Evaluating the performance of a genetic algorithm to solve the line planning problem for a bus service

Elina Avila-Ordóñez[1,2], *Chris M.J. Tampère*[1], *Pablo Vanegas*[2], *Pieter Vansteenwegen*[1]

[1] KU Leuven, Leuven Mobility Research Centre-CIB. Celestijnenlaan 300, 3001, Leuven, Belgium.
[2] Department of Computer Science, University of Cuenca. 12 de Abril Av., 010150, Cuenca, Ecuador.

Autores para correspondencia: elina.avilao@ucuenca.edu.ec

Fecha de recepción: 15 de julio de 2017 - Fecha de aceptación: 20 de septiembre de 2017

ABSTRACT

Planning a bus service requires to explore several feasible solutions attempting to optimize travel time, costs or both. The line planning problem (*lpp*) solves the combinatorial problem to define the routes for bus lines in a bus service under a set of constraints, input parameters and an objective function. The input parameters such as the demand, infrastructure, travel times, etc., describe the current situation, and provide both input data and the constraints that should be considered during the design. An algorithm that obtains feasible and high-quality solutions for *lpp* is essential in search of better urban services. In this study, a genetic algorithm is designed and coded to solve the *lpp*. Finally, an evaluation of the results is carried out from different perspectives, attempting to ensure the solutions obtained by the algorithm are consistent and therefore useful in practice.

Keywords: Genetic algorithm, line planning problem, bus services.

RESUMEN

La planificación de un servicio de buses requiere explorar varias soluciones factibles que intenten optimizar el tiempo de viaje de los pasajeros, los costos de los operadores, o ambos. El problema de planificación de líneas de buses (line planning problem en inglés, *lpp*) es un problema combinatorio que define las rutas para las líneas de un servicio de buses bajo un conjunto de restricciones, parámetros de entrada y una función objetivo. Los parámetros de entrada, como la demanda, infraestructura, tiempos de viaje, etc., describen la situación actual y proporcionan datos iniciales y restricciones que deben considerarse durante el diseño. Un algoritmo que provea de soluciones factibles y de alta calidad para *lpp* es esencial para un análisis más profundo en busca de mejores servicios urbanos. En este estudio, se diseña y codifica un algoritmo genético para resolver el *lpp*. Por último, una evaluación de los resultados se realiza desde diferentes perspectivas, intentando asegurar que las soluciones obtenidas por el algoritmo sean consistentes y, por tanto, útiles en la práctica.

Palabras clave: Algoritmo genético, problema de planificación de línea, servicios de autobús.

1. INTRODUCTION

The planning process for public transport is typically reported in the literature as a sequence of stages starting by line planning, followed by the planning of frequencies, and the sequential planning of the timetable, vehicle and crew schedules (Ceder & Wilson, 1986). The line planning problem (*lpp*) solves the problem of defining the line plan (set of bus lines, their routes and stops) including frequencies (Schöbel, 2012). The line planning process typically assumes that the road network, the location of the stops and the demand for transportation are given.

Mandl (1979) defined a method composed of two phases to solve the *lpp*: the first one starts creating a set of candidate bus lines, and in the second phase the optimal set is found by using a heuristic or metaheuristic. Research on similar methods to solve the *lpp* can be found in Pattnaik, Mohan, & Tom (1998), Chakroborty & Wivedi (2002), Ngamchai & Lovell (2003), Tom & Mohan (2003), Zhao & Zeng (2006), Cipriani, Gori, & Petrelli (2012), Nayeem, Rahman, & Rahman (2014), Zhao, Xu, & Jiang (2015). These studies evaluated the performance of the algorithm mainly by comparisons between their results and others reported in the literature. The focus is the efficiency of the algorithm but the evaluation of other aspects, such as the influence of using random numbers during calculations, and the fact that changes on the input parameters (i.e., number or maximal length of bus lines) should lead to expected results, are not considered.

To design a line plan for a certain region, some data must be provided, the streets available for public transport, the expected number of passengers moving from one point of the city to another, the fleet size, etc., are some examples of the required information to plan bus lines. These details are called *input parameters* in the literature (Guihaire & Hao, 2008; Ibarra-Rojas, Delgado, Giesen, & Muñoz, 2015). Although each of these parameters can contribute to achieve better solutions, it is not necessary to use them all. It is possible to establish, based on the specific goals of the bus service to be designed, which parameters are relevant for each case.

The *lpp* is typically solved considering two different perspectives: minimization of the cost to operate the bus service (the operator's perspective) and maximization of the experience of the passengers (the passenger's perspective) (Kepaptsoglou & Karlaftis, 2009; Schöbel, 2012; Farahani, Miandoabchi, Szeto, & Rashidi, 2013; Ibarra-Rojas *et al.*, 2015). During the quality evaluation of a line plan, the travel time is one of the most appreciated user indicators (Van Oort, 2011). Users strongly prefer short travels, but also more predictable travel times, as reported in a review on the subject (Carrión & Levinson, 2012). Obviously, many researchers try to minimize the total travel time (*ttt*) when designing a line plan (e.g., Baaj & Hani, 1991; Chakroborty & Wivedi, 2002; Borndörfer, Grötschel, & Pfetsch, 2008; Zhao *et al.*, 2015). The *ttt* is the summation of riding and waiting times related to a bus service. Especially for problems with a user's perspective the design of a line planning attempts to minimize the *ttt*.

The most popular metaheuristics to solve the *lpp* are genetic algorithms (GA) and simulated annealing (Farahani *et al.*, 2013). In our study we designed a passenger-oriented GA to solve the *lpp*. A GA mimics the principles of natural evolution, such as variation, selection, recombination and mutation to exchange attributes between two or more good solutions. The basic concepts of GA were developed by Holland (1975) and Goldberg (1989). Unlike other studies, the evaluation takes care of thoroughly analyzing the identified bus lines. The solutions obtained by the GA must differ according to some change in the conditions of the problem (e.g., changes in the values of the input parameters). This difference should be significantly larger than differences caused by the randomness of the GA.

The applied evaluation method has three stages: 1) benchmark comparisons, 2) analysis of the influence of random numbers used in the algorithm; and 3) analysis of expected results due to changes in the input parameters values. After the three-stage analysis, the study concludes if the GA developed is a worthwhile tool and consequently, the line plans calculated by this GA are considered good enough to be studied during decision making for better urban services. Results might advise to use the tested GA as a part of a larger research on improving the bus service for the city of Cuenca and designing flexible line plans.

In the following sections details about the passenger-oriented GA to solve the *lpp* are delivered. The selection of input parameters is discussed at the beginning, and then the generation of the pool of lines and the fitness value calculations are described. Later, the set of genetic operators, specially developed for this specific problem, are described in full. A small-size benchmark network (Mandl, 1979) of 15 bus stops, 21 undirected links and a demand of 15,570 travels, is used in this study. Finally, the evaluation results, conclusions and future work are presented.

2. PASSENGER ORIENTED GENETIC ALGORITHM

The passenger-oriented GA designed in this study returns a line plan minimizing the *ttt* for a given situation. The value of *ttt* is considered as nominal because it is calculated based on the line plan only. In this study, the real waiting and walking times resulting from the timetable and transfers are not measured. We also assume that all demand must be served, with or without transfers. The vehicles are considered large enough, so no capacity restrictions are needed. This corresponds to if the frequency at certain lines can be increased if the vehicles would be too small. Furthermore, we assume that all passengers will take the fastest path, based on travel times (including transfer time if needed), given the developed line plan.

Algorithm 1 presents the general GA procedure to solve the *lpp*. In the first stage the input parameters set the situation for which the bus service will be designed. Later the algorithm creates a set of bus lines that meet the constraints and assumptions. That set is the input to create several line plans that will be part of the initial population. Every population generated by the GA, corresponds to a set of feasible line plans. Having an initial population, the GA starts the evolution process: 1) the *ttt* (fitness value in *lpp*) is calculated for each individual in the population and the best line plan is saved as the best solution for that iteration; and 2) new individuals are created by genetic operators combining parts of selected individuals to create new ones. Finally, when all the iterations are completed, the algorithm chooses among all the individuals with the best *ttt* in each iteration, the one with the shortest *ttt*; that individual is the solution.

```
Start
    1. Read input parameters
    2. Create a pool of feasible bus lines
    3. Create an initial population using the pool of lines
    4. Repeat while the stop criterion is not reached:
        4.1. Calculate the ttt for each individual in the current population
        4.2. Sort the individuals according their fitness value
        4.3. Save the best individual
        4.4. Getting a new population applying the genetic operators

    5. Selecting the best solution achieved in the iterations

End
```

Algorithm 1. Steps of a genetic algorithm.

The following subsections provide further details on this algorithm.

2.1. Input parameters

A set of seven input parameters of the *lpp* was selected in this study to describe a situation. The input parameter *network* is a representation of an urban zone, it is represented by a graph where nodes are bus stops and links are the connections between nodes where buses can drive, in two directions. The *demand* is the total number of passengers. As is often the case, the *demand* in this study will be represented by an origin-destination (OD) matrix, defining the number of passengers going from node *i* to *j*. The OD matrix in this study is aggregated. The next input parameter is a vector with the *travel times*, that is, the time required to go from node *i* to *j* (the length of the links and the speed of the buses are implicitly defined). The *line length* represents the total time available for a bus line to go from its initial starting point to its ultimate destination. The *stop time* is the time required for a bus to stop at a bus stop. Although in many line planning studies this parameter is set to zero, in this work it will be considered explicitly to make the conditions of the situations more realistic. The *fleet size* is another input parameter that provides the maximum number of buses available to serve the passengers. In our study, it plays the role of a budget constraint for the operator. Since using more buses allows avoiding transfers, and thus reduces the *ttt*, the number of bus lines used in a high-quality solution will always be equal to this fleet size. The last input parameter is the *transfer time*, a penalty in the *ttt* when passengers need a transfer. In practice, the *transfer time* depends on the timetable, but since this work is focused on line planning and no timetable is designed yet, a fixed penalty will be used for all transfers (exact waiting and walking times are not calculated). In the literature, there are several studies using a fixed value as a replacement for the real transfer time (e.g., Mandl, 1979; Pattnaik *et al.*, 1998; Chakroborty & Wivedi, 2002; Nayeem *et al.*, 2014).

2.2. *Pool of lines*

The algorithm starts by creating a large set of feasible bus lines. A feasible line must satisfy the following constraints: (1) a node (bus stop) cannot be present twice in a line; (2) the length of the line cannot be greater than the parameter *line length*. It should be noted that we assume that buses stop at each node of their line, so a bus cannot drive by a bus stop without serving it. This corresponds to most research in literature on this *network*. To generate lines, for each pair with a non-zero value in the OD matrix, the algorithm first generates the fastest path (the shortest path in terms of time), and later *m* (6 in our algorithm) simple paths using a blind search method. A simple path, in graph theory, is defined as a path without repeating nodes. The fastest path and the simple paths are considered as bus lines in the pool.

2.3. *Initial population*

A population is illustrated in Figure 1. Each line plan in this figure (or solution or individual) is a set of bus lines and a bus line is a sequence of bus stops. There are only feasible individuals in a population. The individuals of the initial population are created by taking randomly lines from the pool of lines until a feasible line plan is achieved. The following criteria explain when they are feasible: (1) All the nodes of the *network* are present in the line plan and they are connected (the line plan is strongly connected). There is at least one path (with or without transfers) to connect any pair of nodes of the *network* (if there is demand on this pair). The algorithm defined by Tarjan (1972) is used to verify that a line planning is strongly connected. (2) The line planning has exactly *fleet size* lines. If the number of lines of an individual is *fleet size* but constraint (1) is not reached, then the line plan is rejected, and the procedure starts over again.

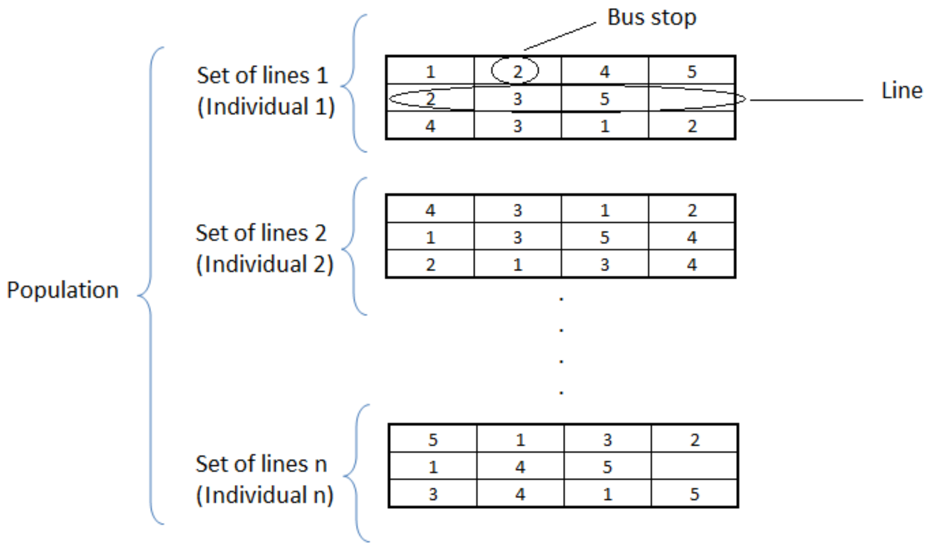

Figure 1. GA population.

The initial population must be sufficiently diverse to avoid premature convergence of the algorithm. An exploration spanning several areas of the search space, helps to achieve higher quality solutions. Mutation and a population with individuals different from each other, are methods that prevent convergence. That is why, when an individual is generated, an external composition control is executed before it is included in the initial population: the overlap between individuals cannot be higher than a certain percentage (30% in this study).

2.4. Individuals evaluation

For each individual in the population, the *ttt*, i.e. the quality of the solution or fitness value, is calculated. The line plans are represented by a graph where all nodes of the *network* are present, but the links are now defined by the routes of the bus lines. This initial graph is extended with additional nodes and links, in order to model the potential transfers and transfer times between the lines. On this artificial graph, the standard algorithm defined by Dijkstra (1959) is used to calculate the fastest path for each OD-pair. The summation of all fastest paths, weighted by the number of passengers on each OD-pair, results in the total travel time (*ttt*) for the line plan.

Next, the individuals in the population are sorted in ascending order based on their *ttt*. Then, an elitist selection process (Affenzeller, Wagner, Winkler, & Beham, 2009) selects the best *n* (10) individuals of the current population to populate the next population. Having *n* lower than the total number of individuals in the population, the new population is completed with the newly created individuals, resulting from applying genetic operators (crossover and mutation) on individuals of the current population. This iterative process is repeated until the stop criterion is fulfilled (in this work: a certain number of iterations: 100). The genetic operators are discussed next.

2.5. Genetic operators

In problems such as *lpp*, the classic genetic operators can produce infeasible individuals if the problem's inherent aspects are overlooked. To mention a case, when a genetic operator changes the sequence of stops along a bus line, it must ensure each stop in the new line can connect to the next directly. In a real network, the bus stops are not connected directly with all the others but only with a subset; these connections are defined by means of an adjacency matrix. Feasible individuals can be obtained if genetic operators use that matrix to create individuals whose stops are connected. Genetic operators considering these kind of problem-specific aspects, need to be developed.

Crossover is a genetic operator that combines individuals to create a new one. Two strategies to perform the crossover operator were designed for *lpp* in this study. In literature there are several methods to select the individuals to combine. In our approach, a rank selection is used. Every individual in the population is ranked based on its fitness value. Then, two individuals are selected randomly and the individual with the better rank becomes parent A. Then another two are selected and the individual with the best rank is selected as parent B. Once the parents are selected, the crossover is performed.

The first strategy tries to combine the parents copying the first *p* (*p* is a random number smaller than fleet size) lines of parent A (the lines are not sorted inside of the individuals) in the new individual and then it verifies if the new individual is feasible. If not, the crossover operator adds a line from parent B that has at least one node that is not yet present in the new individual. This process continues until the individual is strongly connected or until the possible combinations are exhausted, in which case the new individual is rejected, and the procedure starts over again.

The second strategy first selects a line and a stop of this line in parent A. Then it finds a line in parent B who also has the selected node and merges the lines using the node as a pivot. Thus, the new line is composed of the first nodes of the line in parent A up to the pivot, and the nodes from the pivot up to of the end of the line in parent B. The process is repeated until the new individual is strongly connected or until there are no more chances of combining elements.

In both strategies, if there is space for more lines in the new individual, the crossover operator will try to include extra lines that reduce the number of passengers needing a transfer. An OD pair with a transfer and a demand greater than the overall average demand is randomly selected. Crossover adds the fastest path between the OD pair as a line to the new individual until no more lines can be added.

Then the mutation operator is applied to all individuals of the new population. The mutation selects a line of an individual and tries to add a node to this line or delete a node from this line. To do this, the algorithm verifies if the changes in the line produce a feasible line and if the individual remains feasible, if not the mutation is rejected. Both operators, adding or removing, are applied with the same probability.

2.6. Solution

The best line plan for a given situation is the one with the lowest value of *ttt* in any population. The *GA* was coded and run in Matlab R2016b (9.1) under Windows 7 Professional on a computer with processor Intel i7 3.40 *GHz* and 16 *GB* of RAM.

3. CASE STUDY

The *network* used in this research is described by Mandl (1979) and used by many others (e.g., Baaj & Hani, 1991; Chakroborty & Wivedi, 2002; Zhao *et al.*, 2015). This network has 15 nodes and 21 direct links.

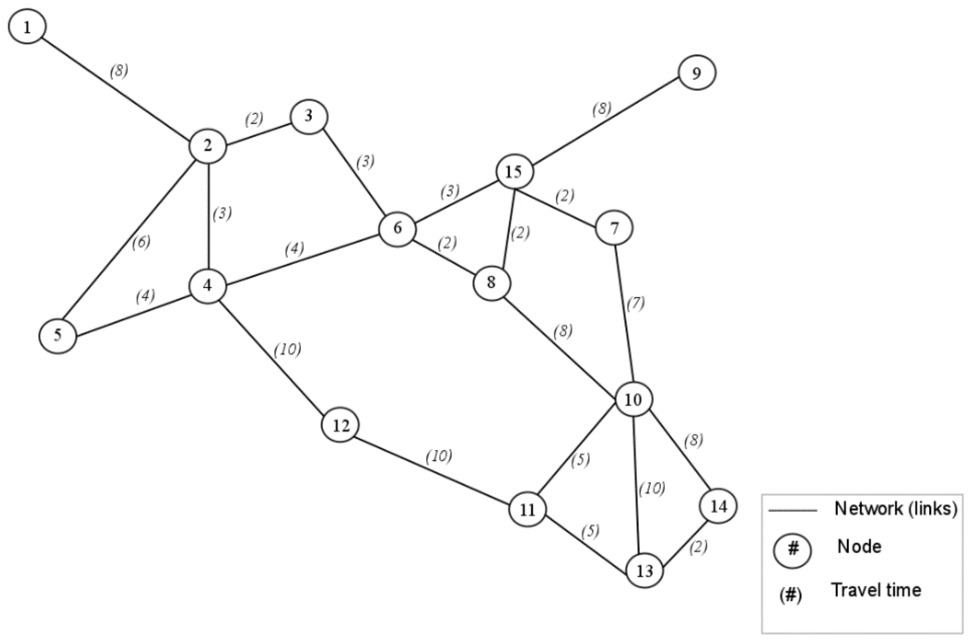

Figure 2. Mandl's network.

Figure 2 illustrates the location of the nodes and the *travel time* on each arc, also provided by Mandl. In this work, the speed of the buses is equal to 30km/h, which is a suitable speed for urban areas. The *demand* is distributed as in Table 1. All the nodes in the network have a demand greater than zero, except node 15, and the total demand is 15,570 travels. Nodes 10, 6 and 1 have the highest demand. The *demand* between nodes is symmetric. The *line length used here* is 50 minutes and the *fleet size* 4 buses. The value for *stop time* is 1 minute, and the *transfer time* is set to 10 minutes.

The *GA* runs 100 iterations for a population of 40 individuals where the 10 best individuals (*n*) are copied to the next generation. Since the *GA* uses random numbers during the calculations, the algorithm is executed 10 times and the best result out of these 10 runs, is considered as the final solution. These values were determined during preliminary tests. Table 2 presents the results of the different runs for Mandl’s network. Run 9 (in bold) has the lowest *ttt*. The last row indicates during which of these 100 iterations, the best *ttt* was obtained.

Figure 3 illustrates the resulting line plan with a *fleet size* of 4. This line plan copes with all *demand* in 185,930 minutes (*ttt*); the percentage of direct travels *(d0*), travels with 1 transfer (*d1*) and travels with more than 1 transfer (*dm*) are 94.24%, 5.38% and 0.38% respectively. The average travel time for passengers is 11.94 minutes.

Table 1. Origin-Destination (OD) matrix.

From/To	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	400	200	60	80	150	75	75	30	160	30	25	35	0	0
2	400	0	50	120	20	180	90	90	15	130	20	10	10	5	0
3	200	50	0	40	60	180	90	90	15	45	20	10	10	5	0
4	60	120	40	0	50	100	50	50	15	240	40	25	10	5	0
5	80	20	60	50	0	50	25	25	10	120	20	15	5	0	0
6	150	180	180	100	50	0	100	100	30	880	60	15	15	10	0
7	75	90	90	50	25	100	0	50	15	440	35	10	10	5	0
8	75	90	90	50	25	100	50	0	15	440	35	10	10	5	0
9	30	15	15	15	10	30	15	15	0	140	20	5	0	0	0
10	160	130	45	240	120	880	440	440	140	0	600	250	500	200	0
11	30	20	20	40	20	60	35	35	20	600	0	75	95	15	0
12	25	10	10	25	15	15	10	10	5	250	75	0	70	0	0
13	35	10	10	10	5	15	10	10	0	500	95	70	0	45	0
14	0	5	5	5	0	10	5	5	0	200	15	0	45	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 2. Runs of the normal situation.

# run	1	2	3	4	5	6	7	8	**9**	10
ttt (min)	188,120	188,870	188,160	189,670	188,130	189,290	188,040	188,700	**185,930**	190,460
# iteration	65	16	49	54	34	20	96	33	**73**	34

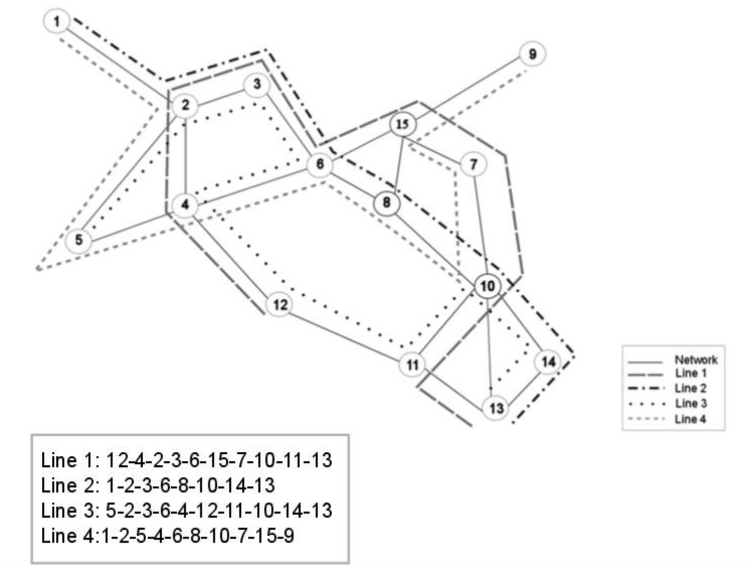
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Network
Line 1
Line 2
Line 3
Line 4
Line 1: 12-4-2-3-6-15-7-10-11-13
Line 2: 1-2-3-6-8-10-14-13
Line 3: 5-2-3-6-4-12-11-10-14-13
Line 4:1-2-5-4-6-8-10-7-15-9

Figure 3. Line plan for Mandl's network.

As the performance indicators showed, using this bus service, some travels can be done directly while others must make one or more transfers. For instance, a direct travel is possible for passengers moving from stop 4 to stop 10. Line 4 provides a direct connection starting at 4 passing by 6 and 8 before arriving at their final destination in 16 minutes. A transfer is required by passengers traveling from stop 3 to 9. In this case, the bus service does not provide a direct travel, so passengers take line 1 until they reach stop 6, then they must make a transfer to line 4 to reach their destination. This trip takes 25 minutes considering the penalty for the transfer. This solution will be referred as "*reference solution*" in Sections 4.2 and 4.3 of this paper.

4. EVALUATION

The three stages of the evaluation proposed in this study are described in the following subsections.

4.1. Benchmark comparisons

In this first stage of the evaluation the results are compared with those reported in the literature. Given the different input parameter settings and assumptions used in the literature, it seems impossible to fairly compare which algorithm is the best to solve the *lpp*. Therefore, the focus is on evaluating that the line plan obtained by our algorithm has at least a high quality. This means that the solution has similar travel times to those that have been reported so far. Furthermore, it should be noted that, until now, no optimal line plans are available for this (small) benchmark network. Studies as Mandl (1979), Baaj & Hani (1991), Chakroborty & Wivedi (2002), Zhao & Zeng (2006), Nikolić & Teodorović (2013), Nayeem *et al.* (2014), Zhao *et al.* (2015) uses Mandl's network as an instance to solve the *lpp* with a passenger´s perspective. Four of these studies report the passenger's average travel time for Mandl's network having 4 bus lines. The settings for our test were set as close as possible to the cases in the literature, but in some situations the values for input parameters are not reported or the assumptions are not the same as ours. The *stop time* was set to 0 min, *transfer time* to 5 min and *fleet size* to 4. Our *GA* then calculates an average travel time of 10.5 minutes for this new situation.

Figure 4 presents the benchmark and our results. In the case of the study of Nayeem et. al. (2014), the value corresponds to the one got by the algorithm called GAWIP.

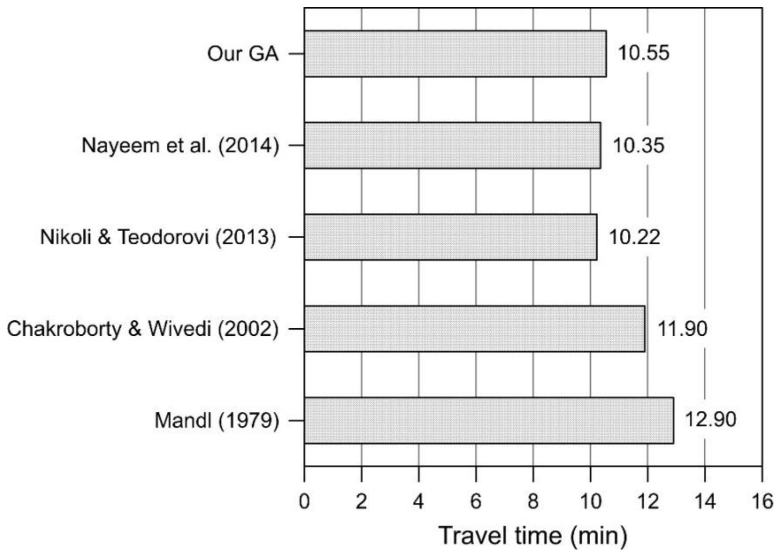

Figure 4. Average travel time reported in literature.

Our average travel time seems a rather good result compared to the ones presented in literature; only the studies of Nikoli & Teodorovi (2013) and Nayeem *et al.* (2014), outperform our results by a small difference. Nevertheless, it should be noted that although all the studies cited in Figure 4 aim to improve travel times for passengers, the objective function they use differs. For instance, Chakroborty & Wivedi (2002) proposed a method based on the sum of the product of some weighted (defined by the user) scores. The calculation of the scores is based on 5 criteria. Nikolić & Teodorović (2013) instead evaluated the solutions through a sum of the total travel time for all served passengers, the number of trips with a transfer and the number of unsatisfied passengers; the last two multiplied by a penalty. This variety of objective functions and assumptions about the problem creates the need to evaluate solutions also from other points of view. Stages 2 and 3 of the evaluation method proposed in this work attempted to make a deeper analysis of the solutions obtained with our algorithm.

4.2. Influence of random numbers

A set of random numbers are used in the algorithm. For instance, during initial population generation, a line is selected from the pool of lines by choosing a random index. This is done using a pseudo random number generator. This kind of tools return, after the application of mathematical and statistical methods, a number from a sequence, and the sequence of numbers is initialized through a number called *seed* and changes for different seeds. As in all cases where random numbers are used, our GA will get the same results if the *seed* remains.

This stage of the evaluation focuses on analyzing the influence of random numbers on the results of GA. Although by the application of different seeds, the results obtained differ and the differences between solutions should be smaller than those generated by changes in the conditions of the problem. That is, a change in the input parameters such as *demand* or *line length* should lead to larger changes in the solution than those produced by the random numbers.

The algorithm is executed 10 times for a given situation. The *fleet size* is the selected input parameter to vary during this test. This was selected since it is easy to see that more bus lines should generate a greater number of direct travels. If the algorithm is working properly, this change should result in a lower *ttt*.

Table 3. GA runs applying different random seeds.

Run	Reference	Situation 2 *fleet size 5*	Situation 3 *fleet size 6*
1	185,910	184,290	181,460
2	186,610	184,330	181,540
3	186,650	184,660	182,510
4	186,670	185,400	182,890
5	186,770	186,000	182,920
6	189,160	186,090	182,920
7	189,620	187,240	183,250
8	189,870	187,340	184,050
9	190,340	188,880	184,550
10	190,490	189,470	184,550
Average	188,209	186,370	183,064
SD	1,829	1,831	1,090
Difference		1,839	5,145

Table 3 presents ten *ttt* (sorted in ascending order) obtained by using different seeds. The *fleet size* changes from 4 in the reference solution to 5 and 6 in situations 2 and 3 respectively. In row number 11 the average of all *ttt* achieved in the runs for each situation is presented, later the standard deviation is calculated as a measure of the difference between runs in a same situation and presented in row 12. Finally, the absolute value of the difference between the average *ttt* of reference solution and situation 2 and 3 are calculated and presented in row 13. In both cases the difference due to changes in the input parameters is greater than the one due to random numbers.

4.3. Changes in the input parameters

An alternative to validate the performance of the algorithm, is based on changing several input parameter values. Then, it can be evaluated if the resulting changes to the line plans correspond to what could be expected. Changes in *fleet size* (4, 5 and 6 available buses), and *line length* (30, 50 and 70 minutes) are considered. The algorithm was used for 9 different situations and the results are presented in Table 4. Each row corresponds to a situation and columns 3 to 6 present the *ttt*, % of direct travels (*d0)*, % of 1 transfer travels (*d1*) and % of more than 1 transfer travels (*dm*) respectively. Column 7 depicts the required CPU-time in seconds to get the solution using the GA. Four types of changes in the line plan could be expected for these perturbations. In the following are these changes one by one discussed to verify if these indeed occur in the result of our algorithm.

1) *Better ttt when more lines are available:* This effect was described in the previous section, we now perform complementary tests were also the value of *line length* changes. The *ttt* in the new tests improves when the number of available lines increases as can be seen in Table 4.

Table 4. Line plan changes due to perturbations.

line length (min)	*fleet size*	*ttt (min)*	*d0%*	*d1%*	*dm%*	*CPU time (sec)*
30	4	196,250	86.64	12.97	0.39	91.06
	5	193,330	88.44	11.05	0.51	94.30
	6	192,030	89.79	10.21	0.00	123.23
50	4	185,930	94.22	5.39	0.39	54.64
	5	183,660	93.51	5.78	0.71	67.14
	6	181,340	96.34	3.66	0.00	80.01
70	4	186,050	94.28	5.59	0.13	59.48
	5	182,670	95.95	4.05	0.00	73.42
	6	179,860	98.84	1.16	0.00	91.36

2) *Better paths and avoiding transfers:* The results illustrate how the algorithm attempts to avoid transfers and to get better paths. Figure 5 illustrates an example when the *line length* is 30 min and the *fleet size* is changing from 4 to 5 and finally to 6. When considering the travels to stop 10 (the most demanding node) in the first line plan, passengers from stops 1 and 9 require a transfer to reach stop 10 and a direct travel is available from stop 12. This leads to 13,430 min to move all these passengers to stop 10. In the second line plan, the *fleet size* changes to 5, now passengers from stops 1 and 9 have a direct travel to 10, but passengers from stop 12 require a transfer. This leads to 13,230 min to move all these passengers to stop 10. Finally, the last line plan has *fleet size* 6, all stops, except 9, have a direct travel to 10. It should be noted that, for instance, more passengers are travelling to stop 10 from 12 than from 9. This leads to 12,470 min to move all these passengers to stop 10. The algorithm decides which transfer should be avoided for further improving the *ttt*. The resulting line plans thus correspond to what could be expected.

3) *Good solutions are recognized:* This expected outcome occurs when longer lines are allowed (*line length* increases). Under that condition, some fragments of the bus lines appear in every solution. For instance, when the *line length* is 30 min, the solution has the line (1-2-3-6-8-9), which needs 25 min to be traveled. If *line length* increases to 50 min the solution has a new largest (33 min) line (1-2-3-6-8-10-14-13) and finally for 70 min the solution has an even larger (52 min) line (1-2-3-6-8-10-11-12-4-5). So, the algorithm allows largest lines to keep the best parts (the fragment 1-2-3-6-8) of the previous smaller lines. These results illustrate that the algorithm improves solutions according to the values of the input parameters.

4) *Optimization focuses on ttt:* The last type of expected result is that higher values of *d0* might be expected as an indicator of improvement. However, since the only objective is to minimize *ttt*, this is not necessarily true. For instance, when the *line length* is 50 min and the *fleet size* changes from 4 to 5 (Table 4) the *ttt* improves, but *d0* gets worse. This can be explained by the fact that *line length* and *fleet size* play the role of constraints and not all travels can be served by a direct travel. Therefore, a good selection of direct and transfer travels will reduce the total travel time.

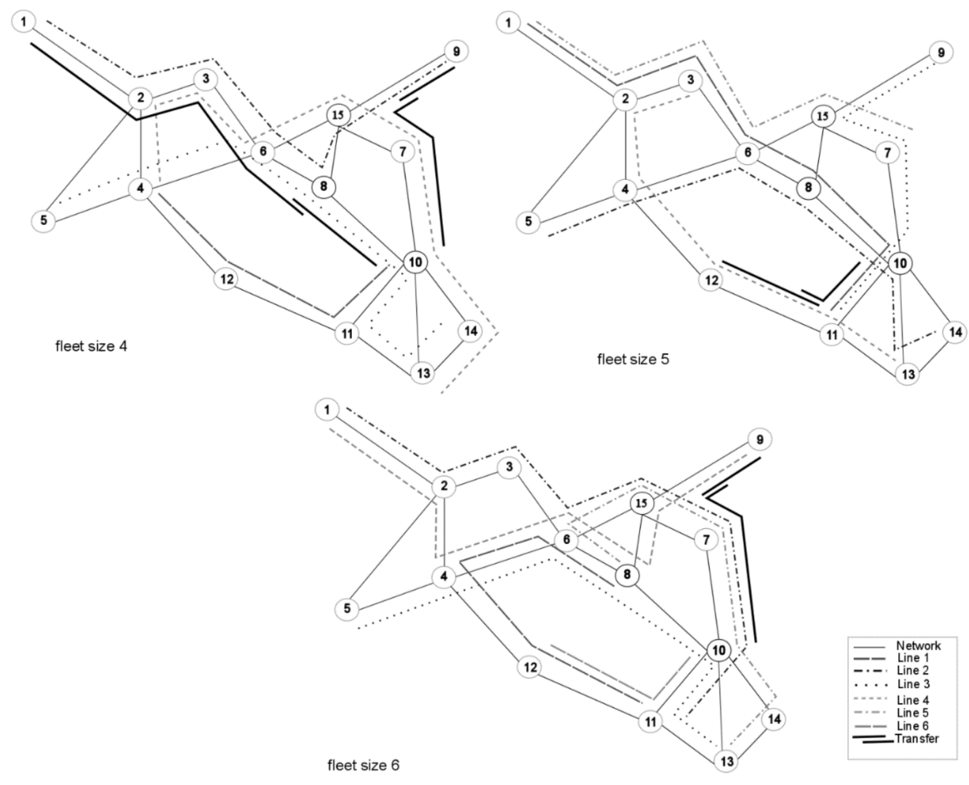

Figure 5. Line plans due to changes in fleet size.

5. CONCLUSIONS AND FUTURE WORK

In this paper, a well performing genetic algorithm aiming minimization of the total travel time (ttt) of passengers for the line planning problem (*lpp*) was developed and evaluated. The *fleet size* and *line length* of the bus lines are fixed and act as budget constraints from the operator side. The three-stage evaluation applied to the results revealed that: 1) the results have travel times similar to the best results reported in previous studies; 2) the bus lines found for a specific situation differ from those found for another situation. This difference is clearly larger than the difference between different runs using random numbers. In this way, it is possible to ensure that the bus lines are designed considering the conditions of their environment and are not a product of chance, and; 3) the solutions change according to changes in the input parameters. It illustrates that the design of the bus lines carried out by the algorithm, is strictly guided by the conditions defined by the input parameters and the objective function.

Future challenges are to evaluate, in the same fashion of this work, frequencies and timetables enabling maximization of the customers' satisfaction regarding the bus service. Analyzing, evaluating and selecting the input parameters and their effect on the solutions can help to achieve algorithms that design line plans for real instances of the problem. The results illustrate that the passenger-oriented GA is working as expected, and consequently it can be used as a tool to design real bus services and for further research on how to improve the performance of the service through a flexible line plan.

REFERENCES

Affenzeller, M., Wagner, S., Winkler, S., Beham, A. (2009). *Genetic algorithms and genetic programming: modern concepts and practical applications*. Boca Ratón, FL: CRC Press.

Baaj, M. H., Hani, S. M. (1991). An AI-based approach for transit route system planning and design. *Journal of Advanced Transportation*, *25*(2), 187-209.

Borndörfer, R., Grötschel, M., Pfetsch, M. E. (2008). *Models for line planning in public transport*. In: Computer-aided systems in public transport, pp. 363-378. Berlin, Germany: Springer.

Carrión, C., Levinson, D. (2012). Value of travel time reliability: A review of current evidence. *Transportation Research Part A*, *46*(4), 720-741.

Ceder, A., Wilson, N. (1986). Bus network design. *Transportation Research Part B*, *20(4)*, 331-344.

Chakroborty, P., Wivedi, T. (2002). Optimal route network design for transit systems using genetic algorithms. *Engineering Optimization*, *34*(1), 83-100.

Cipriani, E., Gori, S., Petrelli, M. (2012). Transit network design: A procedure and an application to a large urban area. *Transportation Research Part C*, *20*(1), 3-14.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, *1*(1), 269-271.

Farahani, R. Z., Miandoabchi, E., Szeto, W., Rashidi, H. (2013). A review of urban transportation network design problems. *European Journal of Operational Reserach*, *229*(2), 281-302.

Guihaire, V., Hao, J. (2008). Transit network design and scheduling: a global review. *Transportation Research part A*, *42*(10), 1251-1273.

Ibarra-Rojas, O., Delgado, F., Giesen, J., Muñoz, J. (2015). Planning, operation, and control of bus transport systems: A literature review. *Transportation Research Part B*, *77*, 38-75.

Kepaptsoglou, K., Karlaftis, M. (2009). Transit route network design problem: review. *Journal of Transportation Engineering*, *135*(8), 491-505.

Mandl, C. E. (1979). Evaluation and optimization of urban public transportation network. *European Journal of Operational Research*, *5*(6), 396-404.

Nayeem, M. A., Rahman, M. K., Rahman, M. S. (2014). Transit network design by genetic algorithm with elitism. *Transportation Research Part C*, *46*, 30-45.

Ngamchai, S., Lovell, D. (2003). Optimal time transfer in bus transit route network design using a genetic algorithm. *Journal of Transportation Engineering*, *129*(5), 510-521.

Nikolić, M., Teodorović, D. (2013). Transit network design by bee colony optimization. *Expert Systems with Applications*, *40*(15), 5945-5955.

Pattnaik, S., Mohan, S., Tom, V. (1998). Urban bus transit route network design using genetic algorithm. *Journal of Transportation Engineering*, *124*(4), 368-375.

Schöbel, A. (2012). Line planning in public transportation: models and methods. *OR Spectrum*, *34*(3), 491-510.

Tarjan, R. (1972). Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, *1*(2), 146-160.

Tom, V. M., Mohan, S. (2003). Transit route network design using frequency coded genetic algorithm. *Journal of Transportation Engineering*, *129*(2), 186-195.

Van Oort, N. (2011). *Service reliability and urban public transport design*. Doctoral thesis, Faculty of Civil Engineering and Geosciences, Delft University of Technology, Delft, The Netherlands.

Zhao, F., Zeng, X. (2006). Optimization of transit network layout and headway with a combined genetic algorithm and simulated annealing method. *Engineering Optimization*, *38*(6), 701-722.

Zhao, H., Xu, W., Jiang, R. (2015). The memetic algorithm for the optimization of urban transit network. *Expert Systems with Applications*, *42*(7), 3760-3773.