Evolutionary approach for bus synchronization

Sergio Nesmachnow[1], Jonathan Muraña[1], Gerardo Goñi[1],
Renzo Massobrio[1], and Andrei Tchernykh[2]

[1] Universidad de la República, Montevideo, Uruguay,
{sergion,jmurana,gerardo.goni,renzom}@fing.edu.uy
[2] Centro de Investigación Científica y Educacion Superior de Ensenada,
Baja California, México
chernykh@cicese.mx

Abstract. This article presents the application of evolutionary algorithms to solve the bus synchronization problem. The problem model includes extended synchronization points, accounting for every pair of bus stops in a city, and the transfer demands for each pair of lines on each pair of bus stops. A specific evolutionary algorithm is proposed to efficiently solve the problem and results are compared with intuitive algorithms and also with the current planning of the transportation system on real scenarios from the city of Montevideo, Uruguay. Experimental results indicate that the proposed evolutionary algorithm is able to improve in up to 13.33% the synchronizations with respect to the current planning and systematically outperforms other baseline methods.

1 Introduction

Transportation systems play a major role in nowadays society and are an important component of modern smart cities [7, 14]. Public transportation accounts for the most travels in large cities and provides the most efficient and environmental-friendly mean for citizens' mobility. However, the efficacy of public transportation systems requires a proper planning of routes, timetabling, buses, drivers, and other relevant subproblems, in order to provide good quality of service [4].

Synchronization of bus frequencies is an important goal from the point of view of users. Traditional approaches for public transportation network design and planning considered that having many different lines with different destinations and few synchronization (or transfer) points in the network allows a better transportation system, but in turn, that approach significantly increases the operation costs, because a larger number of lines are needed. Good quality of service can also be provided having a reduced number of lines and allowing transfers between them. In this scenario, the synchronization problem tries to define frequencies and headways of each line in order to maximize the transfer of passengers without significant waiting times. This way, the resulting public transportation system is more attractive to passengers and provides a better quality of service [4]. Synchronization is one of the most difficult tasks in public transportation planning. It has been often addressed intuitively, assuming that experienced operators are able to take proper decisions [3].

Nowadays, some public transportation systems have no limitations on the number of transfers that a passenger can perform. All bus stops are possible synchronization points, providing a more freely scenario for passengers to commute. In this scenario, the bus synchronization problem is more complex, as updated information must be considered to take into account the more frequently used connections. This is the situation of the public transportation system in Montevideo, which is the case study addressed in this article. Since the implementation of the Urban Mobility Plan [10], all pairs of bus stops are possible synchronization points for passengers to transfer between buses of different lines, using an intelligent card for ticket sales and travels. Thus, the formulation and scenarios of the bus timetabling synchronization problem are different of the ones previously proposed in literature. Furthermore, synchronization has become more important for citizens, as transfers allow improved mobility and more tickets are sold as the system provides a better service.

This article proposes a specific Evolutionary Algorithm (EA) [13] for efficiently solve the bus synchronization problem. The experimental evaluation is performed over realistic instances built considering real data from the Metropolitan Transportation System in Montevideo, Uruguay. Results obtained by the proposed EA are compared with intuitive algorithms to optimize synchronizations and also with the current planning of the transportation system in Montevideo. Plannings computed by the proposed EA improve up to 13.33% the number of synchronized trips, with respect to the current planning.

The research reported in this article was developed within the project 'Public transportation planning in smart cities' [15], funded by Fondo Conjunto de Cooperación Uruguay–México (2018–2019). The article is organized as follows. Section 2 introduces the bus synchronization problem and Section 3 reviews related works. The proposed EA for bus synchronization is described in Section 4. The experimental evaluation of the proposed method over realistic instances in Montevideo is reported in Section 5. Finally, the conclusions and the main lines for future work are formulated in Section 6.

2 The bus synchronization problem

This section presents an integer programming formulation for the bus synchronization problem, based on the previous model presented by Ibarra and Rios [9]. Specific features are included in order to model the reality and flexibility of nowadays Intelligent Transportation Systems.

2.1 Problem model

The problem accounts for the main goals of a modern transportation system: providing a fast and reliable way for the movement of citizens, while maintaining reasonable fares. The problem model mainly focuses on the quality of service provided to the users, i.e., a better traveling experience with reduced waiting times when using more than one bus for consecutive trips.

In the proposed model, the events of favoring passenger transfers with limited waiting times are called *synchronization* events. The study is aimed at solving real scenarios, based on real data from urban transit systems that accounts for the number of passengers that perform transfers between lines on each bus stop.

The main idea of the problem model is to divide any day into several planning periods on the basis of demand and travel time behavior of passengers. This way, the analysis of historical data allows obtaining similar accurate and almost deterministic information to build the problem scenarios. The mathematical formulation of the bus synchronization problem addressed in this article is presented in the next subsection.

2.2 Problem formulation

The mathematical formulation of the bus synchronization problem considers the following elements:

- A set of lines of the bus network $I = \{i_1, i_2, \ldots, i_n\}$. For each line $i \in I$, $J(i)$ is the set of lines that may synchronize with line i (in a synchronization node, see next item). Buses that operate each line have a maximum capacity C for passengers that board the bus in a second leg of a transfer trip.
- A set of synchronization nodes $B = \{b_1, b_2, \ldots, b_m\}$. Each node $b \in B$ is a triplet $<i, j, d_b^{ij}>$ indicating that lines i and j synchronize in b, and that the bus stops for lines i and j are separated by a distance d_b^{ij}.
- A planning period $[0, T]$, expressed in minutes, and the number of trips needed to fulfill the passengers' demand for each line, f_i.
- A *traveling time function* $TT : I \times B \to \mathbf{Z}$. $TT_b^i = TT(i, b)$ indicates the time to reach the synchronization node b for buses in line i (from the origin of the line). Generally, this value depends on several features, including the bus type, bus velocity, traffic in roads, passengers' demand, etc.
- A walking time function $WT : B \times I \times I \to \mathbf{N}$. $WT_b^{i,j} = WT(i, j, b)$ indicates the time needed for a pedestrian to walk the distance d_b^{ij}, according to a walking speed ws and specific features of synchronization node b (e.g, existence of pedestrian lines, crowding, traffic lights in intersections, etc.).
- A *demand function* $P : I \times I \times B \to \mathbf{Z}$. $P_b^{ij} = P(i, j, b)$ indicates the number of passengers that transfer from line i to line j in synchronization node b, in the planning period.
- A maximum waiting time W_b^{ij} for each synchronization node, indicating the maximum time that passengers are willing to wait for line j, after alighting from line i and walking to the stop of line j, in a synchronization node b.
- A headway time, defining the separation between consecutive trips of the same line i, defined in an interval $[h_i, H_i]$. Values of h_i and H_i are usually defined by the city administration.

The synchronization problem proposes finding appropriate values for the departure time for every trip of each line to guarantee the best synchronization for all lines with transfer demands in the planning period T.

The mathematical model is formulated in Eq. 1. Departure times of each trip are represented by integer variables X_r^i. Synchronizations are represented by binary variables Z_{rsb}^{ij} that define if trip r of line i and trip s of line j are synchronized in node b. The proposed objective function weights synchronizations according to the number of passengers that transfer in the planning period, thus giving priority to synchronization nodes with larger transfer demands.

$$\begin{aligned}
\text{maximize} \quad & \sum_{b\in B}\sum_{i\in I}\sum_{j\in J(i)}\sum_{r=1}^{f_i}\sum_{s=1}^{f_j} Z_{rsb}^{ij} \times \min(\frac{P_b^{ij}}{f_i}, C) && \text{(1a)}\\
\text{subject to} \quad & X_1^i \leq H^i && \text{(1b)}\\
& T - H^i \leq X_{f_i}^i \leq T && \text{(1c)}\\
& h^i \leq X_{r+1}^i - X_r^i \leq H^i && \text{(1d)}\\
& (X_s^j + TT_b^j) - (X_r^i + TT_b^i) > WT_b^{i,j} \text{ if } Z_{rsb}^{ij} = 1 && \text{(1e)}\\
& (X_s^j + TT_b^j) - (X_r^i + TT_b^i) \leq W_b + WT_b^{i,j} \text{ if } Z_{rsb}^{ij} = 1 && \text{(1f)}\\
& X_r^i \in \{1, \ldots, T\}, Z_{rsb}^{ij} \in \{0, 1\} && \text{(1g)}
\end{aligned}$$

The objective function of the problem (Eq. 1a) proposes maximizing the number of synchronized transfers, weighted by the corresponding transfer demand for each trip in each synchronization node. When computing the objective function, the demand is split uniformly among the f_j trips of line j. This is a realistic assumption for planning periods where demand does not vary significantly, such as in the case study presented in this article. The number of synchronized passengers on each synchronization node is bounded for the capacity for transfer passengers C. Equations 1a–1g specify the constraints of the problem.

3 Related works

Daduna and Voß [5] studied the schedule synchronization problem on bus networks, to minimize the waiting time of passengers. Different objectives were studied, including a weighted sum considering transfers and the maximum waiting time at a transfer zone. Simulated Annealing and Tabu Search were analyzed for simple versions of the problem. Tabu Search computed better solutions than Simulated Annealing over randomly generated examples based on the Berlin Underground network. In addition, three real-world cases from different German cities were studied. The trade-off between operational costs and user efficiency suggested that multiobjective approaches should be considered.

Ceder et al. [2] studied the problem of maximizing the number of synchronization events between bus lines at shared stops, i.e., maximizing the number of simultaneous arrivals. An heuristic approach based on a greedy procedure to select nodes from the bus network was proposed, to efficiently solve the problem by defining custom timetables. Both articles were focused on simultaneous bus arrivals, and results reported consisted of examples is presented that illustrate synchronizations on small instances with few nodes and few lines.

Fleurent et al. [6] considered a synchronization metric including weights defined by experts and public transport authorities to minimize vehicle scheduling costs. An heuristic algorithm was proposed to solve network flow problems that accounts for the synchronization metric and other operation costs. Experiments performed on just two small scenarios from Montréal, Canada, computed different timetables when varying the weights used in the proposed metric.

Ibarra and Ríos [9] studied the bus synchronization problem in the bus network of Monterrey, Mexico. A flexible formulation of the problem was proposed, considering a time window between travel times to account for variations. A Multi-start Iterated Local Search (MILS) was evaluated over 8 instances modeling the bus network in Monterrey (15 to 200 lines, and 3 to 40 synchronization points). MILS was compared against a Branch & Bound method (which failed to compute optimal solutions in two hours) and a simple upper bound computed by adding the possible trips to synchronize. The method was able to compute efficient solutions for medium-size instances in less than one minute, but the gaps of MILS did not scale, as they were small for large instances.

Later, Ibarra et al. [8] solved the multiperiod bus synchronization problem, optimizing multiple trips of a given set of lines. MILS, Variable Neighborhood Search and a simple population-based approach were proposed to solve the problem. All methods computed solutions with similar quality to an exact approach over academic instances with few synchronization points. Multiperiod timetables were up to 20% better than merging single period timetables. Results for a sample case study using data for a single line showed that maximizing synchronizations for a specific node usually reduces synchronizations for other nodes.

The model considered in our article includes additional features to the one proposed by Ibarra and Ríos [9]: scenarios where every pair of bus stops are possible transfer zones to synchronize, and real transfer demand in each possible transfer zone. The proposed EA also captures the features of existing solutions and accounts for real operation constraints for the case study in Montevideo.

4 The proposed EA for bus synchronization

This section describes the main features of the proposed EA for solving the bus synchronization problem.

4.1 Solution encoding

Candidate solutions to the problem are represented using integer vectors, where each integer value represents the headway (in minutes) of a bus line, i.e., the time between consecutive trips of the same line. More formally, a candidate solution to the problem is represented by $X = x_1, x_2, \ldots x_n$, where n is the number of bus lines in the problem instance, $x_i \in \mathbf{Z}^+$, and $h^i \leq x_i \leq H^i$. Fig. 1 describes a solution representation for a problem instance with N bus lines. In the example shown, buses from line 1 are scheduled to depart every 12 minutes, buses from line 2 every 8 minutes, etc.

12	**8**	**9**	**10**	**...**	**...**	**...**	**...**	**...**	**...**	**7**	**15**	**8**	**12**	**10**
line 1	line 2	line 3	line 4	...	...	...	...	...	...	...	...	...	line N-1	line N

Fig. 1. Example of a solution representation

4.2 Evolution model and evolutionary operators

Evolution model. The $(\mu + \lambda)$ evolution model [1] is applied in the proposed EA: μ parents generate λ individuals, which compete between them and with their parents, to determine the individuals that form the new population on the next generation. Preliminary experiments demonstrated that $(\mu + \lambda)$ evolution was able to provide better solutions and more diversity than a traditional generational model.

Initialization operator. A seeded initialization is applied in the proposed EA. Randomly generated solutions are included in the initial population, accounting for the constraints defined for the headways of each line. In addition, values for the headways from the currently real solution applied by the transportation administration in Montevideo are included in the initial population. Solutions generated by greedy approaches to maximize the number of synchronizations are also included. Some of the initial solutions are modified by applying a *shaking* procedure that randomly modifies some of the information for specific lines. This initialization procedure intends to capture the main features of existing solutions and accounts for real operation constraints for the case study in Montevideo, and also provide diversity to the evolutionary search.

Selection operator. The traditional tournament selection is applied, with tournament size two individuals, and one individual survives. Tournament selection allowed to compute better results than proportional selection in preliminary calibration experiments, mainly due to the appropriate level of selection pressure for the evolution of solutions.

Recombination operator. The recombination operator is a specific variant of two-point crossover. It defines two crossover points randomly and exchanges the information encoded in both parents between the crossover points. This operator was conceived to preserve specific features of lines already synchronized in parent solutions, trying to keep useful information in the offspring generation process. The recombination operator is applied to individuals returned by the selection operator, with a probability p_R.

Mutation operator. The mutation operator applied is a specific variant of Gaussian mutation. Specific position(s) in a solution are modified according to a Gaussian distribution, and taking into account the thresholds defined by the minimum and maximum frequencies for each line. The mutation operator is applied to every gene with a probability p_M.

4.3 Fitness function description

The fitness function accounts for the number of synchronized trips and their corresponding demands, according to the formulation in Eq. 1. The fitness is computed by the procedure described in Algorithm 1. For each synchronization point (*sp*) of the scenario, the demand is accumulated for each pair of synchronized trips. Variables *x*,*y* are the frequencies assigned by the solution (*sol*) to the lines involved in the synchronization point *sp*, and the function *get_trips* generates two vectors with the departing time of each trip of these lines.

Algorithm 1 Fitness evaluation for solutions

INPUT: *sol*, *scenario* **OUTPUT:** *fitness*

```
fitness ← 0
for sp in get_sync_points(scenario) do
    line_i,line_j,TT_i,TT_j,dist,demand,W_b ← get_elements(sp)
    t_dist ← ((dist/1000) /WALK_SPEED) × 60
    x,y ← get_sol_freqs(sol,line_i,line_j)
    trips_i,trips_j ← get_trips(x,y,T)
    for i = 1 to len(trips_i) do
        for j = 1 to len(trips_j) do
            wait_time = (trips_j[j] + TT_j) − (trips_i[i] + TT_i) − t_dist
            if wait_time > y then
                wait_time ← y
            end if
            if wait_time > 0 & wait_time ≤ W_b then
                fitness ← fitness + min(demand × x, C × T)
                break
            end if
        end for
    end for
end for
```

5 Experimental evaluation

This section reports the experimental evaluation of the proposed EA for the bus synchronization problem.

5.1 Methodology for generating problem instances: general considerations

Real problem instances were built using real data from the Metropolitan Transportation System in Montevideo, Uruguay: bus lines description, routes, schedules, and bus stops location in the city. Transfers information corresponds to real data from 2015 [11].

The key elements of the scenario and problem instances and how they were built are described next:

- The *type of day* determines if the considered problem instance corresponds to a working day or a weekend.
- The *period* is the interval of hours considered for the schedule. A period is characterized by its duration, traffic level (rush hour, normal demand, or low demand) and overall bus demand.
- The *demand function* is computed from transfers information, registered by smart cards used to sell tickets.
- *Synchronization points* are chosen according to their demand. The pairs of bus stops with the largest number of registered transfers for the period in the corresponding type of day are selected.
- The *bus lines* are the ones passing by the synchronization points.
- The *time traveling function* $t(i,b)$ for line i in synchronization point b are computed by Eq. 2, where r are trips of line i in the period, at_{ro} is the arrival time of trip r to the first stop in the route, at_{rb} is the arrival time to the stop of the synchronization point b and f_i is the number of trips in the period.

$$t(i,b) = \frac{\sum_{r=1}^{f_i} at_{rb} - at_{ro}}{f_i} \tag{2}$$

- The *walking time function* is the estimated walking speed of a person multiplied by the distance between bus stops in each synchronization point, computed using geospatial information about stops.
- The *headway limits* for each line (h_i and H_i) are computed considering the real bus schedule in period.
- The *maximum waiting time* is equal to λH, with $\lambda \in [0.75,\ 0.9,\ 1.0]$. This formulation allows configuring instances with different levels of quality of service.

5.2 Problem instances using data from Montevideo

Thirty problem instances were defined, accounting for three different dimensions (30, 70, and 110 synchronization points), using real information about bus operating in Montevideo, Uruguay. Synchronization points were chosen randomly from an universe of 170 points (the most demanded transfer zones for the considered period).

Each problem instance is named as BS.[hh].[HH].[NP].[NL].[T].[λ].[id]. hh is the start hour of the period, HH is the end hour of the period, NP is the number of synchronization points in the instance, NL is the number of bus lines, T is the duration of the planning period, λ is the coefficient applied to W_b (percentage) and id is a relative identifier for instances with the same NL and λ.

In the instances solved in the experiments, hh is 12 (12:00 hs), HH is 14 (14:00 hs) NP $in\,[30, 70, 110]$, NL is determined by the selected synchronization points, the period T is 120 minutes, in line with related works, and $\lambda\ in\,[75, 90, 100]$. Figure 5.2 shows sample synchronization points chosen for building instances, distributed in the map of Montevideo.

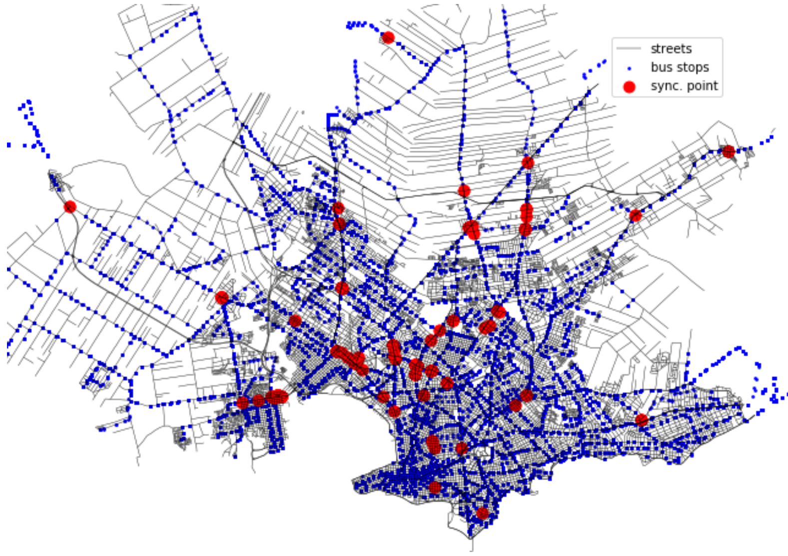

Fig. 2. Geographic distribution of synchronization points in Montevideo

5.3 Baseline solutions and metrics description

A set of baseline solutions were considered for the comparison of the EA results. The main point for comparison is the current schedule in the transportation system of Montevideo (the *real timetable*), which provides the actual level of service regarding direct travels and transfers. In addition, solutions using the minimum headway (h_i) and the maximum headway for each line for each line (H_i) are used. These two solutions are included in the comparison, even though they are not useful to be implemented in practice, because configuring all lines to operate at minimum headway accounts for a very large (and expensive) number of trips, and configuring all lines to operate at maximum headway provides a very limited quality of service, regarding travel time.

The main metrics applied for the evaluation are the number of synchronized trips for passengers and the waiting time, computed as the average of the time difference between the arrivals of a trip of line i and the next trip of line j to the synchronization point, considering the walking time.

5.4 Parameter setting

EAs are stochastic methods, thus parameter setting analysis are required to determine the configuration that allows computing the best results. The values of stopping criterion ($\#gen$), population size (ps), recombination probability (p_R), and mutation probability (p_M) were studied on three instances, different from the ones used in validation experiments, in order to avoid bias. Fifty independent executions of the proposed EA were performed for each problem instance. Candidate values for the studied parameters were $\#gen \in \{500, 1000, 2000\}$, $ps \in \{50, 100, 200\}$, $p_R \in \{0.6, 0.75, 0.9\}$, and $p_M \in \{0.01, 0.05, 0.1\}$.

Results allowed concluding that population size and stopping criterion did not affect solution quality, as the EA rapidly converges to high quality solutions in a few (i.e., hundred) generations, despite the population size, so $\#gen$=1000 and ps=100 were selected for validation experiments. Regarding operator probabilities, the best results according to a Student's t-test applied to analyze the results distributions were obtained with the configuration $p_R = 0.9$ and $p_M = 0.01$.

5.5 Development and execution platform

The proposed EA was implemented in Malva (`github.com/themalvaproject`). The experimental evaluation was performed on a Quad-core Xeon E5430 at 2.66GHz, 8 GB RAM, from Cluster FING, Universidad de la República [12].

5.6 Numerical results

Table 1 reports the numerical results of the proposed EA and the reference results for the baseline solutions. The number of synchronized trips (st) are also reported for both real timetable and EA solutions.

scenario	real timetable		h_i	H_i	EA	
	fitness	*st*			*fitness*	*st*
BS.12-14.30.36.120.90.2	32194	244	33310	22990	34142	345
BS.12-14.30.37.120.100.2	27913	217	30148	22938	30868	301
BS.12-14.30.37.120.90.0	30005	235	32745	22644	33415	338
BS.12-14.30.38.120.100.3	30407	292	31082	27327	31843	346
BS.12-14.30.38.120.100.4	29222	288	30544	25409	31449	355
BS.12-14.30.38.120.90.3	30827	274	32181	23252	32725	305
BS.12-14.30.39.120.100.0	23380	210	24393	18707	25536	295
BS.12-14.30.39.120.75.3	31508	217	33147	22621	33710	386
BS.12-14.30.40.120.90.4	23199	235	24633	18006	25501	278
BS.12-14.30.41.120.100.1	33965	292	34896	29907	**35672**	**359**
BS.12-14.70.59.120.100.3	67065	607	69430	59005	71587	834
BS.12-14.70.62.120.100.0	57979	528	61179	47123	62988	786
BS.12-14.70.63.120.90.4	57665	522	62145	47087	63361	661
BS.12-14.70.64.120.90.2	64486	561	67972	49122	69912	776
BS.12-14.70.65.120.100.2	62293	510	66559	52874	68269	668
BS.12-14.70.65.120.90.0	65681	569	70446	48348	**71938**	808
BS.12-14.70.66.120.90.1	57007	551	60906	43406	62447	740
BS.12-14.70.66.120.90.3	66642	597	70380	50242	71696	**845**
BS.12-14.70.67.120.75.1	63145	525	67926	43789	70183	721
BS.12-14.70.68.120.75.0	61457	543	67678	40321	69648	783
BS.12-14.110.77.120.100.3	103929	935	108875	90401	111314	1335
BS.12-14.110.78.120.90.4	95902	845	101384	74668	103662	1262
BS.12-14.110.79.120.75.3	93480	855	101364	65490	103434	1291
BS.12-14.110.79.120.90.1	96438	838	101931	73964	104232	1111
BS.12-14.110.79.120.90.3	96064	912	101723	74398	103930	1325
BS.12-14.110.81.120.100.1	97745	908	102858	85600	105444	1273
BS.12-14.110.82.120.75.0	96298	886	105089	64377	**107311**	**1346**
BS.12-14.110.83.120.100.2	94269	834	100642	81916	102754	1191
BS.12-14.110.83.120.75.2	89073	799	96965	58840	99235	1113
BS.12-14.110.83.120.90.0	94522	829	101517	71081	103932	1163

Table 1. Numerical results for the bus synchronization problem.

Results in Table 1 indicate that the proposed EA computed accurate solutions, systematically improving the number of synchronized trips over the current real solution according to the actual timetable defined for the transportation system in Montevideo. Best fitness and synchronized trips of the EA in each problem dimension, are marked in bold. Overall, the average number of synchronized trips for EA solutions was 37.62% greater than the one in the current timetable and the most notable difference was 51.92%. The percent improvements (GAP) of the solutions computed by the proposed EA over the considered baseline algorithms are reported in Table 2. The best GAPs of the EA regarding h_i, H_i, and current planning in each problem dimension are marked in bold.

scenario	GAP real	GAP h_i	GAP H_i
BS.12-14.30.36.120.90.2	6.05%	2.50%	48.51%
BS.12-14.30.37.120.100.2	10.59%	2.39%	34.57%
BS.12-14.30.37.120.90.0	**11.36**%	2.05%	47.57%
BS.12-14.30.38.120.100.3	4.72%	2.45%	16.53%
BS.12-14.30.38.120.100.4	7.62%	2.96%	23.77%
BS.12-14.30.38.120.90.3	6.16%	1.69%	40.74%
BS.12-14.30.39.120.100.0	9.22%	**4.69**%	36.51%
BS.12-14.30.39.120.75.3	6.99%	1.70%	**49.02**%
BS.12-14.30.40.120.90.4	9.92%	3.52%	41.63%
BS.12-14.30.41.120.100.1	5.03%	2.22%	19.28%
average $n = 30$	7.77%	2.62%	35.81%
BS.12-14.70.59.120.100.3	6.74%	3.11%	21.32%
BS.12-14.70.62.120.100.0	8.64%	2.96%	33.67%
BS.12-14.70.63.120.90.4	9.88%	1.96%	34.56%
BS.12-14.70.64.120.90.2	8.41%	2.85%	42.32%
BS.12-14.70.65.120.100.2	9.59%	2.57%	29.12%
BS.12-14.70.65.120.90.0	9.53%	2.12%	48.79%
BS.12-14.70.66.120.90.1	9.54%	2.53%	43.87%
BS.12-14.70.66.120.90.3	7.58%	1.87%	42.70%
BS.12-14.70.67.120.75.1	11.15%	**3.32**%	60.28%
BS.12-14.70.68.120.75.0	**13.33**%	2.91%	**72.73**%
average $n = 70$	9.44%	2.62%	42.94%
BS.12-14.110.77.120.100.3	7.11%	2.24%	23.13%
BS.12-14.110.78.120.90.4	8.09%	2.25%	38.83%
BS.12-14.110.79.120.75.3	10.65%	2.04%	57.94%
BS.12-14.110.79.120.90.1	8.08%	2.26%	40.92%
BS.12-14.110.79.120.90.3	8.19%	2.17%	39.69%
BS.12-14.110.81.120.100.1	7.88%	**2.51**%	23.18%
BS.12-14.110.82.120.75.0	**11.44**%	2.11%	66.69%
BS.12-14.110.83.120.100.2	9.00%	2.10%	25.44%
BS.12-14.110.83.120.75.2	11.41%	2.34%	**68.65**%
BS.12-14.110.83.120.90.0	9.96%	2.38%	46.22%
average $n = 110$	9.18%	2.24%	43.07%
overall average	8.79%	2.49%	40.61%

Table 2. GAPs of the proposed EA over baseline algorithms.

Results confirm that the proposed EA is able to improve over both current planning and also over naive solutions that account for the minimum and maximum headway. The best improvement was 13.33% in (weighted) synchronizations with respect to the current planning in Montevideo. In average, the proposed EA improved 8.79% over the current planning.

Regarding robustness and scalability, results indicate that the EA scaled properly when solving problem instances with a larger number of synchronization points. GAPs improved from 7.77% (average) and 11.36% (best) in small problem instances with 30 synchronization points to 9.44% (average) and 13.33% (best) in medium problem instances with 70 synchronization points. The GAPs of the proposed EA over baseline solutions are graphically presented in Fig. 3.

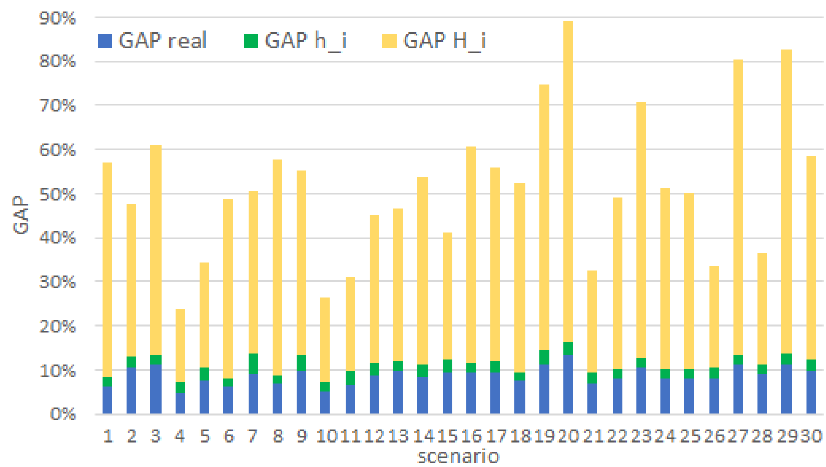

Fig. 3. Improvements (GAP) over baseline solutions

Regarding quality of service metrics, results of average waiting time per trip, reported in Tables 3 (values) and 4 (GAPs) indicates that the EA improves over the real timetable solutions for all scenarios. Furthermore, the EA also improves or equals h_i solutions in six scenarios (marked in bold). The best GAPs over h_i, H_i, and current planning in each problem dimension, are marked in bold.

Finally, regarding performance, the proposed EA had a remarkable computational efficiency. The average execution time required to perform 1000 generations was 78.2 seconds. The execution time did not increase significantly when solving the largest problem instances (in those instances, the average execution time was 122.7 seconds). These results confirm that the proposed EA is a useful tool for performing fast planning in Intelligent Transportation Systems, able to account for dynamic situations required in nowadays smart cities. Results demonstrated that the proposed EA can compute updated frequency plans (e.g., for the next hour) or even analyze different schedules accounting for different quality of service, a scenario that cannot be addressed with traditional exact methods (e.g. Branch and Bound), which fail to compute solutions in one hour for even small problem instances, as reported in related works [9].

scenario	real timetable	h_i	H_i	EA
BS.12-14.30.36.120.90.2	6.54	4.39	11.56	5.28
BS.12-14.30.37.120.100.2	8.14	5.34	14.11	6.08
BS.12-14.30.37.120.90.0	7.16	5.27	12.20	**5.14**
BS.12-14.30.38.120.100.3	7.53	5.25	12.65	6.36
BS.12-14.30.38.120.100.4	7.50	5.24	13.45	5.71
BS.12-14.30.38.120.90.3	6.78	4.59	12.60	6.19
BS.12-14.30.39.120.100.0	7.98	5.96	14.86	**5.86**
BS.12-14.30.39.120.75.3	8.31	5.93	15.17	**5.65**
BS.12-14.30.40.120.90.4	8.00	5.66	12.76	6.22
BS.12-14.30.41.120.100.1	7.50	5.67	14.62	6.42
BS.12-14.70.59.120.100.3	7.85	5.85	13.19	**5.85**
BS.12-14.70.62.120.100.0	7.68	5.46	13.50	5.60
BS.12-14.70.63.120.90.4	8.52	5.37	16.54	6.43
BS.12-14.70.64.120.90.2	7.88	5.40	13.84	5.59
BS.12-14.70.65.120.100.2	8.40	5.66	14.49	6.95
BS.12-14.70.65.120.90.0	7.34	5.05	12.75	**5.03**
BS.12-14.70.66.120.90.1	8.40	5.40	17.07	7.05
BS.12-14.70.66.120.90.3	8.08	5.61	14.76	5.82
BS.12-14.70.67.120.75.1	7.65	5.41	13.62	5.59
BS.12-14.70.68.120.75.0	7.81	5.75	13.48	6.55
BS.12-14.110.77.120.100.3	7.84	5.64	13.75	5.71
BS.12-14.110.78.120.90.4	8.02	5.39	15.42	5.73
BS.12-14.110.79.120.75.3	8.26	5.59	15.90	5.61
BS.12-14.110.79.120.90.1	7.83	5.28	15.06	6.62
BS.12-14.110.79.120.90.3	8.35	5.79	14.90	6.36
BS.12-14.110.81.120.100.1	8.20	5.50	14.47	6.08
BS.12-14.110.82.120.75.0	7.59	5.48	13.08	**5.42**
BS.12-14.110.83.120.100.2	8.27	5.57	15.86	5.81
BS.12-14.110.83.120.75.2	8.47	5.77	16.64	8.05
BS.12-14.110.83.120.90.0	7.88	5.36	13.88	6.27

Table 3. Average waiting time results.

6 Conclusions and future work

This article presented a specific EA designed to efficiently solve a variant of the bus timetable synchronization problem.

A new problem formulation is presented, accounting for features of real scenarios modeled from data collected by nowadays Intelligent Transportation Systems. A specific EA was proposed to solve the problem, including simple and intuitive variation operators to provide both accuracy and diversity on solutions. The proposed fitness values takes into account the number of synchronized trips and the real demands of transfers on each bus stop. The proposed approach is generic and can be easily adapted to be applied and scale up to different scenarios.

scenario	GAP real	GAP h_i	GAP H_i
BS.12-14.110.77.120.100.3	27.17%	-1.24%	58.47%
BS.12-14.110.78.120.90.4	28.55%	-6.31%	62.84%
BS.12-14.110.79.120.75.3	**32.08**%	-0.36%	**64.72**%
BS.12-14.110.79.120.90.1	15.45%	-25.38%	56.04%
BS.12-14.110.79.120.90.3	23.83%	-9.84%	57.32%
BS.12-14.110.81.120.100.1	25.85%	-10.55%	57.98%
BS.12-14.110.82.120.75.0	28.59%	**1.09**%	58.56%
BS.12-14.110.83.120.100.2	29.75%	-4.31%	63.37%
BS.12-14.110.83.120.75.2	4.96%	-39.51%	51.62%
BS.12-14.110.83.120.90.0	20.43%	-16.98%	54.83%
average $n = 30$	21.61%	-11.34%	55.79%
BS.12-14.70.59.120.100.3	25.48%	0.00%	55.65%
BS.12-14.70.62.120.100.0	27.08%	-2.56%	58.52%
BS.12-14.70.63.120.90.4	24.53%	-19.74%	**61.1**2%
BS.12-14.70.64.120.90.2	29.06%	-3.52%	59.61%
BS.12-14.70.65.120.100.2	17.26%	-22.79%	52.04%
BS.12-14.70.65.120.90.0	**31.47**%	**0.40**%	60.55%
BS.12-14.70.66.120.90.1	16.07%	-30.56%	58.70%
BS.12-14.70.66.120.90.3	27.97%	-3.74%	60.57%
BS.12-14.70.67.120.75.1	26.93%	-3.33%	58.96%
BS.12-14.70.68.120.75.0	16.13%	-13.91%	51.41%
average $n = 30$	24.20%	-9.98%	57.71%
BS.12-14.30.36.120.90.2	19.27%	-20.27%	54.33%
BS.12-14.30.37.120.100.2	25.31%	-13.86%	56.91%
BS.12-14.30.37.120.90.0	28.21%	2.47%	57.87%
BS.12-14.30.38.120.100.3	15.54%	-21.14%	49.72%
BS.12-14.30.38.120.100.4	23.87%	-8.97%	57.55%
BS.12-14.30.38.120.90.3	8.70%	-34.86%	50.87%
BS.12-14.30.39.120.100.0	26.57%	1.68%	60.57%
BS.12-14.30.39.120.75.3	**32.01**%	**4.72**%	**62.76**%
BS.12-14.30.40.120.90.4	22.25%	-9.89%	51.25%
BS.12-14.30.41.120.100.1	14.40%	-13.23%	56.09%
average $n = 110$	23.67%	-11.34%	58.57%
overall average	23.16%	-10.88%	57.36%

Table 4. GAPs on waiting time of the proposed EA over baseline algorithms.

The experimental evaluation of the proposed algorithm was performed over instances of significantly larger dimension than those previously addressed in the related literature. Problem instances based on real-data from the ITS in Montevideo, Uruguay were built, consisting of up to 83 lines and 110 synchronization points. Results show that the proposed evolutionary approach is able to compute accurate solutions, improving up to 13.33% in the fitness values and up to 24.20% in the waiting times, when compared to the current real timetable in Montevideo.

The main lines for future work are related to develop explicit multiobjective methods to solve the problem and improve the accuracy of the computed results. In this regard, historical GPS location data of buses can be used to obtain more accurate approximations of headways and travel times in the public transportation system. Furthermore, dynamic models should be explored to account for real-time location information to react to traffic congestion and demand spikes and deal with transfer synchronization at the operational level.

References

1. T. Bäck, D. Fogel, and Z. Michalewicz, editors. *Handbook of evolutionary computation*. Oxford University Press, 1997.
2. A. Ceder, B. Golany, and O. Tal. Creating bus timetables with maximal synchronization. *Transportation Research Part A: Policy and Practice*, 35(10):913–928, 2001.
3. A. Ceder and O. Tal. Timetable synchronization for buses. In *Lecture Notes in Economics and Mathematical Systems*, pages 245–258. Springer Berlin Heidelberg, 1999.
4. A. Ceder and N. Wilson. Bus network design. *Transportation Research Part B: Methodological*, 20(4):331–344, 1986.
5. J. Daduna and S. Voß. Practical experiences in schedule synchronization. In *Lecture Notes in Economics and Mathematical Systems*, volume 430, pages 39–55. Springer Berlin Heidelberg, 1995.
6. C. Fleurent, R. Lessard, and L. Séguin. Transit timetable synchronization: Evaluation and optimization. In *9^{th} International Conference on Computer-aided Scheduling of Public Transport*, 2004.
7. S. Grava. *Urban Transportation Systems: Choices for Communities*. McGraw-Hill, 2002.
8. O. Ibarra-Rojas, F. López-Irarragorri, and Y. Rios-Solis. Multiperiod bus timetabling. *Transportation Science*, 50(3):805–822, 2016.
9. O. Ibarra-Rojas and Y. Rios-Solis. Synchronization of bus timetabling. *Transportation Research Part B: Methodological*, 46(5):599–614, 2012.
10. Intendencia de Montevideo. Plan de movilidad urbana: hacia un sistema de movilidad accesible, democrático y eficiente. [www.montevideo.gub.uy/aplicacion/plan-de-movilidad, 04/2019].
11. R. Massobrio. Urban mobility data analysis in Montevideo, Uruguay. Master's thesis, Universidad de la República, Uruguay, 2018.
12. S. Nesmachnow. Computación científica de alto desempeño en la Facultad de Ingeniería, Universidad de la República. *Revista de la Asociación de Ingenieros del Uruguay*, 61(1):12–15, 2010.
13. S. Nesmachnow. An overview of metaheuristics: accurate and efficient methods for optimisation. *International Journal of Metaheuristics,*, 3(4):320–347, 2014.
14. S. Nesmachnow, S. Baña, and R. Massobrio. A distributed platform for big data analysis in smart cities: combining intelligent transportation systems and socioeconomic data for Montevideo, Uruguay. *EAI Endorsed Transactions on Smart Cities*, 2(5):1–18, 2017.
15. S. Nesmachnow, A. Tchernykh, and A. Cristóbal. Planificación de transporte urbano en ciudades inteligentes. In *I Ibero-american Conference on Smart Cities*, pages 204–218, 2018.