Exact and Metaheuristic Approach for Bus Timetable Synchronization to Maximize Transfers

Sergio Nesmachnow, Jonathan Muraña(✉), and Claudio Risso

Universidad de la República, Montevideo, Uruguay
jmurana@fing.edu.uy

Abstract. This article presents the application of mathematical programming and evolutionary algorithms to solve a variant of the Bus Timetabling Synchronization Problem. A new problem model is proposed to include extended synchronization points, accounting for every pair of bus stops in a city, the transfer demands for each pair of lines, and the offset for lines in the considered scenario. Mixed Integer Programming and evolutionary algorithm are proposed to efficiently solve the problem. A relevant real case study is solved, for the public transportation system of Montevideo, Uruguay. Several scenarios are solved and results are compared with the no-synchronization solution and the current planning of such transportation system too. Experimental results indicate that the proposed approaches are able to significantly improve the current plannings. The Mixed Integer Programming algorithm computed the optimum solution for all scenarios, accounting for an improvement of up to 95% in successful synchronizations when compared with the actual timetable in Montevideo. The evolutionary algorithm is efficient too, improving up to 68% the synchronizations with respect to the current planning and systematically outperforming the baseline solutions. Waiting times for users are significantly improved too, up to 33% in tight problem instances.

Keywords: Smart cities · Mobility · Public transportation · Timetabling · Synchronization

1 Introduction

Transportation systems are a crucial component of modern society, and they are one of the most important services to improve efficiency of activities in nowadays smart cities [6,10]. Transportation systems include a wide range of logistic activities related to transporting passengers and goods. One of the main goals of transportation systems is coordinating the movement of people, providing efficient mobility at reasonable fares. In this regard, public transportation is the most efficient and environmental friendly mean for mobility of citizens. However, the efficacy of public transportation systems in large cities requires

S. Nesmachnow and L. Hernández Callejo (Eds.): ICSC-CITIES 2020, CCIS 1359, pp. 183–198, 2021.
https://doi.org/10.1007/978-3-030-69136-3_13

a proper planning of several issues that affect the quality of service, including routes design and management, timetabling, drivers assignment, and others [3].

A transportation system usually includes timetables accounting for reporting the expected location of vehicles during a day. Timetables are closely related to the transportation network design, and they are usually built to account for specific origin-destination demands. Synchronization of multi-leg trips or *transfers* is usually a secondary goal of the timetabling problem, although it is important for providing an adequate quality-of-service, allowing passengers to wait reasonable times for transfers from one route to another.

The proposed problem is very relevant for the case study proposed: the transportation system of Montevideo, Uruguay [13]. Montevideo has a rather uniform public transportation system, operated by buses with similar capacities and service provision. Many users of the system manage to complete their end-to-end journey using only one line, but several other users rely on connections between different lines to make their trips, using transfers. Transfers can be made between different (geographically separated) bus stops. They are allowed without additional charge and are controlled by the intelligent Metropolitan Transportation System (STM), which identifies users using personal smart cards.

The STM also maintains historical records of the mobility of users. From these data, time periods in the day are identified during which the use of the system is regular, that is, where the utilization numbers have little dispersion and their average values are known. These numbers include: number of passengers boarding or alighting, number of transfers between lines and combinations of stops, bus circulation times at stops on their routes, and transfer times between stops for passengers seeking to transfer. These data are the main inputs used by the local administration to plan the frequency of each line within the uniform periods. Even knowing the frequency of each line, that is, the number of buses to use in the service period to satisfy the demand of the system (including direct and transfer trips), and knowing the circulation times between stops, there is room to adjust the departure time of each bus, which in turn determines the arrival time of that bus at each stop on its route.

Considering the case study described above, the main goal of this article is to optimize the number of successful transfers allowed by a timetable realization. The types of transfer are identified, and for each one a maximum threshold is established for the time that a passenger waits for their connection. If the passenger manages to transfer within a waiting time below that threshold, the transfer is successfully timed. The variant of the transfer synchronization problem elaborated here studies how to coordinate the departure schedule of buses–and therefore of arrivals at stops on their routes–, in order to maximize the number of successful transfers during a uniform time period, for which all previous data are known and fixed.

The article is organized as follows. Section 2 introduces the bus synchronization problem and the variant solved in this article. Section 3 reviews related works. The proposed approaches for bus synchronization are described in Sect. 4. The experimental evaluation of the proposed methods over realistic instances in Montevideo is reported in Sect. 5. Finally, the conclusions and the main lines for future work are formulated in Sect. 6.

2 Bus Timetable Synchronization to Maximize Transfers

This section describes the bus timetable synchronization problem to maximize transfers.

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.

2.2 Problem Formulation

The mathematical formulation of the bus timetable synchronization problem to maximize transfers is presented next.

Problem Data. The set of data that defines an instance of the bus synchronization problem includes the following elements:

- A planning period $[0, T]$.
- A set of lines of the bus network $I = \{i_1, i_2, \ldots, i_n\}$, with predefined routes, and the number of trips f_i needed to fulfill the demand for each line i within the planning period $[0, T]$, accounting for both directs trips and transfers.

– A set of *synchronization nodes*, or *transfer zones*, $B = \{b_1, b_2, \ldots, b_m\}$. Each synchronization node $b \in B$ is a triplet $<i, j, d_b^{ij}>$ indicating that lines i and j may synchronize in b, and that the bus stops for lines i and j are separated by a distance d_b^{ij}. Each synchronization node represents a pair of bus stops for which regular transfers between lines i and j are registered. The value of d_b^{ij} defines the time needed for a passenger that transfers from line i to line j to walk from one stop to another in the transfer zone (see next item).
– A *traveling time function* $TT : I \times B \rightarrow \mathbf{Z}$. $TT_b^i = TT(i, b)$ indicates the time needed 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 *demand function* $P : I \times I \times B \rightarrow \mathbf{Z}$. $P_b^{ij} = P(i, j, b)$ indicates the number of passengers that transfer from line i to line j in synchronization node b, in the planning period. Assuming a uniform demand hypothesis in the planning period, the number of passengers that transfer from a given trip of line i to a given trip of line j is P_b^{ij}/f_i. This is a realistic assumption for planning periods where demand does not vary significantly, such as in the case study presented in this article.
– A maximum waiting time W_b^{ij} for each transfer zone, 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. Trips of line i and j are considered synchronized for transfers if and only if the waiting time of passengers that transfers is lower or equal to W_b^{ij}.
– The departing time of the first trip of each line i (the *offset* of the line) must be lower than a maximum headway time H_i, which is defined by the bus system operator. Subsequent trips depart at a fixed frequency ΔX^i. All trips of each line must start within the planning period $[0, T]$.

Mathematical Model. The bus synchronization problem proposes finding appropriate values for the departure time of the first trip of each line to guarantee the maximum number of synchronizations for all lines with transfer demands in the planning period T.

The control variables of the problem are the offset of each line (X_1^i), which define the whole set of departing times for all trips of each line. Auxiliary variables are needed to capture the synchronization events in each transfer zone. Binary variables Z_{rsb}^{ij} takes value 1 when trip r of line i and trip s of line j are synchronized in node b (i.e., trip r of line i arrives before trip s of line j and allows passengers to complete the transfer, i.e., walk between the corresponding bus stops and wait less than the waiting threshold for that transfer, W_b^{ij}).

The mathematical model of the bus synchronization problem as Mixed Integer Programming (MIP) problem is formulated in Eq. 1.

$$\text{maximize} \quad \sum_{b \in B} (\sum_{r=1}^{f_i} \sum_{s=1}^{f_j} Z_{rsb}^{ij}) \cdot \frac{P_b^{ij}}{f_i} \tag{1a}$$

$$\text{subject to} \quad Z_{rsb}^{ij} \leq 1 + \frac{(A_{rb}^i + d_b^{ij} + W_b^{ij}) - A_{sb}^j}{M} \; \forall b \in B \tag{1b}$$

$$Z_{rsb}^{ij} \leq 1 + \frac{A_{sb}^j - (A_{rb}^i + d_b^{ij})}{M} \; \forall b \in B \tag{1c}$$

$$\text{with } A_{sb}^j = X_1^j + (s-1)\Delta X^j + TT_b^j$$

$$A_{rb}^i = X_1^i + (r-1)\Delta X^i + TT_b^i$$

$$Z_{rsb}^{ij} \in \{0,1\}, 0 \leq X_1^i \leq H_i, \; \forall i \in I \tag{1d}$$

The objective function of the optimization problem (Eq. 1a) proposes maximizing the number of passengers that successfully complete a transfer in the planning period in every synchronization point. The value $\sum_{r=1}^{f_i} \sum_{s=1}^{f_j} Z_{rsb}^{ij}$ is the total number of successful connections between trips of each pair of lines i and j involved in each synchronization point b, while P_b^{ij}/f_i is the demand for each transfer.

Equations 1b–1d specify the constraints of the problem. According to Eq. 1a, the optimization will seek to activate as many variables Z_{rsb}^{ij} as possible. Constraints for variables Z_{rsb}^{ij} prevent them from taking the value 1 if the corresponding transfer is not synchronized. In both, Eqs. 1b and 1c, A_{rb}^i denotes the arrival time of trip r of line i to transfer zone b and A_{sb}^j denotes the arrival time of trip s of line j to transfer zone b. For an interpretation of constraint 1b, consider the maximum time passengers from trip r of line i are willing to wait for a transfer with trip s of line j at transfer zone b. This value defines the limit time $A_r^i + d_b^{ij} + W_b^{ij}$. Whenever the arrival time of trip s of line j does not surpass that limit, the right-hand side of Eq. 1b is greater or equal to 1, so the synchronization variable Z_{rsb}^{ij} is allowed to be 1. In addition, it is also necessary for passengers alighting from trip r of line i to walk to the transfer point (arriving at time $A_{rb}^i + d_b^{ij}$) before the arrival time of the corresponding trip s of line j (A_{sb}^j). Otherwise, those passengers would lose the connection. Whenever this second condition is met, the right-hand side of constraints Eq. 1c also allow Z_{rsb}^{ij} to take the value 1. So far, there is a potential issue when non-synchronized trips lead to values lower than 0 on the right-hand side of Eq. 1c, which derives into unfeasible constraints sets. The proposed model only needs that either $(A_{rb}^i + d_b^{ij} + W_b^{ij}) - A_{sb}^j$ or $A_{sb}^j - (A_{rb}^i + d_b^{ij})$ to be negative to deactivate synchronization variables Z_{rsb}^{ij}. Hence, suffices to get a constant value M, large enough to guarantee that both Eqs. 1b and 1c are always feasible. However, using extremely large values for M might cause numerical stability problems when the model is implemented in a solver. The procedure applied in this article to find compliant and relatively low values for M consisted in computing the maximum value within the union of sets $\{(H_i(j) + (f_i(j) - 1) \times \Delta X^j + TT_b^j) - (TT_b^i + d_b^{ij} + W_b^{ij})\}$ and $\{(H_i(i) + (f_i(i) - 1) \times \Delta X^i + TT_b^i + d_b^{ij}) - TT_b^j\}$, for all synchronization points $b\ inB$. These values of M can be easily calculated during the process of crafting the MIP formulation before using a specific solver, so the problem of finding M is of polynomial complexity. Finally, Eq. 1d defines the domain for decision variables Z_{rsb}^{ij} (binary variables).

The problem formulation assumes, without loss of generality, that $\Delta X^j > W_b^{ij}$, $\forall j \in I$, i.e., headways of bus lines are larger than the waiting time thresholds for users. The case where $\Delta X^j \leq W_b^{ij}$ correspond to a scenario in which the headway of line j is lower than the time users are willing to wait, thus all transfer with line j would be synchronized and they would not be part of the problem to solve.

3 Related Work

The bus timetable synchronization problem was recognized as a relevant issue for modern public transportation systems in early works by Ceder [3]. One of the first approaches for schedule synchronization on bus network systems was presented by Daduna and Voß [4], studying several objective functions (e.g., weighted sum considering transfers and the maximum waiting time at a transfer zone). Metaheuristic algorithms were evaluated for simple versions of the problem with uniform frequencies, using data from the Berlin Underground network and other German cities. Tabu Search computed better solutions than Simulated Annealing over randomly generated examples, and a trade-off between operational costs and user efficiency was concluded.

Ceder et al. [2] studied the Transit Network Timetabling problem to optimize the number of synchronization events between bus lines at shared stops, by maximizing the number of simultaneous arrivals. A greedy algorithm was proposed to solve the problem, based on selecting specific nodes from the bus network to define custom timetables. The article focused on simultaneous bus arrivals, and just some examples to illustrate synchronizations on small instances with few nodes and few lines were reported.

Fleurent et al. [5] proposed a subjective metric to evaluate synchronizations, using weights defined by experts and public transport authorities. The authors solved an optimization problem to minimize variable (vehicle) operation costs. A heuristic method was proposed for optimization, using the defined synchronization metric. Several timetables were computed for small scenarios from Montréal, Canada, using different weights for costs.

Ibarra and Ríos [8] studied a flexible variant of the synchronization problem, considering time windows between travel times. A Multi-start Iterated Local Search (MILS) algorithm was applied to solve eight instances modeling the bus network in Monterrey, Mexico with between three and 40 synchronization points. MILS was able to compute efficient solutions for medium-size instances in less than one minute, when compared with a simple upper bound and a Branch &

Bound exact method. Later, Ibarra et al. [7] applied MILS to solve the multi-period bus synchronization problem, to optimize multiple trips of a given set of lines. MILS was able to compute similar results than a Variable Neighborhood Search and a simple population-based algorithm on synthetic instances with few synchronization points. Results for a sample case study using data for a single line of Monterrey demonstrated that maximizing synchronizations for a specific node usually reduces the number of synchronizations for other nodes.

Our previous article [12] proposed an evolutionary approach for a specific variant of the bus synchronization problem. Results for realistic case studies in Montevideo demonstrated that the evolutionary approach outperformed real timetables by the city administrator and other heuristic methods. This article extends our previous research, accounting for a different variant of the bus synchronization problem aimed at determining the optimal offset values while keeping the headways and number of trips as indicated by the real timetable, in order to not impact in the quality of service offered to direct passengers.

4 Proposed Resolution Approaches

This section describes the exact and metaheuristic approaches developed to solve the bustimetable synchronization problem to maximize transfers.

4.1 Exact Mathematical Programming

The exact resolution of the proposed MIP model was developed using AMPL.

IBM ILOG CPLEX was used as the optimization tool, over the environment defined by ptimization Studio 12.8. Optimal solutions are computed applying a branch-and-cut heuristic, considering the following stop conditions for the execution:

- The time limit for the execution (parameter `CPX_PARAM_TILIM`) was set to ... (explicar: not relevant)
- The GAP tolerance in CPLEX (parameter `CPX_PARAM_EPGAP`) was set to the default value of 0.01% (0.0001). It is considered the default value since, the main goal is to compare with previous solutions obtained with that specific limit. The GAP represents, in percentage terms, the distance between the solution found and the best achievable solution. It is defined as $(f(x) - bestBound)/f(x)$, where x is the solution found and $bestBound$ is the best value achievable by the objective function.

4.2 Evolutionary Algorithm

The proposed EA was implemented in C++, using the Malva library (`github.com/themalvaproject`).

Solution Encoding. Candidate solutions to the problem are represented using integer vectors. In a solution representation, each integer value represents the offset (in minutes) of each bus line, i.e., the time between the start of the planning period and the depart of the first trip of each line. Formally, a candidate solution to the problem is represented by $X = X_0^1, X_0^2, \dots X_0^n$, where n is the number of bus lines in the problem instance, $X_0^i \in \mathbf{Z}^+$, and $0 \leq X_0^i \leq H^i$.

Evolution Model. The $(\mu+\lambda)$ evolution model [1] is applied in the proposed EA: μ parents generate λ offsprings, which compete between them and with their parents, to determine the individuals that will be part of 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 random initialization operator is applied. Randomly generated solutions are included in the initial population, accounting for the constraints defined for the offset of each line. This initialization procedure intends to provide diversity to the evolutionary search.

Selection Operator. A tournament selection is applied. The tournament size is three individuals, and one individual survives. Tournament selection computed better results than proportional selection in preliminary calibration experiments, mainly due to the appropriate level of selection pressure for the evolution.

Recombination Operator. The recombination operator is a specific variant of two-point crossover. It defines two crossover points randomly in $[1, n-1]$ 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 in the proposed representation with a probability p_M.

5 Experimental Evaluation

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

5.1 Methodology

Problem Instances. The experimental evaluation of the proposed methods for bus synchronization is performed in problem instances built using real data from the Metropolitan Transportation System in Montevideo, Uruguay.

Several sources of data from the National Open Catalog were considered to gather information about bus lines description, routes, timetables, and bus stops location in the city. The information about transfers was provided by Intendencia de Montevideo and processed applying a urban data analysis approach [9].

The key elements of the scenario and problem instances are described next: the period is the interval of hours considered for the schedule; the demand function is computed from transfers information registered by smart cards used to sell tickets; the synchronization points are chosen according to their demand, i.e., the pairs of bus stops with the largest number of registered transfers for the period are selected; the bus lines correspond to the lines passing by the synchronization points; the time traveling function TT for each line is computed empirically by using GPS data; the walking time function is the estimated walking speed of a person (assumed constant at $ws = 6\,\text{km/h}$) multiplied by the distance between bus stops in each transfer zone computed using geospatial information about stops. The maximum waiting time is equal to λH, with $\lambda \in [0.3, 0.5, 0.7, 0.9]$, to allow configuring instances with different levels of tolerance/quality of service.

Sixty problem instances were defined, accounting for three different dimensions (including 30, 70, and 110 synchronization points), using real information about bus operating in Montevideo, Uruguay. The synchronization points of each instance were chosen randomly from the most demanded transfer zones for the considered period in the city (a total number of 170 zones).

Each defined problem instance is identified by the following name convention: `[NP].[NL].[`λ`].[id]`, where $\mathsf{NP} = n$ is the number of synchronization points, $\mathsf{NL} = m$ is the of bus lines, λ is the coefficient applied to W_b (percentage) and id is a relative identifier for instances with the same values of NL, NP, and λ. Scenarios are available at https://www.fing.edu.uy/inco/grupos/cecal/hpc/bus-sync/.

Execution Platform. The experimental evaluation was performed on a Quad-core Xeon E5430 at 2.66 GHz, 8 GB RAM, from National Supercomputing Center (Cluster-UY), Uruguay [11].

Baseline Solutions for the Comparison. Two main baseline solutions were considered for the comparison of the solutions computed by the proposed methods. A relevant baseline for comparison is the current timetable applied in the transportation system of Montevideo (the *real* timetable), which provides the actual level of service regarding direct travels and transfers. In turn, another relevant baseline for comparison is the solution without applying any explicit approach for synchronization of transfers, i.e., a solution where the first trip of each line departs at the beginning of the planning period (time 0, the *zeros* timetable). This solution provides a number of synchronized transfers according to the predefined headways for each line.

Metrics. The metrics applied for the evaluation include: i) the number of synchronized trips for passengers, as proposed in the summatory that defines the objective function of the problem; ii) the improvements over the baseline solutions, iii) the average waiting time each passenger wait for the connection (bus of line j) in a synchronization point.

Parameter Setting. EAs are stochastic methods, thus parameter setting analysis are needed to determine the parameter 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 for the proposed EA on three instances, different from the ones used in validation experiments, in order to avoid bias. The best results were obtained with the configuration $\#gen = 10000$, $ps = 20$, $p_R = 0.9$ and $p_M = 0.01$.

5.2 Numerical Results

Table 1 reports the objective function values computed by EA and the exact resolution approach for the considered problem instances. In turn, the relative improvements over the baseline solutions are reported: Δ_r is the relative improvement over the real timetable and Δ_z is the relative improvement over the zeros solution.

Table 1. Objective function results of exact and EA

scenario	*real*	*zeros*	*EA*			*exact*		
			obj	Δ_r	Δ_z	*obj*	Δ_r	Δ_z
30.37.90.0	276.08	286.89	302.09	0.09	0.05	302.09	0.09	0.05
30.37.70.0	224.62	232.79	271.75	0.21	0.17	271.75	0.21	0.17
30.37.50.0	162.41	151.99	208.65	0.28	0.37	208.99	0.29	0.38
30.37.30.0	111.58	107.85	154.49	0.38	0.43	154.62	0.39	0.43
30.40.90.0	218.61	229.78	237.05	0.08	0.03	243.02	0.11	0.06
30.40.70.0	163.64	175.85	199.40	0.22	0.13	219.97	0.34	0.25
30.40.50.0	126.11	127.07	146.39	0.16	0.15	173.24	0.37	0.36
30.40.30.0	92.28	90.97	101.18	0.10	0.11	127.08	0.38	0.40
30.40.90.1	227.36	248.57	252.45	0.11	0.02	262.36	0.15	0.06
30.40.70.1	178.07	193.32	219.92	0.24	0.14	238.19	0.34	0.23
30.40.50.1	129.80	140.22	163.30	0.26	0.16	190.16	0.47	0.36
30.40.30.1	80.24	108.97	117.42	0.46	0.08	156.44	0.95	0.44
30.41.90.0	246.99	260.32	279.49	0.13	0.07	279.49	0.13	0.07
30.41.70.0	197.16	201.00	248.29	0.26	0.24	248.42	0.26	0.24
30.41.50.0	141.93	136.07	186.27	0.31	0.37	186.36	0.31	0.37
30.41.30.0	93.79	98.26	141.87	0.51	0.44	142.63	0.52	0.45
30.42.90.0	241.44	241.45	255.78	0.06	0.06	255.78	0.06	0.06
30.42.70.0	195.96	191.28	228.08	0.16	0.19	228.08	0.16	0.19
30.42.50.0	145.01	140.28	172.52	0.19	0.23	172.52	0.19	0.23
30.42.30.0	95.81	93.08	124.88	0.30	0.34	125.74	0.31	0.35
70.60.90.0	568.51	579.73	609.29	0.07	0.05	609.68	0.07	0.05
70.60.70.0	463.02	454.24	545.02	0.18	0.20	546.03	0.18	0.20
70.60.50.0	339.70	296.11	414.29	0.22	0.40	415.80	0.22	0.40

(*continued*)

Table 1. (*continued*)

scenario	*real*	*zeros*	*EA*			*exact*		
			obj	Δ_r	Δ_z	*obj*	Δ_r	Δ_z
70.60.30.0	218.71	213.86	301.34	0.38	0.41	304.07	0.39	0.42
70.62.90.0	543.67	560.98	590.80	0.09	0.05	591.17	0.09	0.05
70.62.70.0	443.22	443.89	524.86	0.18	0.18	525.81	0.19	0.18
70.62.50.0	325.70	317.75	393.66	0.21	0.24	394.13	0.21	0.24
70.62.30.0	212.04	215.66	295.31	0.39	0.37	298.68	0.41	0.38
70.63.90.0	550.17	575.71	609.44	0.11	0.06	609.68	0.11	0.06
70.63.70.0	441.71	455.92	546.46	0.24	0.20	547.61	0.24	0.20
70.63.50.0	316.46	300.67	427.58	0.35	0.42	429.74	0.36	0.43
70.63.30.0	208.82	202.82	324.20	0.55	0.60	328.02	0.57	0.62
70.67.90.0	510.16	535.04	567.09	0.11	0.06	567.43	0.11	0.06
70.67.70.0	409.57	418.35	512.30	0.25	0.22	513.00	0.25	0.23
70.67.50.0	302.15	299.78	400.06	0.32	0.33	402.49	0.33	0.34
70.67.30.0	194.05	201.63	298.87	0.54	0.48	302.55	0.56	0.50
70.69.90.0	522.36	550.33	583.61	0.12	0.06	583.85	0.12	0.06
70.69.70.0	406.63	435.12	529.19	0.30	0.22	531.05	0.31	0.22
70.69.50.0	298.53	292.98	416.07	0.39	0.42	418.24	0.40	0.43
70.69.30.0	193.05	205.18	324.08	0.68	0.58	328.15	0.70	0.60
110.76.90.0	815.78	843.26	894.86	0.10	0.06	895.72	0.10	0.06
110.76.70.0	656.63	669.18	798.41	0.22	0.19	799.61	0.22	0.19
110.76.50.0	479.29	451.85	622.49	0.30	0.38	627.71	0.31	0.39
110.76.30.0	333.66	294.90	467.28	0.40	0.58	474.09	0.42	0.61
110.78.90.0	847.33	879.90	900.09	0.06	0.02	931.35	0.10	0.06
110.78.70.0	699.49	667.77	763.71	0.09	0.14	835.26	0.19	0.25
110.78.50.0	507.05	453.87	551.42	0.09	0.21	644.97	0.27	0.42
110.78.30.0	324.79	317.42	379.43	0.17	0.20	477.47	0.47	0.50
110.78.90.1	867.23	895.99	910.82	0.05	0.02	941.85	0.09	0.05
110.78.70.1	708.89	689.30	780.93	0.10	0.13	845.71	0.19	0.23
110.78.50.1	525.85	460.71	567.71	0.08	0.23	654.75	0.25	0.42
110.78.30.1	338.42	319.33	390.82	0.15	0.22	489.02	0.45	0.53
110.78.90.2	848.47	872.66	894.37	0.05	0.02	932.92	0.10	0.07
110.78.70.2	681.46	676.82	765.56	0.12	0.13	836.45	0.23	0.24
110.78.50.2	494.80	449.03	571.35	0.15	0.27	654.02	0.32	0.46
110.78.30.2	333.19	309.13	397.57	0.19	0.29	492.76	0.48	0.59
110.83.90.0	810.76	850.69	897.02	0.11	0.05	897.65	0.11	0.06
110.83.70.0	624.94	674.88	803.28	0.29	0.19	806.25	0.29	0.19
110.83.50.0	463.61	460.48	634.77	0.37	0.38	639.36	0.38	0.39
110.83.30.0	299.88	300.96	490.14	0.63	0.63	498.28	0.66	0.66

Results reported in Table 1 indicate that exact and EA methods significantly outperform the baseline solutions in all studied scenarios. The improvements of EA over the real solution were up to 68% in instance `70.69.30.0` and the improvements of the exact method over the real solution were up to 95% in instance `30.40.30.1`. Regarding the comparison with the zeros solution, the improvements of EA were up to 63% and the improvements of the exact method were up to 66%, both in instance `110.83.30.0`. Average improvements over the real timetable were 20% for EA and 25% for the exact solution.

In turn, the proposed EA was able to compute solutions close to the exact method (i.e., the optimal value) in low dimension and high tolerance scenarios, computing the optimal solution in six scenarios.

Table 2 reports the average improvements of exact and EA over the baseline solutions, grouped by scenario size and tolerance. Improvements of the exact method are up to 52% over the real timetable (in scenarios with $\mathsf{NP} = 70$ and $\lambda = 30$) and up to 52% over zeros (in scenarios with $\mathsf{NP} = 100$ and $\lambda = 30$). In turn, the EA improved up to 50% over the real timetable and up to 49% over zeros, both in scenarios with $\mathsf{NP} = 70$ and $\lambda = 30$. The values grouped by tolerance allow concluding that for all sizes, the improvements increase as user tolerance decreases. This result indicates that the proposed methods scale with the complexity of the problem, effectively increasing the quality of service. Improvements of the exact method also increase with the size of the scenario.

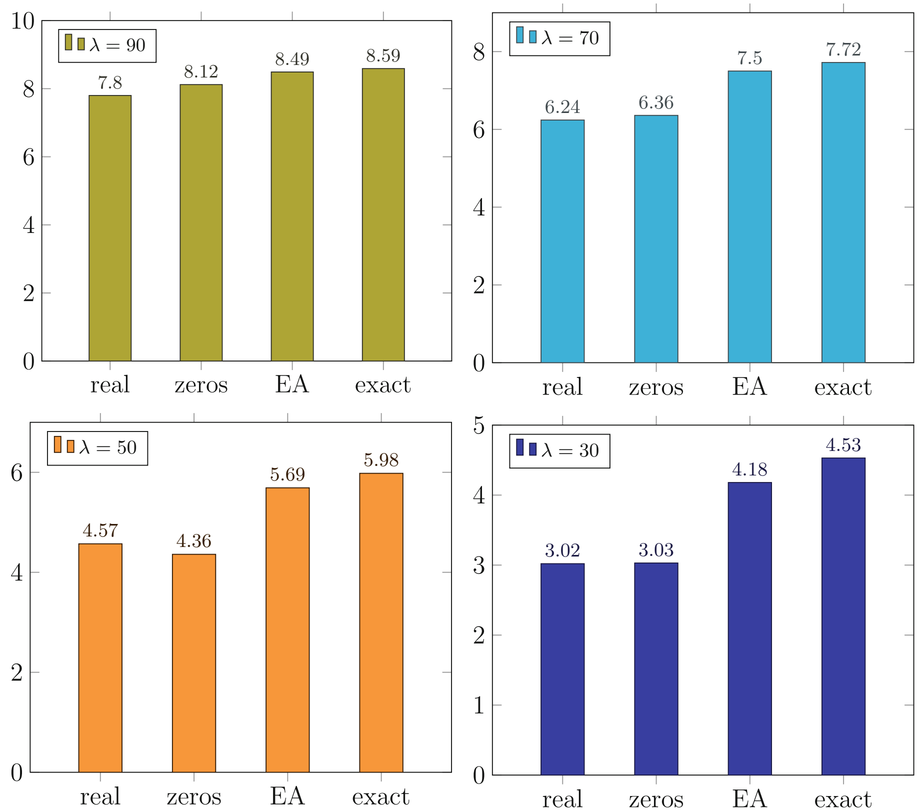

Fig. 1. Objective function comparison grouped by tolerance

Table 2. Improvements of exact and EA over baseline solutions, grouped by dimension and tolerance

NP	λ	*EA*		*exact*	
		Δ_r	Δ_z	Δ_r	Δ_z
30	90	0.10	0.05	0.11	0.06
30	70	0.22	0.17	0.26	0.21
30	50	0.24	0.26	0.32	0.34
30	30	0.35	0.28	0.49	0.42
70	90	0.10	0.06	0.10	0.06
70	70	0.23	0.20	0.23	0.21
70	50	0.30	0.36	0.30	0.37
70	30	0.50	0.49	0.52	0.50
110	90	0.07	0.04	0.10	0.06
110	70	0.16	0.16	0.22	0.22
110	50	0.19	0.30	0.30	0.42
110	30	0.30	0.38	0.49	0.58

Figure 1 shows the average objective values (normalized by NP) for all scenarios, grouped by tolerance. The largest difference in objective values is 1.51 (4.53–3.02), between the exact approach and the real timetable in scenarios with low user tolerance ($\lambda = 30$). The lowest difference is 0.37, between EA and zeros, when λ=90. As for results in Table 2, the graphic clearly shows that improvements of the proposed approaches increase for tight scenarios.

Table 3 reports three values ($r - l_s/l_n$) for the considered solutions, grouped by *NP* and λ. The value r is the ratio of the average waiting time results over the maximum waiting time for each synchronization point, which evaluates the number of successful synchronized trips and the relative waiting time for each synchronization point. Successful synchronization are represented by $r \leq 1.0$, and unsuccessful synchronization are represented by $r > 1.0$. In turn, l_s is the average number of lines successfully synchronized and l_n is the average number of lines not synchronized.

Table 3. Average waiting time results for the considered solutions

NP	λ	*real*	*zeros*	*EA*	*exact*
30	90	0.47–22/0	0.48–22/0	0.46–22/0	0.47–22/0
30	70	0.59–22/0	0.61–21/1	0.54–22/0	0.53–22/0
30	50	0.85–16/6	0.87–16/6	0.76–19/3	0.75–19/3

(*continued*)

Table 3. (*continued*)

NP	λ	*real*	*zeros*	*EA*	*exact*
30	30	1.33–3/19	1.37–3/19	1.19–5/16	1.12–7/15
70	90	0.47–39/0	0.49–39/0	0.46–39/0	0.47–39/0
70	70	0.59–38/1	0.62–38/1	0.54–39/0	0.52–39/0
70	50	0.85–28/10	0.88–28/10	0.74–35/3	0.73–35/4
70	30	1.35–5/33	1.40–4/35	1.14–11/28	1.13–12/27
110	90	0.49–49/0	0.50–49/0	0.48–49/0	0.46–49/0
110	70	0.61–46/2	0.64–47/2	0.57–48/1	0.53–49/0
110	50	0.87–34/14	0.91–34/15	0.79–41/8	0.72–44/5
110	30	1.38–7/42	1.43–4/45	1.24–11/38	1.10–17/32

Results in Table 3 indicate that the proposed approaches significantly improve the quality of service with respect to the baseline solutions, accounting for lower values of the waiting time metric for all scenarios. Largest improvement of EA over baseline solutions occur where $\mathsf{NP} = 70$ and $\lambda = 30$ (0.26 over zeros solution and 0.21 over real solution). Largest improvements of the exact method occur where $\mathsf{NP} = 110$ and $\lambda = 30$ (0.33 over zeros solution and 0.28 over real solution). The proposed approaches achieve better waiting time values in lower tolerance scenarios, with respect to baseline solutions.

Figure 2 presents a histogram comparison of the waiting times (normalized by W_b) for bus lines of a sample scenario (`70.63.30.2`) for a baseline solution (left) and the exact solution (right). The graphic shows that the exact solution manages to reduce the waiting time in a significant percentage of lines in the scenario. The exact solution has more bus lines with a waiting time less than or equal to 1 (16 vs. 6). Also, the exact solution has two lines with wait less than 0.5 while the baseline solution has none. Regarding higher waiting times, in the exact solution only 6 lines are higher than 1.5 of the expected value, while the baseline solution has 13 bus lines where users wait more than 1.5 of the expected value. The histogram comparison clearly indicates that the proposed solution improves the QoS, by synchronizing a larger number of lines of the system.

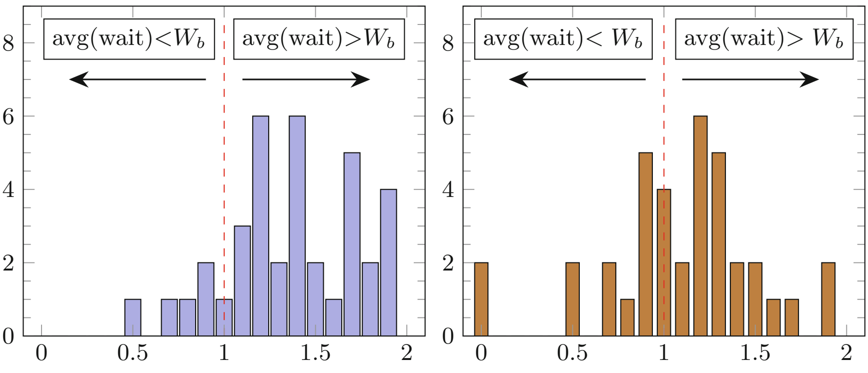

Fig. 2. Histogram comparison of the waiting time metric for baseline (left) and exact solutions (right) in scenario `70.63.30.2`

6 Conclusions and Future Work

This article presented exact and evolutionary approaches to solve a variant of the Bus Timetabling Synchronization Problem considering extended synchronization points for every pair of bus stops in a city, transfer demands, and the line offsets.

A Mixed Integer Programming approach and an evolutionary algorithm were proposed to efficiently solve the problem. Results were compared with the no-synchronization solution and also with real timetables for a real case study in Montevideo, Uruguay.

Experimental results indicate that the proposed approaches significantly improve over current timetable. The exact method computed the optimum solution for all scenarios, improving successful synchronizations up to 95% (25% in average) over the real timetable in Montevideo. The EA is efficient too, improving up to 68% the synchronizations (20% in average) over the current timetable and systematically outperforming other baseline solutions. The proposed EA can be useful for addressing larger scenarios of the considered problem. Waiting times for users are significantly improved too, up to 33% in tight problem instances.

The main lines of future work include solving different variants of the bus timetable synchronization problem, accounting for different headways in the planning period, and modeling the real demand for direct trips too. Multiobjective version of the problem must be included too, by considering other relevant functions: cost and quality of service.

References

1. Bäck, T., Fogel, D., Michalewicz, Z. (eds.): Handbook of Evolutionary Computation. Oxford University Press, New York (1997)
2. Ceder, A., Golany, B., Tal, O.: Creating bus timetables with maximal synchronization. Transp. Res. Part A: Policy Pract. **35**(10), 913–928 (2001)
3. Ceder, A., Wilson, N.: Bus network design. Transp. Res. Part B Methodol. **20**(4), 331–344 (1986)
4. Daduna, J., Voß, S.: Practical experiences in schedule synchronization. In: Daduna, J.R., Branco, I., Paixão, J.M.P., (eds.) Computer-Aided Transit Scheduling. Lecture Notes in Economics and Mathematical Systems, vol. 430, pp. 39–55. Springer, Berlin (1995) https://doi.org/10.1007/978-3-642-57762-8_4
5. Fleurent, C., Lessard, R., Séguin, L.: Transit timetable synchronization: Evaluation and optimization. In: 9^{th} International Conference on Computer-aided Scheduling of Public Transport (2004)
6. Grava, S.: Urban Transportation Systems. McGraw-Hill, New York (2002)
7. Ibarra-Rojas, O., López-Irarragorri, F., Rios-Solis, Y.: Multiperiod bus timetabling. Transp. Sci. **50**(3), 805–822 (2016)
8. Ibarra-Rojas, O., Rios-Solis, Y.: Synchronization of bus timetabling. Transp. Res. Part B Methodol. **46**(5), 599–614 (2012)
9. Massobrio, R., Nesmachnow, S.: Urban mobility data analysis for public transportation systems: a case study in Montevideo. Uruguay Appl. Sci. **10**(16), 5400 (2020)

10. Nesmachnow, S., Baña, S., Massobrio, R.: A distributed platform for big data analysis in smart cities: combining intelligent transportation systems and socioeconomic data for Montevideo, Uruguay. EAI Endorsed Trans. Smart Cities **2**(5), 1–18 (2017)
11. Nesmachnow, S., Iturriaga, S.: Cluster-UY: collaborative scientific high performance computing in Uruguay. In: Torres, M., Klapp, J. (eds.) ISUM 2019. CCIS, vol. 1151, pp. 188–202. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-38043-4_16
12. Nesmachnow, S., Muraña, J., Goñi, G., Massobrio, R., Tchernykh, A.: Evolutionary approach for bus synchronization. In: Crespo-Mariño, J.L., Meneses-Rojas, E. (eds.) CARLA 2019. CCIS, vol. 1087, pp. 320–336. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-41005-6_22
13. Risso, C., Nesmachnow, S.: Designing a backbone trunk for the public transportation network in Montevideo, Uruguay. In: Nesmachnow, S., Hernández Callejo, L. (eds.) ICSC-CITIES 2019. CCIS, vol. 1152, pp. 228–243. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-38889-8_18