Transportmetrica
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/ttra20

Hybrid predictive control strategy for a public transport system with uncertain demand

Doris Sáez [a] , Cristián E. Cortés [b] , Freddy Milla [a] , Alfredo Núñez [a] , Alejandro Tirachini [c] & Marcela Riquelme [a]

[a] Electrical Engineering Department, Universidad de Chile, Av. Tupper 2007, Santiago, Chile

[b] Civil Engineering Department, Universidad de Chile, Blanco Encalada 2002, Santiago, Chile

[c] Institute of Transport and Logistics Studies, Faculty of Economics and Business, The University of Sydney, Sydney, NSW 2006, Australia

Available online: 18 Nov 2010

Transportmetrica
2010, 1–26, iFirst

Hybrid predictive control strategy for a public transport system with uncertain demand

Doris Sáez[a]*, Cristián E. Cortés[b], Freddy Milla[a], Alfredo Núñez[a], Alejandro Tirachini[c] and Marcela Riquelme[a]

[a]*Electrical Engineering Department, Universidad de Chile, Av. Tupper 2007, Santiago, Chile;* [b]*Civil Engineering Department, Universidad de Chile, Blanco Encalada 2002, Santiago, Chile;* [c]*Institute of Transport and Logistics Studies, Faculty of Economics and Business, The University of Sydney, Sydney, NSW 2006, Australia*

(*Received 15 October 2008; final version received 13 January 2010*)

In this article, a hybrid predictive control (HPC) strategy is formulated for the real-time optimisation of a public transport system operation run using buses. For this problem, the hybrid predictive controller corresponds to the bus dispatcher, who dynamically provides the optimal control actions to the bus system to minimise users' total travel time (on-vehicle ride time plus waiting time at stops). The HPC framework includes a dynamic objective function and a predictive model of the bus system, written in discrete time, where events are triggered when a bus arrives at a bus stop. Upon these events, the HPC controller makes decisions based on two well-known real-time transit control actions, holding and expressing. Additionally, the uncertain passenger demand is included in the model as a disturbance and then predicted based on both offline and online information of passenger behaviour. The resulting optimisation problem of the HPC strategy at every event is *Np*-hard and needs an efficient algorithm to solve it in terms of computation time and accuracy. We chose an *ad hoc* implementation of a Genetic Algorithm that permits the proper management of the trade-off between these two aspects. For real-time implementation, the design of this HPC strategy considers newly available transport technology such as the availability of automatic passenger counters (APCs) and automatic vehicle location (AVL) devices. Illustrative simulations at 2, 5 and 10 steps ahead are conducted, and promising results showing the advantages of the real-time control schemes are reported and discussed.

Keywords: hybrid predictive control; public transport; holding; station skipping

1. Introduction

The basic design variables required to set up a public transport system using buses are the number of lines and their associated routes, the fleet composition of each line and the optimal frequency associated with each line. These factors should all be strongly related to passenger demand intensity and distribution, according to the most demanding periods for a typical day of operation (peak periods). Moreover, the frequency of operation and the

*Corresponding author. Email: dsaez@ing.uchile.cl

ISSN 1812–8602 print/ISSN 1944–0987 online

DOI: 10.1080/18128601003615535
http://www.informaworld.com

associated pre-planned schedule must be set differently for various established periods, while still assuming an average behaviour (deterministic) over each period. Unfortunately, in most cases the movement of buses is affected by different disruptions as the day progresses, such as traffic congestion, unexpected delays, randomness in passenger demand (both spatial and temporal), irregular vehicle dispatching times, incidents and so on. These events hinder the dispatch of buses as well as in-route bus operations when following a pre-planned schedule defined at an aggregated level (average) over each period of operation. As an attempt to reduce the negative effects of service disturbance, researchers have devoted significant effort to developing flexible control strategies, either in real-time or offline, depending on the specific features of the problem.

Historically, the literature in this field was evolved from the study of pre-planned fleet assignment and scheduling strategies, to the analysis of real-time control strategies, assuming that real-time information is available through on-vehicle equipment such as automatic passenger counters (APCs) and automatic vehicle location (AVL) devices. The first group of strategies works as a complement to a properly pre-planned bus schedule, since they are defined to deal with well-known demand imbalances at an aggregated level, in specific route sections and periods (for a deep description of these strategies, such as short turning, restricted zonal service and deadheading, amongst others, see Furth and Day (1984)). The second group of strategies has been designed to allow the operator to react dynamically to real-time system disturbances. The most studied strategy of this type in recent years is the holding strategy, in which vehicles are held at specific stations for a certain time, in most cases oriented to keep the headway between successive buses as close as possible to a predefined value.

In terms of the spatial configuration of the different control strategies, Eberlein (1995) classified them into three categories: station control, inter-station control and other strategies. Station control strategies are of two types: holding and station-skipping (deadheading, expressing, short-turning, etc.). Inter-station control strategies include speed control and transit signal priority, amongst others. Other strategies include, for example, train-splitting, which is more oriented to the rail systems control literature.

With regard to the most remarkable contributions in the study of the holding strategy, we can mention Barnett (1974), Turnquist and Blume (1980), Eberlein (1995), Eberlein *et al.* (2001), Hickman (2001), Sun and Hickman (2004), Zolfaghari *et al.* (2004) and Yu and Yang (2007). Barnett (1974) developed a simple holding model at a given control station, where the sum of total waiting time plus the extra delay of passengers on board deadheaded vehicles is minimised. Turnquist and Blume (1980) identified conditions under which holding results are attractive. The study of Hickman (2001) presented a stochastic holding model at a given control station. The author formulated a convex quadratic program in a single variable corresponding to the time lapse during which buses are held. More recent research has explored holding models that rely on real-time information, mainly referring to vehicle location (Eberlein 1995, Eberlein *et al.* 2001, Hickman 2001, Sun and Hickman 2004). Eberlein (1995) and Eberlein *et al.* (1999, 2001) postulated deterministic quadratic programs under a rolling horizon scheme, in which the holding decision for a specific vehicle affected the operation of a specific subset of the precedent vehicles. The authors concluded that having two or more holding stations over a corridor is not necessary. These results contradicted those of Sun and Hickman (2004). Their paper concluded that holding multiple vehicles at multiple control stations would be better than having a single holding station. Most of these models propose heuristics to solve the problems due to the mathematical complexity of the formulations. Zolfaghari *et al.* (2004) developed a mathematical control model for holding using real-time information of locations of buses along a specified route. Waiting times are computed based on the difference of departure times of buses and the optimisation problem is finally solved with simulated annealing. Finally, Yu and Yang (2007) present a dynamic holding strategy, in which the on-time performance of the early bus operation at the next stop is considered and the holding times of the held bus at the stop is optimised. A model based on support vector machine (SVM) for forecasting the early bus departure times from the next stop is also developed. Furthermore, in order to determine the optimal holding times, a model aiming to minimise the total user costs is developed. Genetic algorithms are proposed to optimise holding times.

The operation of express services (expressing) has been studied as a pre-planned strategy (Jordan and Turnquist 1979, Furth 1986) and, more extensively, as a real-time control strategy (Eberlein 1995, Lin *et al.* 1995, Eberlein *et al.* 1999, Fu *et al.* 2003, Sun and Hickman 2005). In the latter case, the approach consists of speeding up buses by skipping stations (one or more) to recover their pre-planned schedule after a disruption or unexpected delay, in order to reduce the impact on the level of service measured by total waiting time of users at stations plus the extra waiting time of passengers whose station has been skipped. In general, a station-skipping decision is made before the buses depart from the terminal, except in the model by Sun and Hickman (2005), who allowed the control action to be taken once the vehicle is moving. The authors consider the first and last stations of the skipped segment as variables, finding many situations in which a strategy that allows buses to stop at a skipped station if there are passengers who need to get off there (allowing some passengers to get on the bus as well) outperforms the basic strategy, where passengers whose destination is inside the skipped segment are forced to get off before their desired station.

Eberlein (1995) formulated an integrated model, which encompassed holding, deadheading and expressing. Additionally, Adamski and Turnau (1998) and Adamski (1996) developed a simulation decision-support tool for dynamic optimal dispatching control, including punctuality control (which compensates for deviations from the schedule), regularity control (which compensates for deviations from regular headway) and synchronising control based on the Linear Quadratic feedback control, while considering system state constraints. They also performed a Linear Quadratic stochastic control with real-time estimation of the model parameters and presented the results using numerical examples.

In the present article, we develop a model integrating two of the aforementioned strategies (holding and expressing) to solve a real-time public transport control problem with uncertain passenger demand, relying on online information of system behaviour. In a way similar to Sun and Hickman (2005), in our model the decision of skipping stations is made in real-time, which makes the proposed framework more adaptable and responsive to real-time delays. The model is formulated as a hybrid predictive control (HPC) problem, since the underlying theory fit nicely into the dynamic conditions of typical public transport problems. Predictive models permit the estimation of future effects of the control actions on the behaviour of the bus system and also allow the inclusion of complex system constraints. They also have the capability of optimising system performance in real-time based on a properly chosen objective function (Hegyi 2004, Hegyi *et al.* 2005, Karer *et al.* 2007a,b). Moreover, predictive approaches are suitable for dynamic environments with high uncertainty of future events, which can become relevant for the decision making process that has to be performed in real-time (Cortés *et al*. 2008, Sáez *et al*. 2008).

Specifically in this research, we propose to design and evaluate a predictive control strategy for a bus system with uncertain passenger arrival at bus stops, under a real-time framework. For this problem, the predictive controller corresponds to the bus dispatcher, who dynamically provides the optimal control actions to the bus system in order to optimise the performance according to an assumed objective function that takes into account the future evolution of the public transport system. In this particular case, the dynamic objective function is mostly oriented to minimising users' total travel time (including waiting time as well as in-vehicle ride time). A more complex model would also include operational cost; however, for the kind of strategies we are considering in this approach (holding and expressing), it seems that the dynamic operator component is not as important as if we had not included strategies involving injection of additional buses in certain route sections; this component is part of ongoing research.

The real-time passenger demand, which is unknown and uncertain, is modelled as a disturbance for the predictive scheme, because different passenger arrival patterns could significantly affect the estimated in-vehicle ride time from longer passenger transfer operations at bus stops. The control strategies will allow us to incorporate into the design the future behaviour of the whole system associated with the operation of the buses, by using a prediction system for the disturbances (demand). The methodology for predicting the demand needs a good estimation of the origin–destination real-time matrix to properly predict the future number of passengers at stops, bus loads and passenger transference time lapses, based on both offline and online data.

Several authors in the public transport real-time control literature designed their strategies based on a typical objective function of regularising the headway between buses, which is found on a total users' waiting time minimisation objective, since the waiting time linearly increases with headway variance (Welding 1957, Osuna and Newell 1972). As discussed before, for the design of the predictive controller, a proper objective function is defined considering the in-vehicle travel and waiting time spent by all users affected by the implementation of the strategies during their trip. Thus, the waiting time component is explicitly included in our formulation, along with a component oriented to maintain the headway as close as possible to the design headway (see Section 2.4 for details).

In the next section, the design of the HPC is described. As explained in Section 3, we propose to use Genetic Algorithms (GA) for efficiently solving the resulting HPC optimisation problem. Next, in Section 4 we describe illustrative simulations at 2, 5 and 10 steps ahead. Finally, in Section 5 the major conclusions of the work are presented.

2. HPC design for a dynamic public transport system

2.1. *Problem statement*

The optimisation of the real-time operations associated with a bus system is formulated under a HPC approach. Both the objective function and the predictive model are essential for HPC design. For the sake of simplicity, in this work the HPC framework is constructed for a single loop bus system, although it could be extended to more complex systems according to a similar modelling framework. The system is represented in Figure 1.

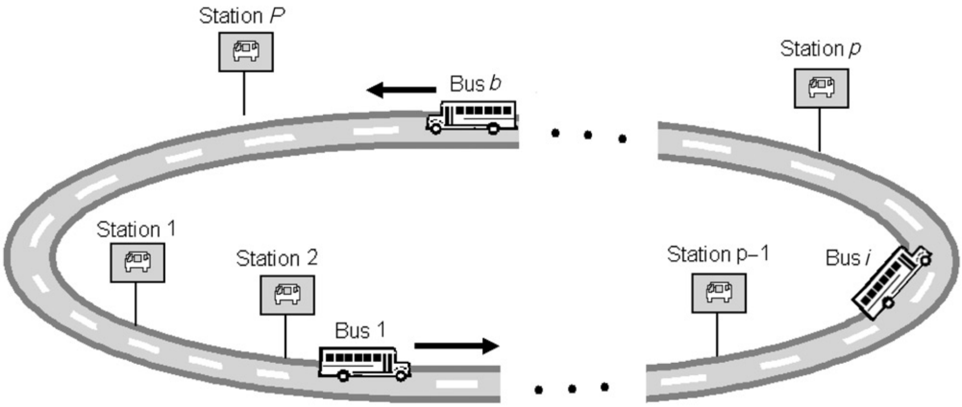

Figure 1. Public transport system.

The network is a one-way loop route, with P equidistant stops and b buses running around the loop, under the control of the dispatcher.

Passengers arrive at each station at a certain rate by following a negative exponential distribution, with destinations randomly chosen among the stations ahead of the station where the passenger is boarding. Then, every passenger is characterised by origin and destination bus stops and by the time the passenger arrives at the stop, without including the time spent by the passenger in getting to the bus stop. From historical data, a representative stop-to-stop demand matrix can be estimated for each modelling period; this is crucial for adding the predictive feature in the real-time model of the system. Online demand data can also be used as a complement to the offline demand matrix to improve this predictive aspect.

In our approach, there are discrete (number of passengers on buses) as well as continuous (bus position and speed) variables. For this reason, we decided to use a HPC approach, in which the optimisation of the control actions considering both kinds of variables can be performed (Bemporad and Morari 1999). The problem is then formulated as a hybrid system, where events are triggered by specific actions. Unlike traditional HPC formulations written for a fixed step-size, this formulation results in a variable step-size, since the problem scheme is based on relevant system events (corresponding to the instants at which control actions must be taken). The events are triggered when a bus arrives at a bus stop, which determines a variable time-step. Hereafter, we denote t as the continuous time, k as the event and t_k as the continuous time at which event k occurs. Note that an event k is always associated with the arrival of a specific bus i to a specific bus stop p.

One major feature of this particular HPC approach, different from typical HPC schemes, is the double dimensionality of this specific dynamic modelling framework: spatial and temporal. Figure 2 shows the closed loop of the bus system and the corresponding main variables, which are functions of continuous and discrete time. When an event k occurs, the hybrid predictive controller generates control actions and then, the outputs are obtained. The variables defined in continuous time, such as bus position and speed, are required to keep track of some system characteristics when an event is triggered (e.g. position of all vehicles when one specific bus arrives at a bus stop).

For every bus i belonging to the fleet, its position at any continuous instant t, $x_i(t)$, and the remaining time for the bus i to reach the next stop, $T_i(t)$, are defined in order to check the buses' status and consequently trigger the events. A new event k is triggered by bus i at any stop p when $x_i(t)$ matches the position of this stop at $t = t_k$. Therefore, the remaining time for the bus i to reach this stop is equal to zero ($T_i(t_k) = 0$).

The manipulated variables are the holding $h_i(k)$ and the station-skipping $Su_i(k)$ actions associated with bus i and event k. Thus, $h_i(k)$ is the lapse during which bus i is held at the stop associated with event k, while $Su_i(k)$ is a binary variable that is equal to one if passengers are allowed to board bus i at the stop associated with event k, zero otherwise. The discrete time output variables correspond to the passenger load $L_i(k+1)$ and the departure time $Td_i(k+1)$ once the bus departs from its current stop, associated with the bus i that triggered event k.

In Figure 2, the variable $\Gamma_p(k)$ is the number of passengers waiting for a bus at stop p and corresponds to a system disturbance. By means of a demand estimator, the variables $\hat{A}_i(k)$, $\hat{B}_i(k)$ and $\hat{\Gamma}_p(k+1)$ are estimated and incorporated to the dynamic model. $\hat{\Gamma}_p(k+1)$ is the prediction of the number of passengers when bus i departure from stop p, $\hat{B}_i(k)$ is the expected number of passengers that will board bus i at event k and $\hat{A}_i(k)$ represents the estimated number of passenger alighting from bus i at event k.

With regard to the inputs of the dynamic model, these correspond to the control action variables, and are analytically defined as follows:

- $Su_i(k)$: Passenger boarding action of bus i, at instant k (associated with expressing),

$$Su_i(k) = \begin{cases} 1 & \text{if } \Theta(i,k) \\ 0 & \text{otherwise} \end{cases},$$

  where condition $\Theta(i,k)$ is true if either passengers are allowed to board bus i or any passenger on board bus i reaches his/her destination at event k.
- $h_i(k)$: Holding action of bus i at instant k. Where $h_i(k) = n_i\tau,\ n_i \in Z^+, \tau > 0$.

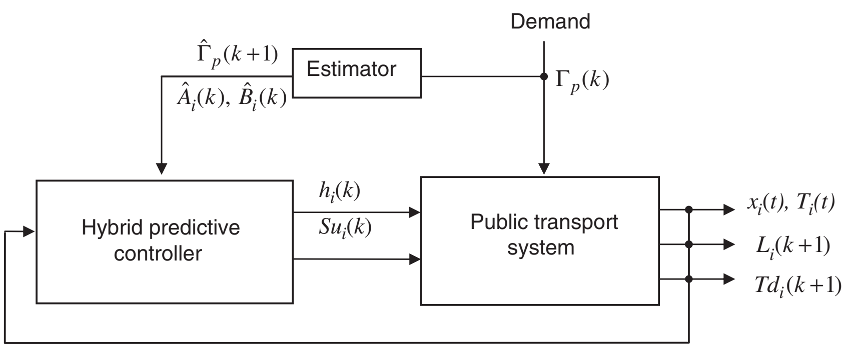

Figure 2. HPC for the public transport system.

These expressions mean that the holding periods are multiples of a fixed step τ. This assumption is applied to simplify both the formulation and the application of the solution algorithm (Section 3). In the numerical example (Section 4), $\tau = 30\,\mathrm{s}$ and $n_i \in \{0, 1, 2, 3\}$. The reason for choosing discrete holding lapses was first, from an operational standpoint, to facilitate the bus drivers to follow the instructions by the central dispatcher. Moreover, having differences of less than 30 s in holding values is not practical, mainly due to constraints given by real driving conditions (unexpected traffic, flexibility to the driver to start operating, communication with the central, etc.).

Next, we analytically define the predictive model, including state space variables and model outputs. In Section 2.3, the operational constraints are described. In Section 2.4, we complete the HPC strategy definition by specifying the proposed dynamic objective function along with the definition of variables specified above.

2.2. *Predictive model*

The predictive model will describe the dynamic behaviour of the aforementioned main variables as a function of the control actions.

First, the expected bus position at instant t, $\hat{x}_i(t)$, is described as a function of the bus's instantaneous speed $v_i(t)$ that depends on the continuous time and the applied control actions. Let us start computing the position of the bus i in continuous time t as follows:

$$\hat{x}_i(t) = x_i(t_k) + \int_{t_k}^{t} \hat{v}_i(\vartheta)\mathrm{d}\vartheta, \tag{1}$$

where t_k is the continuous instant at which the event k is triggered and $x_i(t_k)$ the position of bus i at instant t_k. The instantaneous speed $\hat{v}_i(t)$ is modelled by assuming a constant speed v_0 whenever the vehicle is moving, and the speed is equal to zero otherwise, which implies that the processes of acceleration and deceleration of the buses are ignored. Figure 3 shows the speed function of bus i while it is travelling from the station it reaches at instant k until the bus arrives at the next stop along its route (which is associated with future instant $k+d$). Notice that d corresponds to the time lapses (intervals) triggered by other buses of the fleet arriving at different bus stops, taking place while bus i is travelling between its current stop and the next (including the time it is at its current stop). In the figure, $\hat{Tr}_i(k)$ is the estimated time associated with passenger transference (maximum between the

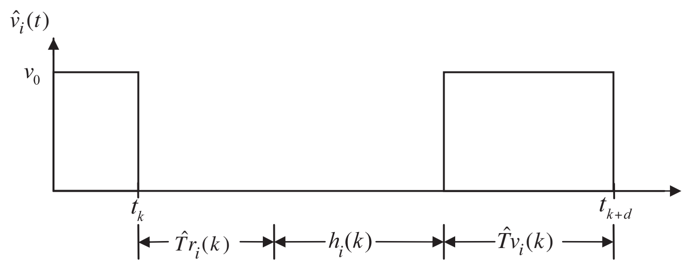

Figure 3. Example of bus speed between consecutive stops.

boarding and alighting times) and $\hat{T}v_i(k)$ is the estimated travel time between two consecutive stations, namely station p and the next station. As defined above, the controller decides the holding time of bus i at event k, denoted $h_i(k)$. Clearly, when a bus is at a bus stop, its velocity equals zero while the bus is transferring passengers and also during the holding period (if the bus is held there), which means that the instant speed actually depend on those variables.

In this context and based on Figure 3, an estimation of the instantaneous speed can be computed as follows:

$$\hat{v}_i(t) = \begin{cases} 0 & t_k \leq t \leq t_k + \hat{T}r_i(k) + h_i(k) \\ v_0 & t_k + \hat{T}r_i(k) + h_i(k) \leq t \leq t_{k+d} \end{cases} \tag{2}$$

In order to trigger the next event of the dynamic model, the expected remaining time (measured from instant t) for the bus i to reach the next stop is required; it can be computed as follows:

$$\hat{T}_i(t) = t_k + Su_i(k) \cdot \left(h_i(k) + \hat{T}r_i(k)\right) + \hat{T}v_i(k) - t, \quad t_k \leq t \leq t_{k+d}. \tag{3}$$

Estimations of the continuous state space variables of our proposed scheme are given by Equations (1) and (3). Next, the prediction of discrete output variables of the dynamic model, required for the HPC strategy ($\hat{L}_i(k+1)$ and $\hat{T}d_i(k+1)$), are defined and analytically computed.

First, let us define the predicted passenger load $\hat{L}_i(k+1)$, as the estimated number of passengers on bus i once it departs from the station. Analytically,

$$\hat{L}_i(k+1) = \begin{cases} \min\left\{\bar{L}, L_i(k) + Su_i(k)\left(\hat{B}_i(k) - \hat{A}_i(k)\right)\right\} & \text{if bus } i \text{ triggered event } k \\ L_i(k) & \text{otherwise} \end{cases}, \tag{4}$$

where $\bar{L}$ is the bus capacity, $L_i(k)$ is the load of bus i at instant k, $\hat{B}_i(k)$ corresponds to the expected number of passengers that will board bus i, constrained by the available capacity of the bus and $\hat{A}_i(k)$ represents the estimated number of passengers alighting from bus i at event k.

Note that $\hat{A}_i(k)$ and $\hat{B}_i(k)$ are obtained through a statistical analysis of data collected from sensors that should be located at stops and buses. In our approach, these estimations are obtained from data of both a set of previous similar days (offline historical data) and dynamic information occurring in the same day (online data).

Based on offline data, we are able to estimate $\hat{A}_i(k)$ using the most frequent destination patterns from previous days over the same period; then, those estimations are corrected with online destination data obtained from observed preferences from passengers already in the system. $\hat{B}_i(k)$ is computed based on both the estimated bus stop load $\Gamma_p(k)$ at instant k and the bus capacity; it is estimated using autoregressive moving average models for the arrival time of passengers at stops. Moreover, the estimated transference time defined before is $\hat{T}r_i(k) = \text{Max}\{t_a \cdot \hat{A}_i(k), t_b \cdot \hat{B}_i(k)\}$ where t_a and t_b are the marginal rate of boarding and alighting, respectively, in seconds per passenger.

In addition, the estimated departure time $\hat{T}d_i(k+1)$ once bus i departs from its current stop can be computed as

$$\hat{T}d_i(k+1) = \begin{cases} t_k + Su_i(k) \cdot \left(h_i(k) + \hat{T}r_i(k)\right) & \text{if bus } i \text{ triggered event } k \\ Td_i(k) & \text{otherwise} \end{cases} \tag{5}$$

The prediction of the bus stop load $\hat{\Gamma}_p(k+1)$ (when bus i departs from stop p), defined as the number of passengers waiting at bus stop (station) p associated with the bus i that triggered event k; it can be computed as follows:

$$\hat{\Gamma}_p(k+1) = \begin{cases} \Gamma_p(k) + \hat{\delta}_p(k) - \hat{B}_i(k) & \text{if bus } i \text{ triggered event } k \\ \Gamma_p(k) + \hat{\delta}_p(k) & \text{otherwise} \end{cases}, \tag{6}$$

where $\Gamma_p(k)$ is the bus stop load at the same stop p at instant k; $\hat{\delta}_p(k)$ provides the number of passengers that arrive at the bus stop between instants k and the instant of the bus departure from this stop; $\hat{\delta}_p(k)$ is generated based on the statistical analysis of the data in both the previous similar days and the same day (both offline and online historical data) and is estimated considering autoregressive moving average models for the arrival time of passengers to stops.

By using the prediction of the departure time as in Equation (5), it is possible to predict the headway $\hat{H}_i(k+1)$ of bus i that triggers the event k, with respect to its precedent bus $i-1$ when it reaches the same stop, which corresponds to event $k+1-z_{i-1}$. Analytically,

$$\hat{H}_i(k+1) = \hat{T}d_i(k+1) - \hat{T}d_{i-1}(k+1-z_{i-1}), \tag{7}$$

where $\hat{T}d_i(k+1)$ is associated with the bus i that triggers the event k, and $\hat{T}d_{i-1}(k+1-z_{i-1})$ represents the predicted departure time of precedent bus $i-1$ that triggers the event $k-z_{i-1}$, at the same stop. The variable z_{i-1} represents the number of events between the arrival of the precedent bus $i-1$ and the bus i, both reaching the same stop.

2.3. *Operational constraints*

The predictive model of the public transport system must satisfy some physical and operational constraints. The first constraint corresponds to the capacity constraint (already stated in Equation (4)). This is a physical constraint as the bus cannot transport more passengers than its maximum capacity. We can also apply a service policy by setting such a capacity differently in order to avoid overcrowding.

Both the precedence constraint and the demand consistency are relevant, because every passenger has a specific origin and destination. Precedence constraints avoid passengers getting off before they get on any bus. With regard to the demand, it is assumed that there are no transfer nodes, and therefore, once a passenger is on board a bus, he (she) will alight from the same bus at his (her) destination stop. Also, once a passenger arrives at their destination, he (she) will always get off the bus there (passengers want to minimise their travel time, so we assume that passengers do not stay on buses in loops).

Regarding bus operation, the model is constrained to stop at a station if there is any passenger requesting to get off, even though the model recommends performing a station-skipping action, similar to what is suggested by Sun and Hickman (2005). Thus, if the next stop is the destination of even one passenger then the skipping action cannot be applied and the bus must stop and the passengers waiting can board. This strategy seems to work better than including that aspect as a penalty in the objective function, in which case some of the passengers could end up getting off at a station different from their planned destination. On the other hand, if the model determines a holding action at a certain stop, which is not physically appropriated for such an operation, then the bus just stops during a lapse required for a normal passenger transfer operation.

As a physical constraint, and also for practical purposes, the control action holding can be applied just at specific stops, properly equipped to perform such an action. On the other hand, station-skipping could be applied at any bus stop. Each bus is identified by a unique internal label. However, the model allows the indices to be updated when a bus arrives at its next stop, sorted in such a way that bus $i-1$ always precedes bus i. One important issue is that overtaking is allowed in the model as the indices associated with buses (i and $i-1$ for two consecutive buses) are set each time an event occurs and a control action is applied; in such cases the indices are properly updated and sorted.

2.4. *Objective function*

The next step is to properly define a predictive objective function in order to make the real-time decisions and optimise the dynamic system. In this case, we will pursue the minimisation of expression (8), which comprises five components, all of them definitely oriented to user cost through total in-vehicle ride and waiting times. Analytically,

$$\min_{\{u(k),u(k+1),\ldots,u(k+Np-1)\}} \sum_{\ell=1}^{Np} \Big[\theta_1 \cdot \hat{H}_i(k+\ell)\hat{\Gamma}_p(k+\ell) + \theta_2 \cdot (\hat{H}_i(k+\ell) - \bar{H})^2 + \theta_3 \cdot \hat{L}_i(k+\ell)h_i(k+\ell-1) + \theta_4 \cdot \hat{L}_i(k+\ell)\hat{T}r_i(k+\ell-1) + \theta_5 \cdot \hat{\Gamma}_p(k+\ell)\hat{H}_{i+1}(k+\ell+z_{i+1})(1 - Su_i(k+\ell-1))\Big]\Big|_{\substack{i=i(k+\ell-1)\\ p=p(k+\ell-1)}}, \tag{8}$$

where $\{u(k),\ldots,u(k+Np-1)\}$ is the control action sequence with $u(k+\ell-1) = \begin{bmatrix} h_i(k+\ell-1) \\ Su_i(k+\ell-1) \end{bmatrix}$ when bus i triggers event $k+\ell-1$; Np is the prediction horizon and b is the number of buses in the fleet.

Note that $i = i(k+\ell-1) \in \{1,\ldots,b\}, p = p(k+\ell-1) \in \{1,\ldots,P\}$, if we consider that the future event $k+\ell-1$ is triggered by one bus $i(k+\ell-1)$ arriving to a specific station downstream $p(k+\ell-1)$. In expression (8), $\theta_j, j = 1,\ldots,5,$ are weighting parameters, and have to be tuned depending on the specific problem to be treated and on the physical interpretation of the different components as well.

$\bar{H}$ corresponds to the desired headway (set-point) designed for servicing the system demand during a certain time period. Normally, the design headway is related to the design frequency that directly depends on the segment loads, and can be determined, for example, as the minimum required for moving the passengers on the most loaded segment along the bus route. In more sophisticated systems, the design frequency is computed by minimising a static objective function involving operator as well as user costs, in which case the optimal frequency is in most cases larger than the minimum frequency able to carry all passengers at an aggregated level.

The first term in Equation (8) quantifies the total passenger waiting time at stops and depends on the predicted headway along with the bus stop load. The second term captures the regularisation of bus headways, to maintain the headway as close as possible to the design headway. The third component measures the delay associated with passengers on-board a vehicle when they are held at a control station due to the application of the holding strategy. The fourth component corresponds to the extra travel time incurred by the passengers on board due to the transference of passenger process. The longer the transference is, the higher this component becomes. This component was included mainly for the evaluation of station-skipping (apart from the fifth terms explained next) as every time the controller decides to skip a stop, there is an extra benefit for all passengers on board since they will save time because the bus is not going to decelerate and stop for a while to board and alight new passengers. Finally, the fifth component is the extra waiting time of passengers whose station is skipped by an expressed vehicle, associated with the station-skipping strategy.

Note that the proposed objective function is oriented to the satisfaction of users through travel and waiting time because we are proposing an operational level scheme. Therefore, assuming a fixed fleet size obtained from the design frequency, which is the inverse of the design headway defined in Equation (8), the only relevant benefit of applying the proposed real-time control strategies, is on passengers' level of service. Given these considerations, operational cost components were not considered in the objective function specification, although under other conditions they could become important in the real-time decisions.

In the next section we describe the solution algorithm proposed and implemented in order to dynamically solve the formulation in Equation (8) using the predictive model described in Section 2.2 and the operational constraints presented in Section 2.3.

3. HPC solution based on genetic algorithms (HPC-GA)

Genetic algorithms are used to solve the optimisation of the objective function, since they can efficiently cope with mixed-integer non-linear problems. Another advantage is that the objective function gradient does not need to be calculated, reducing computational effort. The GA approach in HPC provides a sub-optimal discrete control law close to the optimal one. When the best solution is maintained in the population, it can be shown that the GA converges to the optimal solution (Rudolph 1994). However, due to the limited time between the sampling instances, reaching the global optimum is not guaranteed. Nevertheless, the probabilistic nature of the algorithm ensures that it finds an approximately optimal solution. In contrast with this, the application of traditional optimisation techniques to solve the same problem cannot guarantee even the calculation of a feasible solution, because of the complexity of the optimisation problem and the time required to make the real-time decision. Since in this case we are dealing with a complex mixed-integer and non-linear programming (MINLP), using the GA optimisation is justified.

A potential solution of the GA is called individual. The individual can be represented by a set of parameters related to the genes of a chromosome and can be described in a binary or integer form. The individual represents a possible control action sequence $\{u(k), \ldots, u(k+Np-1)\}$, where each element is a gene, and the individual length corresponds to the prediction horizon Np.

Using genetic evolution, the fittest chromosome is selected to assure the best offspring. The best parental genes are selected, mixed and recombined for the production of offspring in the next generation. For the recombination of genetic populations, two fundamental operators are used: crossover and mutation. For the crossover mechanism, the portions of two chromosomes are exchanged with a certain probability in order to produce the offspring. The mutation operator alters each portion randomly with a certain probability (Man *et al.* 1999).

In this work, there are two manipulated variables: holding action and station-skipping. The holding action takes integer values at the selected bus stops. Station-skipping is defined with '0' value when the bus skips the stop and '1' otherwise. Both manipulated variables are exclusive to a bus stop, as when the station-skipping is applied, the holding action cannot be applied.

Considering these definitions, the following states of the manipulated variables are defined:

$$u(k+\ell-1) = \begin{bmatrix} h_i(k+\ell-1) \\ Su_i(k+\ell-1) \end{bmatrix} \in \{U^1, U^2, \ldots, U^j, \ldots, U^Q\},$$

where U^j corresponds to one of the Q specific control actions.

Considering these definitions and using four integer values for the holding action: 0, 30, 60 and 90 s at the selected bus stops, the following states of the manipulated variables are defined:

$$u(k+\ell-1) \in \left\{ \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 30 \\ 1 \end{bmatrix}, \begin{bmatrix} 60 \\ 1 \end{bmatrix}, \begin{bmatrix} 90 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right\},$$

where the first row represents the holding action and the second one represents station-skipping. In order to apply GA, the following codification is proposed:

$$U^1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad U^2 = \begin{bmatrix} 30 \\ 1 \end{bmatrix}, \quad U^3 = \begin{bmatrix} 60 \\ 1 \end{bmatrix}, \quad U^4 = \begin{bmatrix} 90 \\ 1 \end{bmatrix}, \quad U^5 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Also, as mentioned in Section 2.3, the following constraints for the control actions should be satisfied:

- If the passenger needs to get off, the bus should be stopped, and therefore station-skipping action cannot be applied.
- The holding action is defined for some specified bus stops.

The complete procedure for the GA applied to this control problem corresponds to an efficient adaptation of the GA proposed by Man *et al.* (1999). The major modifications with respect to the original GA are the proposed mutation operator and the way we avoid repeating the computation of future states already computed in previous steps of the GA implementation. The algorithm is as follows.

(1) Initialise a random population of individuals, i.e., create random integer feasible solutions of manipulated variables (control action sequence) for the HPC problem.

For example, let us take a prediction horizon equals to 4 ($Np = 4$); then, there are 5^4 possible individuals (Q^{Np}) not all feasible because of the constraints explained above. In the next example, the size of the population is n individuals per generation.

$$\text{Population } i \Leftrightarrow \begin{pmatrix} \text{Individual 1} \\ \text{Individual 2} \\ \vdots \\ \text{Individual n} \end{pmatrix} \Leftrightarrow \begin{pmatrix} U^1, U^1, U^2, U^5 \\ U^2, U^1, U^3, U^4 \\ \vdots \\ U^4, U^5, U^1, U^3 \end{pmatrix}$$

For example, Individual 1 means that the vector of the future control action is

$$\text{Individual 1} = [u(k)u(k+1)u(k+2)u(k+3)]^T = [U^1, U^1, U^2, U^5]^T = \begin{bmatrix} 0 & 0 & 30 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix}^T.$$

Thus, in this example the future control actions $u(k)$ and $u(k+1)$ indicate no holding at the bus stops for passengers transferring. At instant $(k+2)$, holdings of 30 s is proposed. At instant $(k+3)$, station-skipping is applied.

(2) Evaluate the fitness function for all initial individuals in the population using (8). If the individual is not feasible, penalise it (pro-life strategy). In this step, we suggest to sort the individuals according to their first element corresponding to future control actions in order to evaluate and record the predictive variables for each control sequence. So, if we evaluate the fitness of individual $\left[U^1, U^1, U^2, U^5\right]^T$, the computation of other individuals with the same initial control actions such as $\left[U^1, X, X, X\right]^T$, $\left[U^1, U^1, X, X\right]^T$, $\left[U^1, U^1, U^2, X\right]^T$ will be less expensive computationally as the recursion for the predictions will not be performed again. Moreover, if the same individual $\left[U^1, U^1, U^2, U^5\right]^T$ appears in new generations, its fitness, as it was obtained before, will not be calculated again.

(3) Select random parents from the population (different vectors of the future control actions). For example, Individual 2 and Individual 6 are chosen as the parents.

$$\overbrace{\underbrace{U^2, U^1}_{\text{2A}}, \underbrace{U^3, U^4}_{\text{2B}}}^{\text{Individual 2}} \quad \overbrace{\underbrace{U^4, U^5}_{\text{6A}}, \underbrace{U^2, U^1}_{\text{6B}}}^{\text{Individual 6}}$$

(4) Generate a random number between 0 and 1. If the number is less than the probability p_c, choose a random integer in the range $0 < c_p < N_p$ (c_p denotes the crossover point) and apply the crossover to the selected individuals in order to generate an offspring.

After the crossover step:

$$\overbrace{\underbrace{U^2, U^1}_{\text{2A}}, \underbrace{U^2, U^1}_{\text{6B}}}^{\text{New Individual 1}} \quad \overbrace{\underbrace{U^4, U^5}_{\text{6A}}, \underbrace{U^3, U^4}_{\text{2B}}}^{\text{New Individual 2}}$$

(5) For each gene of all the individuals in the offspring, generate a random number between 0 and 1. If the number is less than the probability p_m, apply the modified mutation operator to the gene. The modified mutation considers that the gene will change to a possible control action belonging to the set $\{U^1, U^2, \ldots, U^j, \ldots, U^Q\}$ with a different probability. So, the probability for mutation from any gene, to the control action U^i equals p_{U^i}, where $\sum_{i=1}^{Q} p_{U^i} = 1$. By doing this, some control actions that are very common will be analysed with a higher probability. For example, the probability for the mutation to a station-skipping $\left(U^5 = \left[\begin{smallmatrix}0\\0\end{smallmatrix}\right]\right)$ or not holding $\left(U^1 = \left[\begin{smallmatrix}0\\1\end{smallmatrix}\right]\right)$ control actions will be larger, as it is allowed doing those control actions in all stops.

After the mutation step in the third gene and fourth gene, respectively,

$$\overbrace{U^2, U^1, \underset{\uparrow}{U^5}, U^1}^{\text{New Individual 1}} \quad \overbrace{U^4, U^5, \underset{\uparrow}{U^3}, U^1}^{\text{New Individual 2}}$$

(6) Evaluate the fitness given by the objective function (8) of all the individuals in the offspring population. If the individual is not feasible, penalise its corresponding fitness. We recommend the same time savings procedure used in step 2.
(7) Select the best individuals according to the objective function (8). To record the best individual obtained so far, as it could be the optimal control action sequence to be applied.
(8) Replace the weakest individuals from the previous generation with the strongest individuals of the new generation selected in step 6.
(9) If the objective function value reaches the defined tolerance or the maximum generation number is reached (stopping criteria), then stop. Otherwise, go to step 2.

Since we proposed a real-time control strategy, the best stopping algorithm criterion corresponds to the number of generations, which is associated with the maximum computational time available to solve this problem.

The genetic algorithm approach in HPC provides a sub-optimal discrete control law close to the optimal one. The tuning parameters of the GA method are the number of individuals (*Nind*), number of generations (*Ngen*), crossover probability (p_c) and mutation probabilities (p_m, p_{U^i}).

4. Simulation results

4.1. *Experiment description*

The proposed strategy is applied over a bus corridor of 8000 m with a fleet of $b = 6$ buses, of capacity for 72 passengers. The system comprises $P = 10$ stations evenly distributed over the bus route (station spacing of 800 m). The holding control action is applied at bus stops 3 and 7, while the skipping actions can be applied at all stations. The simulation assumes uncertain online demand for the arrival of passengers to stations, which follows a Poisson process with demand rates differentiated by station and period (Figure 4). The marginal boarding and alighting rates are $t_a = 3\,\mathrm{s\,pax}^{-1}$ and $t_b = 5\,\mathrm{s\,pax}^{-1}$, respectively, in seconds per passenger. The desired headway (set-point) is $\bar{H} = 6\,\mathrm{min}$. Moreover, we assume that

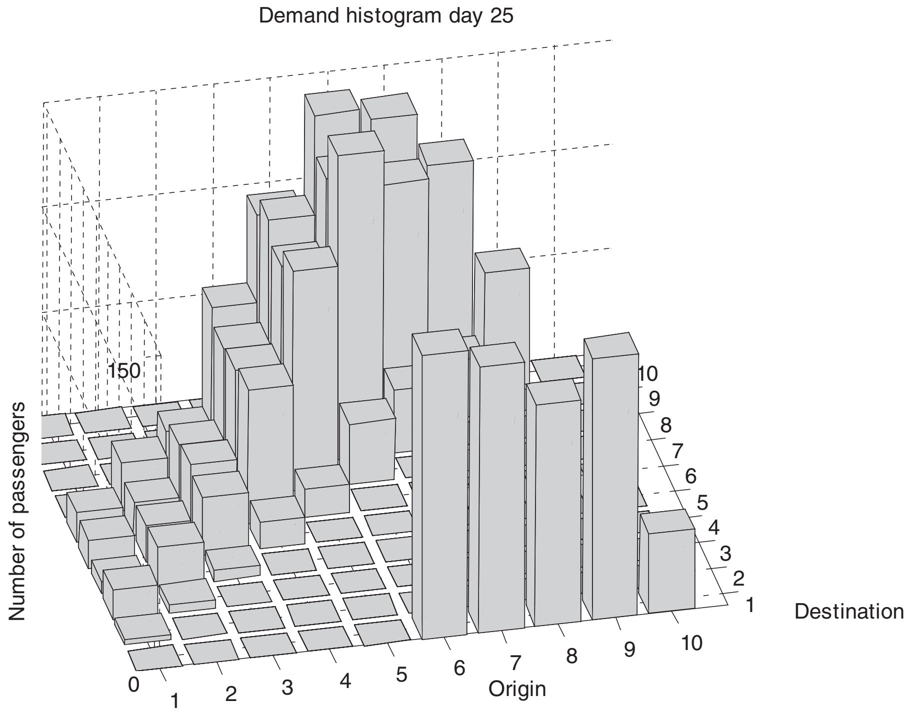

Figure 4. Demand configuration for a specific day (number of passengers per O–D pair).

buses moves at a constant speed $v_0 = 25\,\mathrm{km\,h^{-1}}$ when they are not at a stop. The total simulation period was 2 h, including a warm-up period (discarded for statistics) of 15 min at the beginning and at the end of the simulation. All processes were run on a Computer Pentium Core 2 duo, 2×2.4 GHz with 3 GB RAM.

The demand distribution corresponds to the behaviour of the passengers along a linear corridor, in which the first five stations are evenly distributed over one direction of the route, while the last five stops are evenly distributed over the opposite direction of the route. Thus, Station 2, for example, is in front of the physical location of Station 8. In this example, there are some origin–destination pairs with no demand, as shown in Figure 4. However, the modelling approach described in the previous section can be extended to any demand configuration.

For the proposed genetic algorithm, the chosen parameters are $p_c = 0.8$, $p_m = 0.1$, $p_{U^1} = 0.26$, $p_{U^2} = 0.2$, $p_{U^3} = 0.13$, $p_{U^4} = 0.07$ and $p_{U^5} = 0.34$. The available period set for solving the real-time optimisation problem before the expected occurrence of an event is 30 s. This lapse considers the running time of the algorithm plus a preparation period to give instructions to the driver. Therefore, the number of individuals (*Nind*) and generations (*Ngen*) are set in a fixed value such that the controller is able to solve the optimisation problem in less than 20 s assuming a preparation time for drivers of around 10 s additionally. Note that *Ngen* and *Nind* must be set differently for different prediction horizons to fulfil the computation time constraint. In numbers, for $Np = 2$ $Ngen = 5$ $Nind = 5$, for $Np = 5$ $Ngen = 20$ $Nind = 40$ and for $Np = 10$ $Ngen = 20$ $Nind = 40$.

Next, we propose an analysis of the objective function weighting parameters of expression (8) for being used in the experiments described in Section 4.4.

4.2. *Analysis of the weighting parameters in the objective function*

We analyse the weighting parameters of the objective function (8) for the Hybrid Predictive Controller. The aim of this study is to set the weights that provide not only optimal total travel times (in-vehicle ride times as well as waiting times) but also a minimum standard deviation when different demand patterns are considered on different days. The weighting parameters could reproduce existing values of waiting and in-vehicle times savings for public transport users, which can be estimated using stated or revealed preferences techniques; for example, Australian Transport Council (2006) provides a survey of several studies on valuation of time. This study shows that the users value waiting time savings between 1.17 and 2.88 times as much as in-vehicle time savings, depending on several factors such as perceived waiting conditions, length of the waiting time, bus arrival reliability, etc. Nevertheless, for illustrative purposes, in this simulation we decide to evaluate all combinations of weights θ_i of magnitude 1, 0.01, 0.0001 and 0 (81 possible combinations) for 25 days of data, as a useful way to analyse the performance of the different components of the objective function (obtaining significant variation in the mean performance values – waiting time plus in-vehicle travel time – for different combinations of weighting parameters), instead of attempting to reproduce reported user's perceptions of time costs.

Next, the criterion for choosing the weights is to minimise the following expressions:

$$E_i = \bar{x}_i + \frac{2\sigma_{x_i}}{\sqrt{n}}, \quad i = 1, \ldots, 1024 \tag{9}$$

$$\sigma_{x_i} = \sqrt{\frac{\sum_{j=1}^{n} (x_{ij} - \bar{x}_i)^2}{n}}, \quad i = 1, \ldots, 1024 \tag{10}$$

where x_{ij} is the mean time (waiting and in-vehicle ride times) for the weights' combination i during day j, with $n = 25$ days. $\bar{x}_i$ is the mean value of x_{ij} for $j = 1, \ldots, 25$ days.

In Table 1, the results for the best combinations in terms of E_i and σ_{x_i} are reported for two prediction horizons: $Np = 2$ and $Np = 5$.

All cases presented in Table 1 provide a reasonable waiting time and standard deviation. Using those parameters in the HPC, the level of service remains almost constant. In cases like those, a more accurate prediction of the total time required for going from one stop to another could be provided to customers in advance.

In the next section we present a heuristic based on an expert control algorithm, which was designed to keep the bus headways as regular as possible. The goal of this procedure is to provide a benchmark for the HPC algorithm performance.

4.3. *Expert control algorithm*

The aim of this expert control strategy is to regularise the headway between the arrivals of consecutive buses to stops and then to avoid bus bunching. In order to achieve this

Table 1. Average waiting time and in-vehicle ride time per passenger.

					Prediction horizon $Np=2$				Prediction horizon $Np=5$			
					E_i		$100 \cdot \sigma_{x_i}/E_i$		E_i		$100 \cdot \sigma_{x_i}/E_i$	
θ_1	θ_2	θ_3	θ_4	θ_5	Waiting time (min)	In-vehicle ride time (min)	Waiting time (min)	In-vehicle ride time (min)	Waiting time (min)	In-vehicle ride time (min)	Waiting time (min)	In-vehicle ride time (min)
1	1	1	0	1	6.34	9.74	12.46	3.04	6.61	9.70	11.11	4.05
1	1	0.0001	0	1	6.59	9.87	9.04	4.36	6.60	9.81	12.50	4.23
1	1	0.01	0	0.0001	6.53	9.71	11.02	4.05	6.42	9.71	10.55	3.25
1	1	1	1	1	6.40	9.92	13.01	3.01	6.69	9.84	12.61	3.11
1	1	0.01	0.01	1	6.35	9.76	12.50	3.07	6.68	9.71	11.15	4.07
0.01	0.01	1	1	0.01	6.45	9.90	12.98	3.03	6.70	9.90	13.01	3.14

Note: $Np=2$ and $Np=5$.

objective, the strategies aim at keeping each group of three consecutive buses equidistant. In Figure 5, we depict the relative position of three consecutive buses $i-1$ (precedent bus), i (current bus), $i+1$ (next bus). Let us define $x_{i-1}(k)$ as the position of the precedent bus $i-1$, $x_i\,(k)$ the position of the current bus i and $x_{i+1}(k)$ the position of the next bus $i+1$, measured at event k when bus i arrives at a stop. Then, the distance $d_i(k)$ of bus i with respect to the centre of the distance between the precedent and the next bus is given by

$$d_i(k) = x_i(k) - \left(\frac{x_{i-1}(k) - x_{i+1}(k)}{2}\right). \tag{11}$$

Figure 6 shows a generic closed loop diagram for a control strategy, in which the control actions are triggered when a bus i reaches a stop (event k). The same control strategies associated with event k proposed for the HPC, which are holding $h_i(k)$ and stop-skipping $Su_i(k)$, are considered for this static control heuristic. As seen in Figure 6, one advantage of this method is its simplicity, since it does not require prediction of the demand (myopic strategy). In this application, we chose the discrete values for the holding lapse, namely $h_i(k)$ equals to 0 s, $\beta = 30$ s, $2\beta = 60$ s and $3\beta = 90$ s, like in the HPC strategy. Assuming at least six people on an average getting on buses at each bus stop, and considering 5 s as the marginal boarding time dominating alighting, we finally use a reasonable value of $\beta = 30$ s between the discrete values for the holding. On the other hand, stop-skipping is defined as $Su_i(k) = 0$ when the bus skips the stop and $Su_i(k) = 1$ otherwise. Both manipulated variables are excluding at every bus stop; then, when the station-skipping is decided, the holding action cannot be applied, and vice versa.

In simple terms, the expert control strategy consists of moving forward the bus i, if it is late with respect to the central position of the trajectory between the precedent bus $i-1$ and the next bus $i+1$; otherwise, the bus i is delayed.

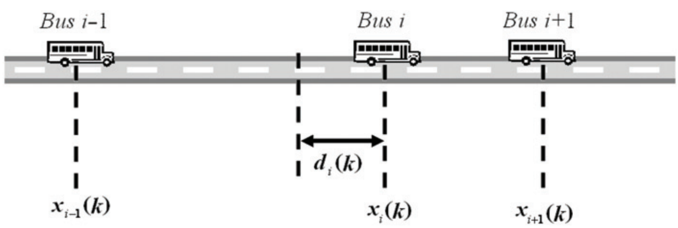

Figure 5. Relative positions of three consecutive buses.

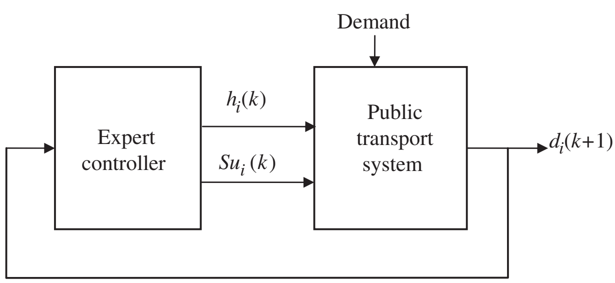

Figure 6. Expert control for the public transport system.

Next, we define the expert controller as a set of heuristic rules. We assume that buses move at an average speed of $v_0 = 25\,\mathrm{km\,h^{-1}}$, that is, $6.94\,\mathrm{m\,s^{-1}}$. Therefore, if we design $2g$ as the distance that a bus refrains from travelling due to a holding control action equivalent to β, then we can show that $2g = v_0 \cdot \beta = 208.2\,\mathrm{m}$. As a consequence, if the bus is held a lapse of 2β – it will refrain from travelling a distance of $4g$. Similarly, if the bus is held a lapse of 3β – it will refrain from travelling $6g$. Moreover, 208.2 m is approximately the same distance that the bus would travel additionally as the effect of a station-skipping control action, assuming an average lapse for transference of passengers around 30 s.

Therefore, if the holding control action takes a value β, we can define a neighbourhood radius g around $d_i(k) = 2g$ (namely $g < d_i(k) < 3g$), where this control action will be applied.

Following the same reasoning, within the range $3g < d_i(k) < 5g$, the holding control action will take a value 2β ($h_i(k) = 2\beta$) and for $5g < d_i(k) < 7g$ the holding control action will take a value $3\beta (h_i(k) = 3\beta)$. Instead, if $-g < d_i(k) < g$, the holding and station-skipping control actions are not necessary ($Su_i(k) = 0, h_i(k) = 0$). Finally, if $d_i(k) < -g$, the recommended control action will be station-skipping only ($h_i(k) = 0, Su_i(k) = 1$).

Thus, adding the limit cases (equalities), we can formulate the expert control strategy (holding and station-skipping based on rules) as the following five heuristic rules:

If $d_i(k) \le -g$ then $h_i(k) = 0, Su_i(k) = 0$
If $-g < d_i(k) \le g$ then $h_i(k) = 0, Su_i(k) = 1$
If $g < d_i(k) \le 3g$ then $h_i(k) = \beta, Su_i(k) = 1$
If $3g < d_i(k) \le 5g$ then $h_i(k) = 2\beta, Su_i(k) = 1$
If $5g < d_i(k)$ then $h_i(k) = 3\beta, Su_i(k) = 1$

If station-skipping is not possible due to operational constraints (namely, a passenger wants to get off there), then $h_i(k) = 0$ and $Su_i(k) = 1$ regardless of the Expert Controller recommendation.

4.4. *Illustrative results*

Below, we report the simulations of the public transport operation for two randomly chosen days (namely, days 15 and 18) to illustrate the behaviour of the system controlled by HPC-GA for a time horizon $Np = 2$ in comparison with two operational schemes: (1) an open-loop (OL) system, which does not consider any type of real-time control, and (2) a Simple Expert Controller as described above, without considering demand prediction features in the control decisions.

Tables 2 and 3 report the average waiting time, the in-vehicle ride time and total travel time per passenger for different weighting factors of the objective function, as in Equation (8). These experiments were run by considering a two-step ahead prediction (Cases 3–8). In the same tables, the open-loop response (Case 1) and the Expert System (Case 2) response are also reported. The open-loop control strategy implies no feedback from both, the output variables and the disturbances; in this case, the holding and skipping control actions are not applied whatsoever. Also, Tables 2 and 3 show the percentage of passengers affected by the holding strategy (%Ph) as well as by station-skipping (%PSu). Besides, in the last column we report Av(h), accounting for the average time that

Table 2. Comparison of HPC-GA, open-loop and expert system for day 15.

Case	Control strategy	Weight actors θ_1–θ_2–θ_3–θ_4–θ_5	Waiting time (min)	In-vehicle ride time (min)	Total time (min)	%Ph	%Psu	Av(h) (min)
1	Open loop	–	10.54	9.61	20.16	–	–	–
2	Expert system	–	7.98	9.85	17.83	23	16	0.87
3	HPC-GA	1–1–1–0–1	7.33	9.91	17.24	4	2	1.23
4	HPC-GA	1–1–0.0001–0–1	7.28	10.01	17.29	7	5	1.70
5	HPC-GA	1–1–0.01–0–0.0001	7.61	9.71	17.32	1	7	1.10
6	HPC-GA	1–1–1–1–1	7.35	9.88	17.23	3	5	1.11
7	HPC-GA	1–1–0.01–0.01–1	7.34	9.95	17.29	7	5	1.25
8	HPC-GA	0.01–0.01–1–1–0.01	7.01	9.98	16.70	5	7	1.54

Note: $Np = 2$.

Table 3. Comparison of HPC-GA, open-loop and expert system for day 18.

Case	Control strategy	Weight factors θ_1–θ_2–θ_3–θ_4–θ_5	Waiting time (min)	In-vehicle ride time (min)	Total time (min)	%Ph	%Psu	Av(h) (min)
1	Open loop	–	12.23	9.40	21.64	–	–	–
2	Expert system	–	7.34	9.80	17.14	29	23	0.85
3	HPC-GA	1–1–1–0–1	6.75	9.96	16.71	2	4	1.07
4	HPC-GA	1–1–0.0001–0–1	6.01	10.5	16.51	20	3	1.68
5	HPC-GA	1–1–0.01–0–0.0001	6.56	9.97	16.53	2	8	1.34
6	HPC-GA	1–1–1–1–1	6.85	9.99	16.84	7	5	1.41
7	HPC-GA	1–1–0.01–0.01–1	6.78	9.99	16.77	5	7	2.34
8	HPC-GA	0.01–0.01–1–1–0.01	6.98	9.89	16.87	6	3	1.17

Note: $Np = 2$.

passengers are held on buses (in minutes per passenger) considering only those passengers affected by the holding strategy somewhere in their journey.

We observe a 20% and 10% savings in total travel time for users when using the HPC-GA strategy in comparison with the open-loop system and the proposed Expert controller, respectively. The most significant benefits are associated with a reduction in waiting time for the HPC-GA case (around 38%) while keeping in-vehicle ride times almost constant. These results validate the predictive capabilities of the proposed HPC strategy.

In Tables 2 and 3, note that when in the objective function, the component that measures the additional in-vehicle time due to holding becomes relevant (Case 3, $\theta_3 = 1$), then the HPC-GA strategy generate almost no holding control action (%Ph = 4 and 2 for days 15 and 18, respectively). However, as this weighting factor begins to reduce (Case 4), the HPC strategy proposes more holding actions (for Case 4 %Ph = 7 and 20 for days 15 and 18, respectively). As a consequence the average values of holding per passenger (represented in Av(h)) start increasing. Such results are reasonable, since the HPC-GA strategy begins to benefit those passengers waiting at stations (through the regularisation

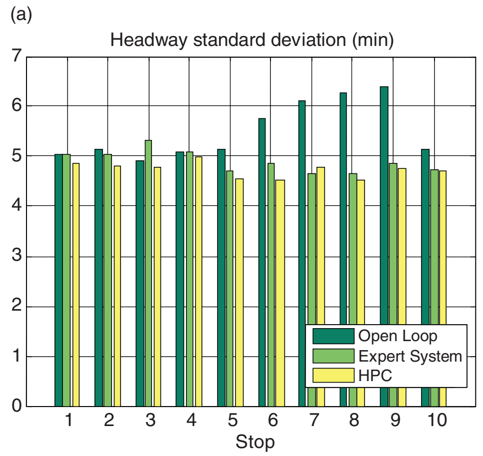

Figure 7. HPC-GA case 3 (weights 1–1–1–0–1): (a) headway standard deviation, day 15 and (b) headway standard deviation, day 18.

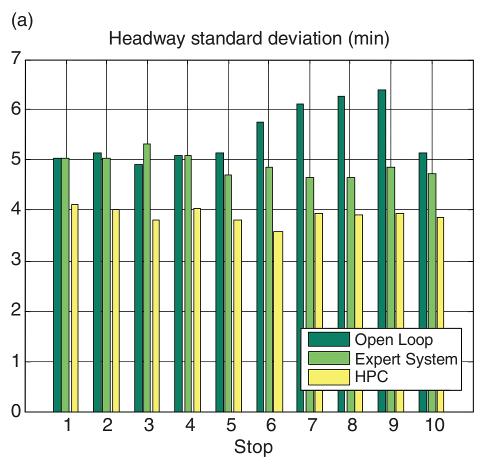

Figure 8. HPC-GA case 4 (weights 1–1–0.0001–0–1): (a) headway standard deviation, day 15 and (b) headway standard deviation, day 18.

of the headways) at the expense of those passengers stopped because of the application of holding. Notice also that as the weight factor θ_5 increases, the number of passengers affected by station-skipping (%PSu) decreases, which consequently produces a slight reduction in waiting time.

To get a better idea of what is happening at the station level, in Figures 7 and 8 we show the headway responses (measured through the standard deviation) for all bus stops, in cases where the system is operated without applying any control strategy (open-loop), by an Expert System (without prediction), and when the HPC-GA strategy is applied ($Np = 2$).

In Figures 7 and 8, we note that although the Expert System strategy shows a reasonable performance, mainly in terms of waiting time, it is not as good as HPC-GA in terms of the stability of headways at bus stations. From Figures 7 and 8, we also observe that HPC-GA provides the best performance in terms of minimising the standard deviation at practically all bus stops. The open-loop case results in the largest standard deviations, which is reasonable since no objective function is minimised. Note that in the open-loop case as well as the Expert System approach, the probability of having some passengers experiencing very long waiting times while others experience very short ones is larger than in the HPC-GA scheme. Therefore, at least from these experiments, HPC-GA improves the system performance in terms of operation and the image of the bus system the passengers have, because of the regularisation of the headways. This also has some practical advantages for the implementation of a scheduled system in which the operator could promise some headways to users (bus departure times from stops) with a high level of certainty.

In Tables 4 and 5, we show the HPC-GA results for three prediction horizons ($Np = 2$, 5 and 10), for Case 3 ($\theta_1 = \theta_2 = \theta_3 = \theta_5 = 1, \theta_4 = 0$).

From such tables, we note differences in performance based on changing the prediction horizon, and therefore, in most cases $Np = 2$, appears to be a good prediction horizon for this system configuration with its specific features in terms of supply and demand. Overall, for larger than $Np = 2$ time-horizons ($Np = 5$ and $Np = 10$) the resulting waiting times become larger. This phenomenon can be explained by the deterioration of the prediction capabilities as the time horizon gets longer due to the high uncertainty associated with future demand.

In order to verify the quality of the proposed GA algorithm for the HPC scheme (explained in detail in Section 3) in terms of both computation effort and accuracy of the solutions, selected tests were conducted applying explicit enumeration of all feasible solutions (HPC-EE). To measure the performance of HPC-GA, the following indices

Table 4. HPC-GA performance for day 15.

Prediction horizon (Np)	Waiting time (min)	In-vehicle ride time (min)	Total time (min)	%Ph	%PSu	Av(h) (min)
2	6.93	9.61	16.54	0	2	1.16
5	6.97	9.91	16.88	0	3	1.21
10	7.00	10.10	17.10	0	3	1.19

Table 5. HPC-GA performance for day 18.

Prediction horizon (Np)	Waiting time (min)	In-vehicle ride time (min)	Total time (min)	%Ph	%PSu	Av(h) (min)
2	5.83	9.78	15.61	1	2	1.03
5	6.22	10.22	16.44	1	3	1.10
10	6.04	10.00	16.04	0	2	1.12

are defined:

$$\mathrm{PCT} = \left[1 - \frac{\text{Computation time (HPC-GA)}}{\text{Computation time (HPC-EE)}}\right] \cdot 100\%,$$

$$\mathrm{PWT} = \left[\frac{\text{Waiting time (HPC-GA)} - \text{Waiting time (HPC-EE)}}{\text{Waiting time (HPC-EE)}}\right] \cdot 100\%,$$

$$\mathrm{PTT} = \frac{[\text{Total time (HPC-GA)} - \text{Total time (HPC-EE)}]}{\text{Total time (HPC-EE)}} \cdot 100\%.$$

The three indices are defined as a comparison between the HPC-GA and HPC-EE algorithms for the same time horizon, to provide a consistent comparison of algorithms' performance. PCT shows a measure of savings (in percentage) associated with computation time between GA and EE. PWT and PTT represent measures of the accuracy of GA when compared with EE (in percentage) for waiting and total travel time, respectively. A summary of the conducted experiments in terms of these indices is shown in Table 6.

GA shows considerable savings in computational effort (by means of PCT) when compared with EE. These savings get larger as the prediction horizon increases, providing high-quality results (by means of PWT and PTT), with errors smaller than 3% in all cases. On the other hand, the Expert System used as benchmark reports very small computation time, but a significantly worse quality of the solution in the order of magnitude. These results are promising and open the door for further improvements in the GA implementation to tackle real-size systems with more complex configurations and implemented for longer time horizons. The computation time of GA for solving the optimisation problem with different prediction horizon ($Np = 2$, 5 and 10) is considerably smaller than the explicit enumeration, mainly when the prediction horizon is long as explicit enumeration explodes with Np. Under these conditions, explicit enumeration can be applied only for short prediction horizons as it takes 53 and 1197 s for $Np = 5$ and 10, respectively.

Note that in case of GA all the proposed strategies can be applied in a real-time setting as computation times are all less than the threshold of 20 s explained before. Moreover, the problem for $Np = 10$ implies a much larger solutions-search space than that of the problem for $Np = 5$. Given that the computation times reported in Table 6 are quite similar (to satisfy the constraint of 20 s maximum), then the quality of the final solution obtained for GA $Np = 10$ is worse than that obtained in the case $Np = 5$.

Table 6. Performance comparison – HPC based on EE, GA and expert system.

Control strategy	Computation total time (s)	Computation per event time (s)	PCT (%)	PWT (%)	PTT (%)
Expert system	0.97	0.0039	–	14	8
HPC-EE $Np = 2$	2500	9.9601	–	–	–
HPC-EE $Np = 5$	13,200	52.5896	–	–	–
HPC-EE $Np = 10$	300,330	1196.5338	–	–	–
HPC-GA $Np = 2$	1750	6.9721	30	1.8	0.5
HPC-GA $Np = 5$	3565	14.2031	73	1.4	0.3
HPC-GA $Np = 10$	4450	17.7290	98.5	2.7	0.4

5. Conclusions

In this article, we have shown a HPC model to optimise in real-time the performance of a public transport system along a linear corridor with uncertain demand at bus stops. The optimisation is conducted by applying, holding and expressing (station-skipping). The proposed HPC strategy was formulated under a discrete event simulation environment and solved by GA tools to efficiently make optimal real-time decisions based on the proposed framework, in terms of both accuracy and computation time. The proposed strategy is compared with a benchmark algorithm (Expert System Control), which does not consider prediction in the decision-making process.

Several objective function options were tested, obtaining very intuitive and reasonable results in all cases, when compared to the benchmark Expert System, and both greatly outperformed the case without any control of real-time decisions. These results support the structure and design conditions of the HPC controller. For example, when the holding penalisation becomes high, the controller avoids applying holding and prefers to implement expressing instead to optimise the dynamic objective function. This flexibility in the formulation allows the controller to accommodate his (her) actions to different service policies, depending on the case. However, from the different results and tests conducted, we recommend developing detailed sensitivity analyses with respect to both prediction horizon and weight parameters in order to obtain optimal policy strategies.

For future research, we plan to work on more complex system configurations, such as trunk schemes combined with feeder transit lines connected with transfer points. Moreover, we plan to test a modified version of the station-skipping action in our model by relaxing the constraint that does not allow a bus to skip a stop if anybody on board requests to get off. This indeed will force us to change the objective function to be consistent with the extra penalty due to either transferring to another bus or walking to the final destination.

As part of ongoing research, we are studying other type of strategies, such as real-time injection of buses where the extra operational cost becomes relevant due to the extra fleet acquisition and operation, and in that case the objective function could require other terms.

In addition, we are working on fine-tuning the weight parameters, under a dynamic multi-objective optimisation scheme also using GA. Finally, we will also test our schemes under a microscopic simulation environment in order to properly capture the dynamic effects of such a transit system.

Acknowledgements

Dr Sáez, Mr Milla, Dr Núñez and Ms Riquelme thank the financial support of the ACT-32 Project 'Real-Time Intelligent Control for Integrated Transit Systems'. Dr Cortés thanks the financial support of ACT-32 Project, Fondecyt Chile Grant: 1061261, and the Millennium Institute Complex Engineering Systems (ICM: P-05-004-F, CONICYT: FBO16).

References

Adamski, A., 1996. Flexible dispatching control tools in public transport. *In*: L. Bianco and P. Toth, eds. *Advanced methods in transportation analysis*. Heidelberg, Germany: Springer, 481–506.

Adamski, A. and Turnau, A., 1998. Simulation support tool for real-time dispatching control in public transport. *Transportation Research Part A: General*, 32 (2), 73–87.

ATC – Australian Transport Council, 2006. *National guidelines for transport system management in Australia*, Volume 4, Urban Transport. Available from: http://www.atcouncil.gov.au/documents/NGTSM.aspx [Retrieved on January 2009].

Barnett, A., 1974. On controlling randomness in transit operations. *Transportation Science*, 8, 102–116.

Bemporad, A. and Morari, M., 1999. Control of systems integrating logic, dynamics and constraints. *Automatica*, 35, 407–427.

Cortés, C.E., Núñez, A., and Sáez, D., 2008. Hybrid adaptive predictive control for a dynamic pickup and delivery problem including traffic congestion. *International Journal of Adaptive Control and Signal Processing*, 22 (2), 103–123.

Eberlein, X.J., (1995). *Real-time control strategies in transit operations: models and analysis*. Thesis (PhD). Massachusetts Institute of Technology, Cambridge, MA.

Eberlein, X.J., Wilson, N.H.M., and Bernstein, D., 1999. Modeling real-time control strategies in public transit operations. *Lecture Notes in Economics and Mathematical Systems*, 471, 325–346.

Eberlein, X.J., Wilson, N.H.M., and Bernstein, D., 2001. The holding problem with real-time information available. *Transportation Science*, 35 (1), 1–18.

Fu, L., Liu, Q., and Calamai, P., 2003. Real-time optimization model for dynamic scheduling of transit operations. *Transportation Research Record*, 1857, 48–55.

Furth, P., 1986. Zonal route design for transit corridors. *Transportation Science*, 20 (1), 1–12.

Furth, P.G. and Day, F.B., 1984. Transit routing and scheduling strategies for heavy demand corridors. *Transportation Research Record*, 1011, 23–26.

Hegyi, A., 2004. *Model predictive control for integrating traffic control measures*. Thesis (PhD). Delft University of Technology, Delft Center for Systems and Control.

Hegyi, A., De Schutter, B., and Hellendoorn, H., 2005. Model predictive control for optimal coordination of ramp metering and variable speed limits. *Transportation Research Part C: Emerging Technologies*, 13, 185–209.

Hickman, M., 2001. An analytic stochastic model for the transit vehicle holding problem. *Transportation Science*, 35 (3), 215–237.

Jordan, W.C. and Turnquist, M.A., 1979. Zone scheduling of bus routes to improve service reliability. *Transportation Science*, 13(3), 242–268.

Karer, G., *et al.*, 2007a. Hybrid fuzzy modeling for model predictive control. *Journal of Intelligent and Robotic Systems*, 50, 297–319.

Karer, G., *et al.*, 2007b. Hybrid fuzzy model-based predictive control of temperature in a batch reactor. *Computers and Chemical Engineering*, 31, 1552–1564.

Lin, G., *et al.*, 1995. *Adaptive control of transit operations*. Report No. MD-26-7002, US Department of Transportation, Federal Transit Administration, November 15.

Man, K., Tang, K., and Kwong, S., 1999. *Genetic algorithms, concepts and designs*. Heidelberg: Springer.

Osuna, E.E. and Newell, G.F., 1972. Control strategies for an idealized public transportation system. *Transportation Science*, 6, 52–72.

Rudolph, G., 1994. Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks*, 5 (1), 96–101.

Sáez, D., Cortés, C.E., and Núñez, A., 2008. Hybrid adaptive predictive control for the multi-vehicle dynamic pick-up and delivery problem based on genetic algorithms and fuzzy clustering. *Computers and Operations Research*, 35 (11), 3412–3438.

Sun, A. and Hickman M., 2004. The *holding problem at multiple holding stations* [online]. *In: 9th International conference on computer-aided scheduling of public transport*, San Diego, CA: EE.UU. Available from: http://fugazi.engr.arizona.edu/caspt/sun.pdf

Sun, A. and Hickman, M., 2005. The real-time stop-skipping problem. *Journal of Intelligent Transportation Systems*, 9 (2), 91–109.

Turnquist, M.A. and Blume, S.W., 1980. Evaluating potential effectiveness of headway control strategies for transit systems. *Transportation Research Record*, 746, 25–29.

Welding, P.I., 1957. The instability of a close-interval service. *Operational Research Quarterly*, 8 (3), 133–148.

Yu, B. and Yang, Z., 2007. A dynamic holding strategy in public transit systems with real-time information. *Applied Intelligence*, 31 (1), 69–80.

Zolfaghari, S., Azizi, N., and Jaber, M., 2004. A model for holding strategy in public transit systems with real-time information. *International Journal of Transport Management*, 2, 99–110.