for a local improvement or evolutionary algorithm (Mauttone and Urquhart, 2009). The algorithm is inspired by the route generation algorithm (RGA) of Baaj and Mahmassani (1995).

Fan and Mumford proposed a model of the urban transit routing problem that evaluated candidate route sets (Fan and Mumford, 2010). The proposed approach used hill-climbing and simulated annealing techniques. Bagloee and Ceder studied the design of a transit network for the actual-size road networks (Bagloee and Ceder, 2011). The proposed algorithm was tested on the network of the city of Winnipeg, Canada, as well as on the transit network of Mandl benchmark.

Szeto and Wu studied the bus network design problem in the case of Tin Shui Wai, a suburban residential area in Hong Kong (Szeto and Wu, 2011). The authors proposed the model that simultaneously performed the route design and bus frequency setting. The proposed solution method represented the combination of the genetic algorithm and a neighborhood search heuristic.

Miandoabchi et al. studied the design of urban road and public transit networks (Miandoabchi et al., 2012). The proposed multicriteria model considered construction of new roads, adding lanes to the existing roads, lane allocation in two way streets, and the orientation of the one way streets. At the same time, the model was used to propose new routes of a given bus routes.

Blum and Mathew studied the transit route network redesign problem. The proposed approach was tested in the case of city of Mumbai, India (Blum and Mathew, 2012).

A few review and survey papers have also been published in the literature that documents and discusses the results in the literature from different perspectives. Kepaptsoglou and Karlaftis presented and reviewed research results in the area of transit route network design problem (Kepaptsoglou and Karlaftis, 2009). Design objectives, operating environment parameters and solution approaches are especially analyzed in the paper. The review paper of Derrible and Kenneday is devoted to the applications of the graph theory in transit network design (Derrible and Kenneday, 2011). Schoebel made a review of the various bus, railway, tram, and underground line planning models (Schoebel, 2012).

In summary, we found that the majority of researchers tried to minimize the total travel time, or the generalized cost. Also most of the works introduced simplified assumption about fixed demand for transit services. A more realistic assumption is to consider that the passenger flows depend on the transit network design, and that should be determined as a solution of an equilibrium problem. The decision variables are transit network route configuration and/or bus frequencies. Various papers in the literature also dealt with both type of passengers assignment among possible transit routes: single path assignment and multiple path assignment. Due to the problem hardness and computational complexity, the majority of papers offer heuristic or metaheuristic approach.

3. Statement of the problem

We have treated the Transit Network Design Problem (TNDP) in the similar fashion as in Nikoli and Teodorovi (2013). Here we are given a road network described by the graph $G = (N, A)$, where N is the set of nodes representing the bus stops and A is the set of edges representing the street segments. A route used by the transit passengers is described by a path in the graph. We assume that the given road network is connected and undirected and there are sufficient vehicles on each route to ensure that the demand between every pair of nodes is satisfied.

We have a demand matrix denoted by d_{ij}, which represents the number of trips per time unit between node i and node j. We also denote by D the origin–destination matrix (O–D matrix) as follows:

$$D = \{d_{ij} | i, j \in [1, 2, \ldots, |N|]\} \tag{1}$$

We also know the travel time matrix for the road network denoted by tr_{ij}, which represents the in-vehicle travel time between the node i and the node j. By TR, we denote the travel time matrix:

$$TR = \{tr_{ij} | i, j \in [1, 2, \ldots, |N|]\} \tag{2}$$

The main indicator that we use to describe the level of transit service is the total travel time spent by the users of the transit service. We express the quality of the solution generated in minutes. We calculate the total travel time of all passengers T in the network in the following way:

$$T = TT + w_1 TTR + w_2 TU, \tag{3}$$

where TT is the total in-vehicle time of all served passengers, TTR is the total number of transfers in the network, TU is the total number of unsatisfied passengers (we assume that the passenger is unsatisfied when she/he has to make more than two transfers during the trip), w_1 is the time penalty for one transfer (5 min), and w_2 is the time penalty for one unsatisfied passenger (Optimal Average Travel Time (ATT) + 50 min).

So our objective is to find a set of routes R such that T is minimized.

We also assume that the passengers choose the route based on the shortest travel time principle. Many researchers decomposed the transit network design problem into the following two stages: (1) generation of the set of transit routes and (2) determination of the frequency of service for each generated route. We do the same in this paper, and we focus exclusively on the generation of the set of transit routes.

So the transit network design problem could be defined in the following way: For a given set of n nodes, known origin–destination matrix D that describes demand among these nodes, and known travel time matrix TR, generate a set of transit routes on a network such that the total travel time T of all passengers is minimized.

4. The proposed solution

In this paper, we present Genetic Algorithm with Elitism (GAWE) for solving the problem. Also, we experiment our algorithm by increasing the population size after each generation. The resulting implementation provides the best result which is better than ever before.

Notably Pattnaik, Mohan and Tom proposed a genetic algorithm for the transit network design problem in Pattnaik et al. (1998). Later, Chakroborty and Dwivedi also employed GA to solve the problem at hand using a different approach (Charkroborty and Dwivedi, 2002). While designing a solution, it is very important to make an efficient initialization. Both of the previous GA based approaches used different initial route set up procedure. Here, we use an initial route set up procedure which is more efficient than the previous GA based approaches.

4.1. Genetic algorithm

We have to design suitable initialization and selection procedures, crossover and mutation operators for using GAWE, we discuss these in the following subsections.

4.1.1. Representation

In our problem each individual is a set of paths, known as a route set from the given road network. We represent a route set as vector of lists where each list stores a path as a sequence of nodes. For a road network with 8 nodes denoted by integers from 1 to 8, an individual having 4 routes is shown in Fig. 2.

4.1.2. Initialization

We have constructed initial solution using our greedy algorithm which is a modification of the initialization procedure of Nikoli and Teodorovi (2013). In the initial solution, each route is a shortest path based on the travel time between a selected pair of nodes. Using a greedy algorithm we select the required number of pairs (i, j) with high ds_{ij} values. ds_{ij} is the total number of passengers that enjoy direct services along the shortest path between nodes i, j. So, formally,

$$ds_{ij} = \sum_{m \epsilon N} \sum_{n \epsilon N} d_{mn}, \tag{4}$$

where N is the set of nodes in the shortest path between nodes i and j, d_{mn} is the number of trips per time unit between nodes i and j.

We denote by DS the corresponding matrix that contains information about the number of passengers who are able to enjoy the direct service.

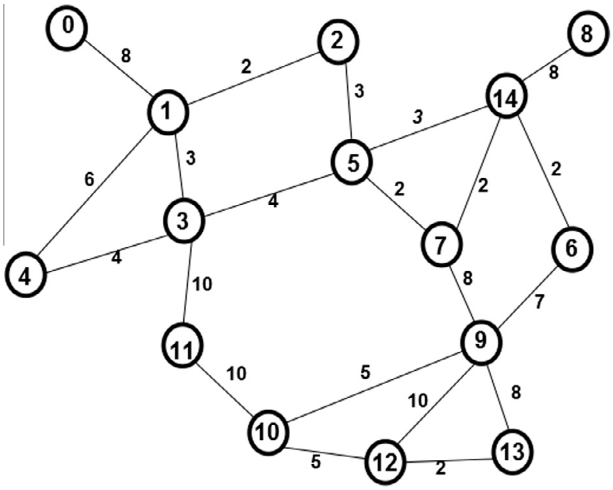

Fig. 1. Mandl's route network.

Index	Route
0	1-2-3-4-5
1	1-2-3-8-7
2	8-7-5-4
3	6-2-1

Fig. 2. An individual with 4 routes.

$$DS = \{ds_{ij} | i, j \epsilon [1, 2, \ldots, |N|]\} \quad (5)$$

The number of routes in an individual is a parameter of the algorithm, namely, *routeSetSize*. At the very beginning all the individuals in the population are identical which is the generated initial solution. The greedy algorithm for generating initial solution from Nikoli and Teodorovi (2013) is summarized as follows:

Step 1: First we decide the total number of routes, N, in the solution. We denote the set of routes by Y. We set $Y = \phi$. Let $m = 0$.
Step 2: Now we find the pair of nodes, say (a, b), that has the highest ds_{ij} value. The nodes a and b become the terminals of the new route. Now we find the shortest path between these two nodes. The nodes that belong to the shortest path represent stations in the routes. We add this route in set Y.
Step 3: We set $m = m + 1$. If $m = N$, we stop.
Step 4: We recalculate the matrix DS after setting all the passenger travel demands that are satisfied by the recently added route to zero. We return to Step 2.

The eight route sets extracted by our greedy initialization procedure from Mandl's 15 node Swiss network is shown in Table 1.

Although it seems contrary to the natural intuition, for this problem, it turns out that the use of identical individuals in the initial population rather than a diverse random population works better. This is verified through a series of experiments

Table 1
The initial route sets obtained by our initialization procedure for Mandl's route network.

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

as will be reported later in the experimental sections. The reason for this apparently strange behavior can be attributed to the following discussion. Note that, our sole objective here is to improve the initial route set gradually using the genetic approach. Although at first our crossover operator makes no change, in a few generations our mutation operator introduces reasonable diversity in the population so as to make our crossover operator effective. We have designed the crossover based on the fact that all the individuals are closely related, i.e. the corresponding genes (paths) selected to be swapped have much similarity. This is to ensure that the crossover continues to make smooth changes to the initial route set maintaining *linkage*. If we start with significantly different individuals in which the corresponding genes are not similar, then after applying crossover the resulting individuals can be weird as there is high risk that the resulting individuals would either lose some important features or some features can be repeated. This might cause sharp changes in the fitness violating *linkage*.

4.1.3. Crossover

We use Uniform Crossover, where we treat each route like a gene. With a small probability P_{swap} we decide at each index of the individual whether we can swap the routes of that position between the two selected parents. For example, if at index 1 a randomly generated number between 0 and 1 is greater than or equal to p then we swap the route at that index between the two parents (see Fig. 3).

As in the initial population all individuals are identical, initially crossover has no impact. We will see that it will play the role gradually.

4.1.4. Mutation

We have designed our mutation in such a way so that most of the time it would create only slight modification. It would allow big jump only occasionally to escape from a local optima.

For doing mutation we would first select a route from the individual according to a certain probability using the roulette wheel selection. Probability of taking route l can be calculated as:

$$p_l = \frac{\frac{1}{ds_{ij}}}{\sum_{q\epsilon L} \frac{1}{ds_{rs}}}, \tag{6}$$

where i and j are the terminals of the route l, L is the route set, r and s are the terminals of the route q, and ds_{ij} is the total number of passengers that can travel without any transfer by using the route l that connects terminals i and j.

From the above equation, we see that we select those routes more that fulfill less demand of the passengers. After selecting the route, with a *high* probability P_{ms} we would allow *small* modification and with a *low* probability $(1 - P_{ms})$ we would allow *big* modification.

Small Modification Here we choose one of the terminals of the selected route randomly. Then we decide to delete this terminal with probability P_{delete}. The rest of the route is kept as it is and thus the old route is shortened. On the other hand if the selected terminal survives (with probability $(1 - P_{delete})$), the old route is expanded as follows: a new bus stop is added to the bus line chosen at random from among the nodes adjacent to the chosen terminal.

Big Modification In this case as well, we choose one of the terminals (say terminal i) of the selected route randomly. Then, we choose a new terminal k using roulette wheel selection with the probability:

$$P_k = \frac{ds_{ik}}{\sum_{r\epsilon N} ds_{ir}} \tag{7}$$

The new route will be the shortest path between terminal i and terminal k (see in Fig. 4).

4.1.5. Selection

We use *Tournament Selection* in our algorithm. In the tournament we select individuals based on *fitness*. As all the individuals in our initial population are identical and diversity is introduced gradually, the fitness values of the individuals can be very close. So we need a selection technique which is sensitive enough to distinguish the individuals of nearly same fitness values. This is the motivation for using Tournament Selection.

4.1.6. The algorithm

With the operators and procedures described above the pseudo code of our GAWE algorithm for the transit network design problem is as follows:

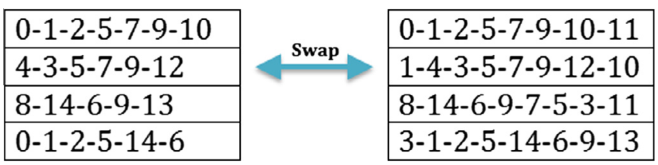

Fig. 3. The entire route at the chosen index will be swapped.

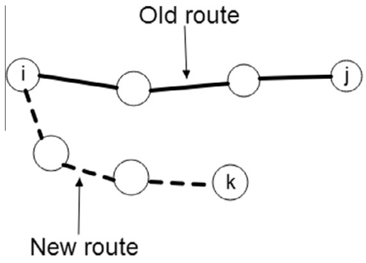

Fig. 4. Big modification.

Algorithm 1. Genetic Algorithm with Elitism (GAWE)

```
popSize ← desired population size
eliteSize ← desired number of eilte individuals
routeSetSize ← number of routes in the individual
t ← tournament size for fitness
P_swap ← probability of swapping an index in Uniform Crossover    ▷low
P_ms ← probability of doing small modification in Mutation    ▷high
P_delete ← probability of deleting the selected terminal in small modification
maxGen ← maximum number of generations
P ← {}
for popsize times do
  P ← P ∪ InitialRouteSet (routeSetSize)
end for
Best ← ∅
repeat
for each individual P_i ∈ P do
  AssessFitness (P_i)
  if Best = ∅ or Fitness (P_i) >Fitness (Best) then
    Best ← P_i
  end if
end for
Q ← the eliteSize fittest individuals in P, breaking ties at random
    for (popSize − eliteSize)/2 times do
    Parent P_a ← TournamentSelection (P)
    Parent P_b ← TournamentSelection (P)
    Children C_a, C_a ← UniformCrossover (Copy (P_a), Copy (P_b))
    Q ← Q ∪ {Mutation (C_a), Mutation (C_b)}
end for
P ← Q
until maxGen is reached
return Best
```

4.2. The GA with increasing population approach to transit network design problem

In this section, we modify GAWE to set a new algorithm called GAWIP. In this version, after we select the *popSize* number of individuals for the next generation through breeding, we also append some high quality individuals of the current generation to the next generation. The number of elite individuals to be appended is predefined and is denoted by *eliteSize*.

5. Experimental results

We conduct three separate sets of experiments. To conduct our experiments, we have used a gcc-g++3.4.5 compiler with ParadisEO2.0 framework in Code::Blocks10.05 IDE. We used a Desktop PC of 3.5 GHz Intel Corei7 processor, with 8 GB RAM. Both the road networks with associated data used in our experiments along with the source code of our experiment is available at http://transit-network-design.googlecode.com/svn/trunk/.

5.1. Mandl's network analysis

Presently Mandl's 15 node Swiss network instance is the only published instance that comes with all necessary information. So, the comparisons of the results are made in the case of Mandl's road network (Mandl, 1979) as shown in Fig. 1. A sample final solution of our GAWIP algorithm for Mandl's road network has been shown in Table 2.

Following the trend of Nikoli and Teodorovi (2013), we have compared the performance of our algorithms with other methods found in the literature in terms of the performance metrics shown in Table 3. At this point a brief discussion on performance metrics of Table 3 is in order. The values of the performance metrics d_0, d_1, d_2 and d_{un} collectively show the quality of a route set. For any route set, the summation of values of these four metrics equals to 1. At first, a route set tries to meet the passenger demand as much as possible without any transfer; then it tries to meet the remaining demand as much as possible with only one transfer; then it tries to do the same for the remaining demand with two transfers; and finally, the rest of the demand are left as unsatisfied. A good route set always tries to meet the most of the passenger demand with the least number of transfers. The effect of the distribution of passenger demand among d_0, d_1, d_2 and d_{un} is summarized by the value of *ATT* as it considers the time wasted by the transfers. Therefore, it shows the overall quality of a route set. So if we have two route sets having very close values of d_0, d_1, d_2 and d_{un}, we can compare them using *ATT*, which should be lower for a better solution.

Following the trend of Nikoli and Teodorovi (2013), we have shown comparisons with our final solution to other previous works considering four situations: 4 routes, 6 routes, 7 routes and 8 routes in each route set. The comparison is shown in Table 4. We can clearly see that the results obtained by both GAWE and GAWIP have better values for d_0, d_1 and d_{un} in all the cases. The GAWE and GAWIP results have better values for d_2 in three out of four cases. But as we have discussed above, the quality of a solution should not be judged solely based on the values of d_2 as d_0, d_1, d_2 and d_{un} altogether judge a solution. The GAWE and GAWIP results have better values for *ATT* in all the cases. As *ATT* gives the overall performance, we can argue that our results are better than all previous results. We have also reported the best, average and worst results of our two approaches over 20 independent runs in Table 5. The parameter values of our Algorithm 1, used for all these experiments are shown in Table 6.

From Table 5, we can see that both the GAWE and GAWIP versions of our algorithm are almost equally capable of finding the best solutions from 20 independent runs. GAWIP can produce more consistent results than GAWE. However, as the population size is increasing at every generation, GAWIP requires more time to produce output than GAWE. This gives us the option to choose between GAWE and GAWIP depending on the specific requirement. If we just need the best result within a short time then GAWE is the best option. Whereas GAWIP can best serve those environments which need stable and consistent results regardless of running time.

Table 2
A sample final solution obtained by GA with increasing population for Mandl's route network.

Situation	Number of routes	Route description
1	4	0–1–2–5–7–9–10–11–3–4
		0–1–4–3–5–7–9–12–13
		8–14–6–9–13–12–10–11–3–1–0
		12–13–9–6–14–5–2–1–0
2	6	0–1–2–5–7–9–10–12
		0–1–4–3–5–7–9–12–13
		4–1–2–5–3–11–10–9–6–14–8
		0–1–2–5–14–6–9–13–12–10–11–3–4
		8–14–5–2–1–0
		0–1–3–11–10–12–13–9–7–14–6
3	7	0–1–2–5–7–9–10–12–13
		10–12–13–9–7–5–3–4–1–2
		13–12–10–9–6–14–7–5–3–4–1–0
		8–14–5–2–1–0
		8–14–5–2–1–3–11–10–12–13–9
		0–1–2–5–14–6–9–13–12–10–11–3–4
		8–14–6–9–10–11–3–1–0
4	8	0–1–2–5–7–9–10–12
		2–1–4–3–5–7–9–10–12
		8–14–5–2–1–0
		8–14–7–5–2–1–3–11–10–9–13–12
		11–10–12–13–9–6–14–7–5–3–4–1–0
		0–1–2–5–14–6–9–13–12–10–11–3–4
		8–14–5–3–11–10–12–13–9–7
		8–14–6–9–10–11–3–1–0

Table 3
Performance metrics.

d_0	The percentage of demand satisfied without any transfers
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 in minutes per transit user (mpu). This travel time includes transfer waiting times, and transfer time that is equal to 5 min per passenger

Table 4
The comparison among the final solutions generated by our approaches and the previous approaches for Mandl's route network. Best results are shown in bold-font. Tied best results are shown in normal-font.

Number of routes	Parameters	Mandl (1979)	Baaj and Mahmassani (1991)	Kidwai (1998)	Charkroborty and Dwivedi (2002)	Fan and Machemehl (2008)	Nikoli and Teodorovi (2013)	GAWE	GAWIP
4	d_0	69.94	N	72.95	86.86	93.26	92.1	**96.14**	95.83
	d_1	29.93	N	26.92	12	6.74	7.19	**3.47**	3.60
	d_2	0.13	N	0.13	1.14	**0**	0.71	0.39	0.57
	d_{un}	0	N	0	0	0	0	0	0
	ATT	12.9	N	12.72	11.9	11.37	10.51	10.49	**10.35**
6	d_0	N	78.61	77.92	86.04	91.52	95.63	98.39	**98.91**
	d_1	N	21.39	19.68	13.96	8.48	4.37	1.61	**1.09**
	d_2	N	0	2.4	0	0	0	0	0
	d_{un}	N	0	0	0	0	0	0	0
	ATT	N	11.86	11.87	10.3	10.48	10.23	10.14	**10.10**
7	d_0	N	80.99	93.91	89.15	93.32	98.52	99.17	**99.55**
	d_1	N	19.01	6.09	10.85	6.36	1.48	0.83	**0.45**
	d_2	N	0	0	0	0.32	0	0	0
	d_{un}	N	0	0	0	0	0	0	0
	ATT	N	12.5	10.69	10.15	10.42	10.15	10.07	10.07
8	d_0	N	79.96	84.73	90.38	94.54	98.97	99.86	**99.87**
	d_1	N	20.04	15.27	9.62	5.46	1.03	0.14	**0.13**
	d_2	N	0	0	0	0	0	0	0
	d_{un}	N	0	0	0	0	0	0	0
	ATT	N	11.86	11.22	10.46	10.36	10.09	**10.03**	10.04

Table 5
Comparison between our two algorithms for Mandl's route network.

Number of route sets	Parameters	GAWE			GAWIP		
		Best	Avg	Worst	Best	Avg	Worst
4	d_0	96.14	93.39	86.96	95.83	93.76	91.84
	d_1	3.47	5.55	8.8	3.60	5.34	7.06
	d_2	0.39	1.06	4.24	0.57	0.90	0.10
	d_{un}	0	0	0	0	0	0
	ATT	10.49	10.5	11.12	10.35	10.45	10.51
6	d_0	98.39	97.50	96.08	98.91	98.08	96.98
	d_1	1.61	2.49	3.92	1.09	1.92	3.02
	d_2	0	0.01	0	0	0	0
	d_{un}	0	0	0	0	0	0
	ATT	10.14	10.17	10.22	10.10	10.14	10.19
7	d_0	99.17	98.35	97.24	99.55	99.09	98.07
	d_1	0.83	1.65	2.76	0.45	0.91	1.93
	d_2	0	0	0	0	0	0
	d_{un}	0	0	0	0	0	0
	ATT	10.07	10.11	10.16	10.07	10.08	10.12
8	d_0	99.87	99.28	98.65	99.87	99.54	99.29
	d_1	0.13	0.72	1.35	0.13	0.46	0.71
	d_2	0	0	0	0	0	0
	d_{un}	0	0	0	0	0	0
	ATT	10.03	10.07	10.1	10.04	10.05	10.06

Table 6
Parameter values used in our experiments.

Parameter	Value
popSize	16
eliteSize	4
t	10
P_{swap}	$\frac{1}{routeSetSize}$
P_{ms}	0.7
P_{delete}	0.4
maxGen	400

5.2. Real bus network analysis

In Mumford (2013a), author performed three experiments on real life data sets namely Mumford1, Mumford2 and Mumford3 based on information manually extracted from bus route network maps for real cities: one in China (Yubei) and two in the UK (Brighton and Cardiff), respectively (see Table 10). But she applied biobjective optimization technique. However, one of her objective, average passenger travel time, incorporating in-vehicle travel time and transfer penalty, matches our objective. She gave separate results for her two objectives which gives us a ground to compare our results with her. We have shown the comparative results with our GAWE Algorithm in Table 11. All the results, shown in Table 11 are the average values over 20 independent runs each with 200 generations. It also shows that our results are better. Sample final solutions of our GAWE algorithm for data sets Mumford1, Mumford2 and Mumford3 have been shown in Tables 7–9 respectively.

Table 7
A sample final solution for Mumford1 network by GAWE algorithm.

Route	Route description
Route 1	12–25–27–41–34–51–56–6–21–63–24–30–26–16–10–17–50–2–0–69–48–29–3–19
Route 2	62–42–30–15–49–26–8–51–41–36–54–18–39–48–66–69–23–0–50
Route 3	58–35–6–21–24–62–42–49–26–16–10–20–33–22–19–66–29–23–2–14–44
Route 4	2–14–44–60–61–43–33–58–57–51–56–28–10–20–55–16–1–11–68
Route 5	41–51–64–52–8–55–17–44–60–61–43–13–22–36–54–19–69–0–50–33–58–35
Route 6	53–4–40–46–7–47–6–56–8–26–1–68–9–31–11
Route 7	8–10–17–60–44–61–14–2–50–33–58–41–12–40–46–53–47–7
Route 8	23–3–66–18–65–67–27–12–40–53–7–4–25–34–58–33–50–2–14–61–17–60
Route 9	63–24–42–49–11–9–17–20–33–22–19–18–32–37–5–65–39–48
Route 10	57–58–64–52–10–16–9–31–1–26–8–51–41–27–67–37–65–19–38–66
Route 11	56–28–8–26–62–15–49–1–59–17–50–23–3–48–29–19–54–65–67
Route 12	69–29–3–38–22–19–65–37–5–54–36–45–58–64–52–20–55–8–51–35–34–57–25–41
Route 13	29–69–66–19–54–32–65–39–48–3–23–50–17–9–68–11–15–62–42
Route 14	1–26–62–21–30–24–63–6–35–41–36–19–66–48–3–0
Route 15	33–58–51–35–34–45–22–13–50–0–23–69–38–36–41–12–4–53–47–40

Table 8
A sample final solution for Mumford2 network by GAWE algorithm.

Route	Route description
Route 1	4–77–5–42–57–7–41–47–58
Route 2	17–56–61–0–80–38–53–90–49–52–89–11–5–77
Route 3	46–21–27–101–55–45–53–62
Route 4	50–95–27–94–83–15–18–92–48–107–77–4
Route 5	48–79–108–5–11–89–52–91–62–12–74–47–41
Route 6	60–47–12–8–35–34–3–19–33–99–69
Route 7	69–99–33–19–3–34–35–8–12–74
Route 8	42–57–10–52–49–8–63–55–39–17–75–50
Route 9	29–26–44–92–18–93–35–8–62–103–41–16–70–84
Route 10	84–7–37–57–10–52–49–35–83–94–27–59–68–50–32
Route 11	16–12–8–35–93–9–23–82–99
Route 12	95–68–20–17–39–55–63–8–49–52–89–73–106
Route 13	29–30–26–82–19–85–94–27–17–65–61–1
Route 14	32–20–17–78–39–55–45–2–15–18–92–64–79–48
Route 15	12–60–74–38–55–101–27–21–86–71
Route 16	16–60–12–63–2–34–93–100–52–49–7
Route 17	81–86–71–102–24–54–15–35–8–12–41
Route 18	25–7–40–10–52–100–93–3–6–86–46
Route 19	65–78–39–55–63–8–49–52–89–73–42–5

(*continued on next page*)

Table 8 (*continued*)

Route	Route description
Route 20	28–59–32–13–21–86–24–19–82–44–48–107–77–4
Route 21	1–87–38–53–90–49–52–89–64–36–4–107
Route 22	1–0–78–95–59–21–86–51–102–76–69–30
Route 23	84–7–52–100–18–9–19–102–88–71
Route 24	46–28–86–24–19–82–44–72–79–36
Route 25	103–7–40–57–106–73–79–72–44–82–99–76–31
Route 26	46–28–13–32–17–39–55–63–12–41–104–70
Route 27	7–70–10–50–52–100–93–3–22–85–66–34–35
Route 28	102–76–31–24–6–94–101–55–39–87–1–97
Route 29	99–14–69–43–72–79–73–106–57–37–104
Route 30	21–46–86–6–54–15–35–49–7–104–37
Route 31	43–29–30–69–99–33–19–3–15–2–45–80–97
Route 32	37–40–10–52–100–18–9–19–33–14–102
Route 33	102–76–33–19–9–18–98–93–35–90–91–62
Route 34	10–37–84–25–70–103–12–63–55–39–17–50–68
Route 35	4–36–72–44–82–33–31–51–46–21–32
Route 36	41–70–10–52–100–18–9–23–82–26–30
Route 37	13–32–27–94–83–15–18–92–64–109–73–89
Route 38	44–30–43–72–79–11–89–52–7–41–60
Route 39	64–36–108–72–92–18–66–34–35–91–100
Route 40	11–5–36–72–44–82–19–22–6–85–54
Route 41	9–23–31–22–54–15–2–45–38–58
Route 42	84–25–7–49–35–83–94–27–21–28
Route 43	83–34–93–67–18–15–2–45–55–39–65–61
Route 44	11–73–42–96–64–92–9–3–81–94–105–27
Route 45	80–58–60–12–62–91–52–89–64–36–107–48–72–26
Route 46	61–0–87–45–2–15–18–92–48–107–36
Route 47	75–20–95–27–94–85–19–82–26–30–44
Route 48	46–51–31–33–82–26–30–43–29
Route 49	16–47–58–80–55–101–27–21–86–51–88–71
Route 50	76–69–99–33–19–85–94–101–55–87–56–39–65
Route 51	10–40–25–70–16–74–58–97–0–61–1
Route 52	57–106–11–64–92–9–19–102–71–51
Route 53	36–5–11–89–52–49–35–83–94–27–32–13–21
Route 54	5–77–107–48–44–82–33–31–88–71
Route 55	10–25–7–52–89–11–79–72–43–29–26–82
Route 56	21–68–59–27–94–83–35–49–52–89–106–42

Table 9
A sample final solution for Mumford3 network by GAWE algorithm.

Route	Route description
Route 1	75–60–73–48–0–11–1–118–52–89–17–64–111
Route 2	10–86–24–5–9–72–49–2–48–73–60–32–20
Route 3	110–87–24–5–29–27–67–46–118–122
Route 4	19–20–32–60–73–48–2–49–72–9–5–24–86–109–4
Route 5	110–87–61–53–23–2–79–0–101–88–91–123–125–121
Route 6	4–64–28–103–109–86–24–87–84–68–12
Route 7	117–10–86–24–5–29–27–1–11–0–120–26–7
Route 8	32–19–50–40–82–88–39–31–89–113–98–17
Route 9	76–50–40–82–88–39–31–89–42–43–103–18
Route 10	21–22–30–39–46–11–2–114–78–3–56–34
Route 11	68–34–56–3–78–114–2–79–0–101–88–121–125
Route 12	15–3–100–14–66–80–29–122–52–51–54–30–31
Route 13	18–104–59–89–31–39–88–82–40–50–83
Route 14	85–19–13–50–40–7–11–1–27–29–5–24–117–86–18
Route 15	92–76–50–40–82–88–39–31–89–42–43–109–69
Route 16	62–16–114–23–53–61–102–24
Route 17	86–10–69–43–42–89–31–30–22–119
Route 18	109–69–111–64–43–105–63–80–66–74–114–16–112–60
Route 19	113–17–42–105–58–5–24–87–84–68–34
Route 20	56–84–87–24–5–29–122–52–38–30–22–119–8
Route 21	104–58–63–80–81–1–77–11–97–37–2
Route 22	62–112–6–73–48–26–101–88–125–21–119
Route 23	13–95–82–88–39–31–89–59–104–18–117–86
Route 24	84–87–61–14–66–124–72–23–37–74–49
Route 25	82–91–25–39–46–1–27–29–5–24–102–106–61
Route 26	14–108–87–102–61–53–23–2–48–73–60–45

Table 9 (*continued*)

Route	Route description
Route 27	30–8–41–25–88–101–26–48–73–6–112
Route 28	18–117–24–5–9–72–49–2–48–126–92–76–50
Route 29	102–108–106–14–53–23–2–11–46–39–30–22–119–8
Route 30	106–14–53–23–2–11–46–39–25–41–8–35
Route 31	12–107–3–100–14–66–80–63–105–42–113–90–103–98
Route 32	110–68–3–78–114–16–55–60–32–20–85–19
Route 33	68–56–15–3–78–114–2–79–0–101–88–91–35
Route 34	34–68–84–87–24–5–58–104–59–113–90–43–17
Route 35	107–12–78–114–2–11–7–65–44–82
Route 36	17–59–104–58–63–9–72–49–2–97–48–79–11–67
Route 37	104–105–58–63–80–81–1–11–7–93–82–116
Route 38	32–45–73–48–2–49–72–9–63–105–43–69
Route 39	4–43–105–63–9–72–49–2–48–73–6–32–126
Route 40	84–87–61–53–23–2–79–0–120–44–40–65–95–123
Route 41	35–91–88–101–0–79–2–49–72–70–80–66
Route 42	18–10–86–24–5–80–66–74–114–16–62–112
Route 43	13–19–85–92–40–82–88–39–31–51–47–38–25
Route 44	6–60–32–126–26–11–1–118–52–89–42–90–103
Route 45	42–98–43–104–58–5–24–102–61–115–3–71–96
Route 46	103–28–43–104–58–36–5–29–94–27–118–52–31
Route 47	54–33–39–46–11–0–48–99–55–60–75
Route 48	28–111–64–17–89–31–39–88–93–101–116–40–26
Route 49	41–8–22–125–91–82–40–50–20–75–85–76
Route 50	45–55–16–114–74–66–80–5–24–86–109
Route 51	32–75–19–13–83–92–126–48–2–23–53–61–87–84–110–68
Route 52	10–4–43–42–89–31–39–88–82–40–50–76–20
Route 53	38–54–31–47–51–52–118–1–11–2–114–16–62
Route 54	125–123–35–91–88–101–26–48–99–55–16–62
Route 55	15–56–12–100–14–66–80–29–94–81–1
Route 56	111–28–17–89–31–39–88–82–40–92–85
Route 57	111–103–69–10–86–24–102–61–14–100–78–96–71
Route 58	119–22–30–39–46–11–2–74–23
Route 59	8–57–30–88–101–26–126–32–75
Route 60	8–21–35–91–25–38–31–89–59–104–58–63–5

Table 10
Properties of Real Data Sets.

Data set	Location	Number of nodes	links	Number of Routes
Mumford1	Yubei	70	210	15
Mumford2	Brighton	110	385	56
Mumford3	Cardiff	127	425	60

Table 11
Comparison of our results with Mumford (2013a).

Parameters	Mumford1		Mumford2		Mumford3	
	Mumford	GAWE	Mumford	GAWE	Mumford	GAWE
d_0	36.60	37.71	30.92	32.53	27.46	29.15
d_1	52.42	56.37	51.29	63.53	50.97	64.31
d_2	10.71	5.88	16.36	3.93	18.76	6.5
d_{un}	0.26	0	1.44	0	2.81	0
$ATT(mins)$	24.79	23.96	28.65	26.63	31.44	29.65

Visual representations of these data sets borrowed from Mumford (2013a) have been shown in Figs. 8–10. For convenience, the Euclidean distances between the generated nodes correspond to the travel time in minutes. Here, the parameter values used in our algorithm is same as Table 6 except *maxGen*, which is 200 for these experiments.

In Nikoli and Teodorovi (2013), the authors performed a second experiment for the bus network containing 110 nodes and 275 links. This network was randomly generated by the authors of Mumford et al. (2009). Although we have done experiments on that network and have found good results, we do not report those results in this paper since this dataset has been reported to be unreliable (Mumford, 2013b).

5.3. Different individuals in the initial solution

As has been discussed above, we have used identical individuals to form the initial population. We conducted some experiments to measure its effectiveness against a more intuitive random diverse population. We produce a random initial population as follows. We create some paths of varied lengths from the input, i.e., the Mandl graph (see Fig. 1) and put it in a file. Then we take more than *routeSetSize* number of paths from that file. Before adding *popSize* number of individuals to form the initial population, we randomize the positions of the routes.

We present some comparison in Figs. 5–7. In this figure, *identical* represents the algorithm that used identical individuals in the initial solution. On the other hand *Randomi*, where $1 \leqslant i \leqslant 8$ represent 8 different versions of random diverse initial

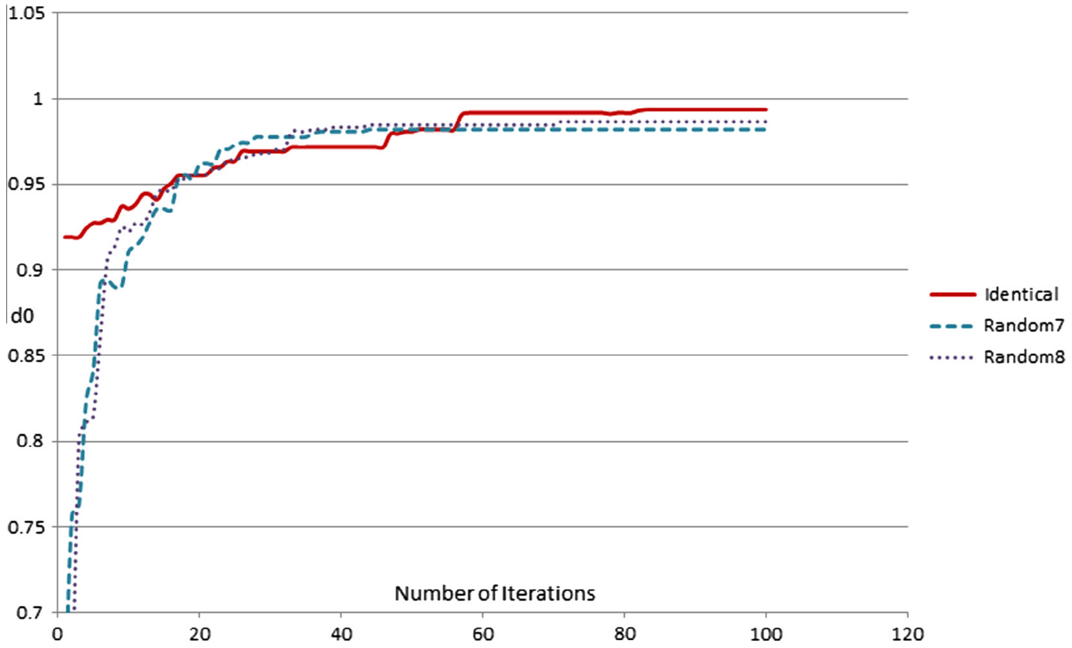

Fig. 5. Comparison of (identical individuals vs. random individuals) in initial solution for 100 generations.

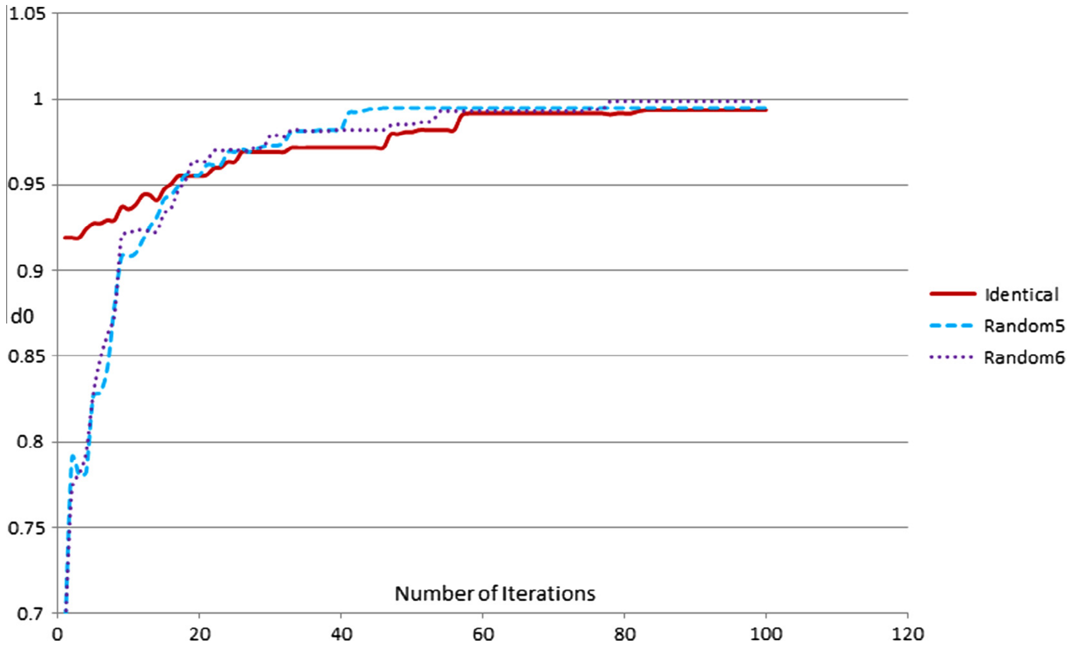

Fig. 6. Comparison of (identical individuals vs. random individuals) in initial solution for 100 generations where random seems better.

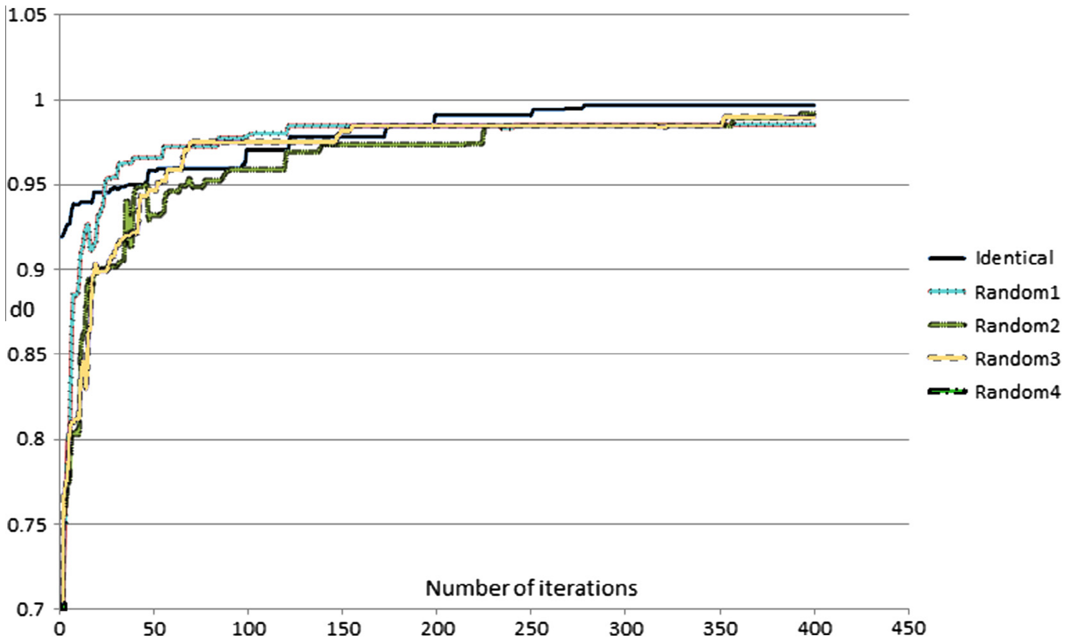

Fig. 7. Comparison of (identical individuals vs. random individuals) in initial solution for 400 generations.

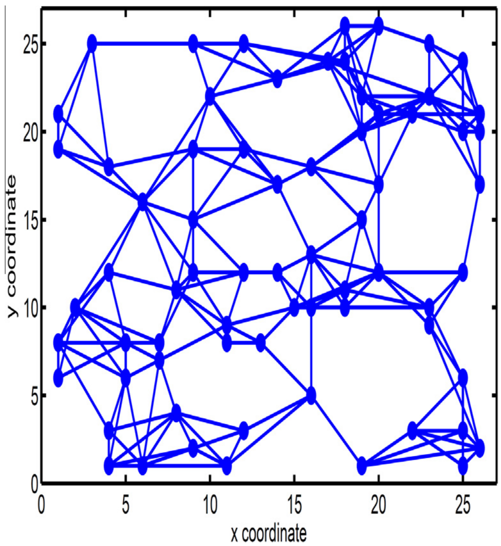

Fig. 8. Transport network for Mumford1.

population. In what follows we will abuse the notation a bit and may use a phrase like ‘value of identical’ (‘value of random’) to indicate the value of d_0 computed by the algorithm with identical (random) initial population.

In Fig. 5, we see that the initial solutions (i.e., solutions from the initial runs) with random individuals have lower values of d_0 than the values of identical. Then, the values of random increase rapidly. Sometimes, these values cross the same for identical and continue progression. But in Fig. 6, we see a different case. Here, the initial solutions with random individuals have lower values of d_0 than that of identical. Then, the values of random increase rapidly. At one point, these values crosse the values of identical. But unfortunately, after certain number of generations, it gets trapped in no progression i,e. local optima. However, the values of identical progress smoothly. Notably, we have conducted these two experiments for 100 generations. Finally, in Fig. 7, we see that at one point, the values of random cross the values of identical. But after 400 gener-

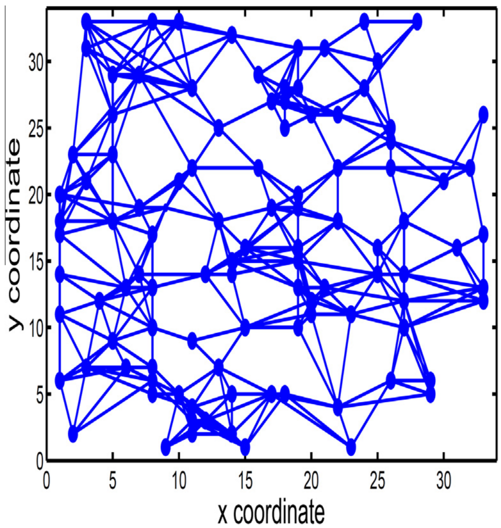

Fig. 9. Transport network for Mumford2.

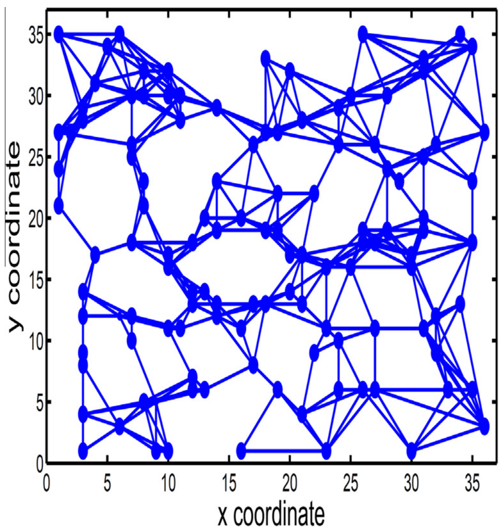

Fig. 10. Transport network for Mumford3.

ations, identical wins. At this point the readers are kindly referred back to our brief discussion in Section 4.1.2 regarding a rationale behind such results of identical and random.

6. Conclusions

The transit network design problem is a large combinatorial problem whose optimal solution is difficult to find. Therefore, heuristic approaches have been used extensively in the literature to solve this problem. In this paper, we have developed two versions of GA based metaheuristic for the transit network design problem. Our first algorithm, namely GAWE, is competitive with other approaches in the literature, and it can generate high-quality solutions within reasonable CPU times. Then, GA with increasing population, beats all the previous results in all cases.

We can add many other objectives for future research to this multi-objective transit network design problem such as traffic jam and performance of vehicles. Our future plan is to extend our study to even larger problems, and incorporate operator costs into our model, by making some simple assumptions regarding service frequencies.

References

Baaj, M.H., Mahmassani, H.S., 1991. An AI-based approach for transit route system planning and design. J. Advan. Transport. 25 (2), 187–210.
Baaj, M.H., Mahmassani, H.S., 1995. Hybrid route generation heuristic algorithm for the design of transit networks. Transport. Res. C 3 (1), 31–50.
Bagloee, S.A., Ceder, A., 2011. Transit-network design methodology for actualsize road networks. Transport. Res. B 45, 1787–1804.
Bielli, M., Caramia, M., Carotenuto, P., 2002. Genetic algorithms in bus network optimization. Transport. Res. Part C: Emerg. Technol. 10 (1), 19–34.
Blum, J.J., Mathew, T.V., 2012. Implications of the computational complexity of transit route network redesign for metaheuristic optimization systems. IET Intell. Trans. Syst. 6, 124–131.
Byrne, B.F., 1975. Public transportation line positions and headways for minimum user and system cost in a radial case. Transport. Res. 9, 97–102.
Ceder, A., Israeli, Y., 1998. User and operator perspectives in transit network design. Transport. Res. Rec. 1623, 3–7.
Ceder, A., Wilson, H.M., 1986. Bus network design. Transport. Res. 20B (4), 331–334.
Chakroborty, Partha, 2003. Genetic algorithms for optimal urban transit network design. Comp.-Aided Civil Infrastruct. Eng. 18, 184–200.
Chakroborty, P., 2003. Genetic algorithms for optimal urban transit network design. Comp.-Aided Civil Infrastruct. Eng. 18, 184–200.
Charkroborty, P., Dwivedi, T., 2002. Optimal route network design for transit systems using genetic algorithms. Eng. Optim. 34 (1), 83–100.
Cormen, T.H., Leiserson, C.E., Rivest, C.E., Stein, C., 1990. Introduction to Algorithms. MIT press (ISBN:978-0-262-03384-8).
Derrible, S., Kenneday, C., 2011. Applications of graph theory and network science to transit network design. Transp. Rev. 31, 495–519.
Fan, W., Machemehl, R.B., 2006. Optimal transit route network design problem with variable transit demand: genetic algorithm approach. J. Transport. Eng. – ASCE 132, 40–51.
Fan, W., Machemehl, R.B., 2008. Tabu search strategies for the public transportation network optimizations with variable transit demand. Comp. Aided Civil Infrastruct. Eng. 23, 502–520.
Fan, L., Mumford, C.L., 2010. A metaheuristic approach to the urban transit routing problem. J. Heurist. 16, 353–372.
Guan, J.F., Yang, H., Wirasinghe, S.C., 2003. Simultaneous optimization of transit line configuration and passenger line assignment. Transport. Res. Part B 40 (10), 885–902.
Guihaire, V., Hao, J.K., 2008. Transit network design and scheduling: a global review. Transport. Res. A 42, 1251–1273.
Kepaptsoglou, K., Karlaftis, M., 2009. Transit route network design problem: review. J. Transport. Eng. – ASCE 135 (8), 491–505.
Kidwai, F.A., 1998. Optimal Design of Bus Transit Network: A Genetic Algorithm Based Approach. Ph.D. dissertation, Indian Institute of Technology, Kanpur, India.
Lampkin, Saalmans, 1967. The design of routes service frequencies and schedules for municipal bus undertaking: a case studies. Operat. Res. Quart. 18 (4), 375–397.
Lee, Y.J., Vuchic, V.R., 2005. Transit network design with variable demand. J. Transport. Eng., ASCE 131, 1–10.
Mandl, C.E., 1979. Evaluation and optimization of urban public transportation network. Euro. J. Operat. Res. 5, 396–404.
Mauttone, A., Urquhart, M.E., 2009. A route set construction algorithm for the transit network design problem. Comp. Operat. Res. 36, 2440–2449.
Miandoabchi, E., Farahani, R.Z., Dullaert, W., Szeto, W.Y., 2012. Hybrid evolutionary metaheuristics for concurrent multi-objective design of urban road and public transit networks. Netw. Spat. Econ. 12, 441–480.
Mumford, C.L., 2013a. New heuristic and evolutionary operators for the multi-objective urban transit routing problem. Evolut. Comput. 939–946 ((CEC), Cancun).
Mumford, C.L., 2013b. Private communication.
Mumford, C.L., Evans, D., 2009. A simple multi-objective optimization algorithm for the urban transit routing problem. Evolut. Comput., 1–7 (Trondheim).
Newell, G., 1979. Some issues relating to the optimal design of bus routes. Transport. Sci. 13, 20–35.
Nikoli, M., Teodorovi, D., 2013. Transit network design by Bee colony optimization. Exp. Syst. Appl. 40 (15), 5945–5955.
Pacheco, J., Alvarez, A., Casado, S., Gonzalez-Velarde, J.L., 2009. A tabu search approach to an urban transport problem in Northern Spain. Comp. Operat. Res. 36, 967–979.
Pattnaik, S., Mohan, S., Tom, V., 1998. Urban bus transit route network design using genetic algorithm. J. Transport. Eng. 124 (4), 368–375.
Schoebel, A., 2012. Line planning in public transportation: models and methods. OR Spect. 34, 491–510.
Simman, L.A., Barzily, L.A., Passy, U., 1974. Planning the route system for urban buses. Comp. Operat. Res. 1, 201–211.
Szeto, W.V., Wu, Y., 2011. A simultaneous bus route design and frequency setting problem for Tin Shui Wai, Hong Kong. Euro. J. Operat. Res. 209, 141–155.
Zhao, F., Zeng, X., 2006. Simulated annealing-genetic algorithm for transit network optimization. J. Comp. Civil Eng. 20 (1), 57–68.
Zhao, F., Zeng, X., 2007. Optimization of transit route network, vehicle headways and timetables for large-scale transit networks. Euro. J. Operat. Res. 186, 841–855.