An Incremental Probabilistic Model to Predict Bus Bunching in Real-Time

Luis Moreira-Matias[1,2], João Gama[2,4],
João Mendes-Moreira[2,3], and Jorge Freire de Sousa[5,6]

[1] Instituto de Telecomunicações, 4200-465 Porto, Portugal
[2] LIAAD-INESC TEC, 4200-465 Porto, Portugal
[3] DEI-FEUP, U. Porto, 4200-465 Porto, Portugal
[4] Faculdade de Economia, U. Porto 4200-465 Porto, Portugal
[5] UGEI-INESC TEC, U. Porto, 4200-465 Porto, Portugal
[6] DEGI-FEUP, U. Porto, 4200-465 Porto, Portugal
{luis.m.matias,jgama,joao.mendes.moreira}@inescporto.pt,
jfsousa@fe.up.pt

Abstract. In this paper, we presented a probabilistic framework to predict Bus Bunching (BB) occurrences in real-time. It uses both historical and real-time data to approximate the headway distributions on the further stops of a given route by employing both offline and online supervised learning techniques. Such approximations are incrementally calculated by reusing the latest prediction residuals to update the further ones. These update rules extend the Perceptron's delta rule by assuming an adaptive beta value based on the current context. These distributions are then used to compute the likelihood of forming a bus platoon on a further stop - which may trigger an threshold-based BB alarm. This framework was evaluated using real-world data about the trips of 3 bus lines throughout an year running on the city of Porto, Portugal. The results are promising.

Keywords: supervised learning, probabilistic reasoning, online learning, perceptron, regression, bus bunching, travel time prediction, headway prediction.

1 Introduction

The bus has become a key player in highly populated urban areas. Inner-city transportation networks are becoming larger and consequently, harder to monitor. The large-scale introduction of GPS-based systems in the bus fleets opened new horizons to be explored by mass transit companies around the globe. This technology made it possible to create highly sophisticated control centers to monitor all the vehicles in real-time. However, this type of control often requires a large number of human resources, who make decisions on the best strategies for each case/trip. Such manpower requirements represent an important slice of the operational costs.

H. Blockeel et al. (Eds.): IDA 2014, LNCS 8819, pp. 227–238, 2014.

It is known that there is some schedule instability, especially in highly frequent routes (10 minutes or less) [1–5]. In this kind of routes it is more important the headway (time separation between vehicle arrivals or departures) regularity than the fulfilment of the arrival time at the bus stops. In fact, a small delay of a bus provokes the raising of the number of passengers in the next stop. This number increases the dwell time (time period where the bus is stopped at a bus stop) and obviously, it also increases the bus's delay. On the other hand, the next bus will have fewer passengers, shorter dwell times without delays. This will continue as a snow ball effect and, at a further point of that route, the two buses will meet at a bus stop, forming a platoon as it is illustrated in Fig. 1. This phenomenon is denominated as **Bus Bunching(BB)** [3,6].

The emergence of these events is completely stochastic as you *never know* when or where they may occur. However, there are some behavioural patterns that may anticipate its occurrence such as *consecutive* headway reductions and travel times longer than expected. Such patterns uncover some regularities on the causes that may be explored by Machine Learning algorithms to provide decision support. It can be done by mining not only the historical location-based data on the daily trips but also on their real-time tracking. Consequently, the problem complexity turns the off-the-shelf learning methods as inadequate to predict BB events in real-time.

By predicting these events, we can not only automatically forecast where a BB occurrence may emerge but also which is the problematic trip/vehicle and **how** can we prevent it from happening. In this work, we introduce a complex framework to predict BB occurrences in a short-term time horizon. This event detection is build over a stepwise methodology which starts by performing an 1) offline regression to predict the Link Travel Times (the travel time between two consecutive stops) which is **incrementally updated** by considering the

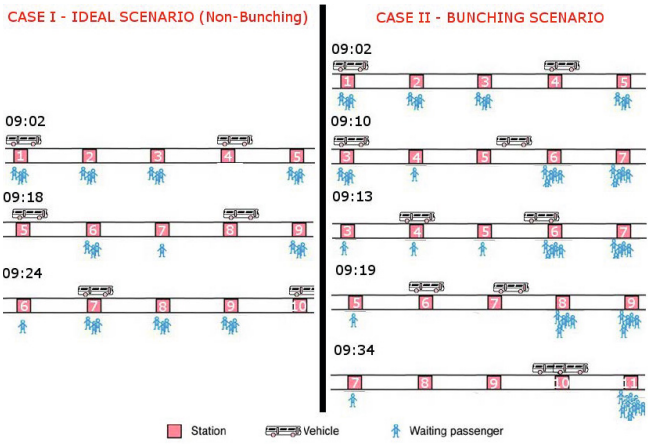

Fig. 1. Bus Bunching illustration

2) error measured from trip to trip and from 3) stop to stop as seeds for a Perceptron-based update rule. Then, a 4) **probabilistic** framework is devised to express the **likelihood** of a pair of buses to form a platoon on a given stop. Finally, 5) these probabilities are used to compute a *Bunching score* which, given a certain context-based threshold, triggers an alarm on a BB occurrence. Our main contributions are threefold:

1. we introduce a novel data driven approach to predict the emergence of BB events in a short-term horizon. More than maintaining the headway stable on the network in exchange of some schedule unreliability, it aims to anticipate last-resource contexts where a corrective action must be took;
2. by producing numerical scores rather than `BUNCHING`/`NO_BUNCHING` labels, we favour the framework's interpretability and, consequently, its ability to adapt to different scenarios;
3. we validated such framework using a large-scale dataset containing timestamped trip records of three distinct bus routes running on the city of Porto, Portugal, during an one-year period.

2 Problem Overview

The Public Transportation (PT) companies operate on high competitive scenarios where there are many options to perform this short connections such as other bus companies, trains, light trams or even private transportation means. The service reliability is key to maintain their profitability. By guaranteeing on-time arrivals, the passengers' perception of the service quality will rise and, consequently, they will pick it often. On the other hand, an unreliable schedule may decrease the number of customers running on that company and therefore, lead to important profit losses [7, 8]. One of the most visible characteristics of an unreliable service is the existence of BB events. Two (or more) buses running together on the same route is an undeniable sign that something is going terribly wrong with the company's service.

To avoid such occurrences, the PT companies installed advanced Control centers where experienced operators are able to monitor the network operations in real-time. Their goal is to suggest **corrective actions** to the bus drivers able to prevent such occurrences. There are four typical methods employed as real-time control strategies [6, 9]:

1. **Bus Holding**: It consists of forcing the driver to increase/reduce the dwell time[1] on a given bus stop along the route;
2. **Speed Modification**: This strategy forces the driver to set a maximum cruise speed on its course (lower than usual on that specific route);
3. **Stop-Skipping**: Skip one or more route stops; also known as *short-cutting* when it requires a path change to reduce the original length of the route.

[1] The time spent by a bus stopped on a given stop.

4. **Short-Turning**: This complex strategy consists of causing a vehicle to skip the remaining route stops (usually at its terminus) to fill a large service gap in another route (usually, the same route but in the opposite direction). In a worst case scenario, the passengers may be subjected to a transfer.

By studying the BB phenomenon, we expect not only to anticipate **when** it may occurs but also which is the most adequate corrective action to employ in each situation. However, these actions must be took as a last resource as they also affect negatively the schedule reliability (even if they do it in a smaller scale). The idea is to be able to automatically perform the following decisions: 1) *when does it worth to take an action?* 2) *which is the action to employ?* 3) *which is the bus/pair of buses to be affected by such action?* Such framework will represent considerable savings to any PT company by reducing the manpower needs on the control department.

The most important variable regarding the BB events is the *distance* (in time) between two consecutive buses running on the same route. Such distance is denominated as **headway**. Let the trip k of a given bus route be defined by $T_k = \{T_{k,1}, T_{k,2}, ..., T_{k,s}\}$ where $T_{k,j}$ stands for the arrival time of the bus running the trip k to the bus stop j and s denotes the number of bus stops defined for such trip. Consequently, the headways between two buses running on consecutive trips $k, k+1$ be defined as follows

$$H = \{h_1, h_2, ..., h_s\} : h_i = T_{k+1,i} - T_{k,i} \tag{1}$$

Theoretically, the headway between two consecutive trips should be *constant*. However, due to the stochastic events (e.g. traffic jams, unexpected high demand on a given stop, etc.) arose during a bus trip, the headway suffers some variability. Such variability can provoke other events that may decrease the existing headway following a *snowball effect* (as illustrated in Fig. 1). The BB occurs not only when a bus platoon is formed but sooner, when the headway becomes **unstable**. The headway between two consecutive buses is defined as unstable whenever it is strictly necessary to apply a corrective action in order to recover the headway value to acceptable levels. Such threshold is usually defined in function of the frequency $f = h_1$ (the time between the departure of two consecutive buses) [6]. Let the BB occurrence be expressed as an boolean variable defined as follows

$$\text{BUNCHING} = \begin{cases} 1 \textbf{ if } \exists\ h_i \in H : h_i < f/4 \\ 0 \text{ otherwise} \end{cases} \tag{2}$$

Consequently, a relationship between the BB occurrences, the headway and the arrival time $T_{k,i}$ can be established. Let the arrival time be defined as $T_{k,i+1} = T_{k,i} + dw_{k,i} + CTT_{k,i,i+1}$ where $dw_{k,i}$ denotes the dwell time on the stop i and $CTT_{k,i,i+1}$ stands for the Cruise Travel Time between those two consecutive stops. Therefore, it is possible to anticipate the occurrence of BB events if we are able to predict the value of $dw_{k,i} + CTT_{k,i,i+1}$, which is often denominated by **Link Travel Time** [10]. In this work, we develop a probabilistic method to detect BB events that settles on Link Travel Time predictions based on the data described below.

2.1 Case Study

The source of this data was STCP, the Public Transport Operator of Porto, Portugal. It describes the trips from three distinct lines (A, B, C) during 2010. Each line has two routes – one for each way {A1, A2, B1, B2, C1, C2}. Line A is common urban line between *Viso* (an important neighbourhood in Porto) and *Sá da Bandeira*, a downtown bus hub. Line B is also an urban line but it is an arterial one. It traverses the main interest points in the city by connecting two important street markets: *Bolhão* - located in downtown - and *Mercado da Foz*, located on the most luxurious neighbourhood in the city. Line C connects the city downtown to the farthest large-scale neighborhood on the region (*Maia*).

This dataset has one entry for each stop made by a bus running in the route during that period. It has associated a timestamp, the weekday (MON to SUN) and a day type (1 for work days, 2-6 for other day types i.e.: holidays and weekends). Table 1 presents some statistics about the set of trips per route considered and the BB events identified. The BB Avg. Route Position represents the percentage of route accomplished when these events typically arise.

3 Travel Time Prediction

Let the Link Travel Time Prediction be defined as an offline **regression** problem where the target variable is the cruising time between two consecutive bus stops. Such predictions are computed in a daily basis (the forecasting horizon) using the θ most recent days (the learning period) to train our model. Consequently, we obtain a set of predictions for all the t trips of the day denoted as $\mathbb{P} = \bigcup_{i=1}^{t} P_i = \{P_{1,1}, P_{1,2}, ..., P_{t,s}\}$. These predictions are then incrementally refined in two steps: 1) trip-based and 2) stop-based. Both steps are based on the Perceptron's Delta Rule [11] by reusing each prediction's residuals to improve the further ones.

Let e denote the last trip completed before the current trip starts (i.e. c). The trip-based refinement consists into comparing the predictions to e P_e =

Table 1. Descriptive statistics for each route considered. The frequencies are in minutes.

	A1	**A2**	**B1**	**B2**	**C1**	**C2**
Number of Trips	20598	20750	20054	19361	26739	26007
Nr. of Stops	26	26	32	32	45	45
Min. Daily Trips	44	45	56	57	65	71
Max. Daily Trips	76	76	85	84	100	101
Min. Frequency	10	11	12	13	10	10
Max. Frequency	112	100	103	120	60	60
Nr. of Trips w/ BB	682	553	437	634	1917	1702
Nr. of HD events detected	63.22%	74.86%	58.31%	68.54%	49.71%	53.63%

$\{P_{e,1}, P_{e,2}, ..., P_{e,s}\}$ with the real times T_e to update P_c. Firstly, we compute the **residuals** as $R_e = T_e - P_e$ and then its average value as $\nu_e = \sum_{i=1}^{s} \frac{R_{e,i}}{s}$. Secondly, an user-defined parameter $0 < \alpha << 1$ is employed to set a threshold th able to identify trips where the error is larger than expected. Consequently, $th = \alpha * f_e$ where f_e stands for the current frequency on this route (i.e. the difference between the departure time of c and e). Three other variables are then defined: $\vartheta_p = 0, \vartheta_n = 0$ and $\beta' = \beta$. The first two are counters that are incremented whenever the prediction error is going to the same way (positive/negative) on consecutive trips (e.g. if $\mu_e > th$ ϑ_p is incremented; otherwise, $\vartheta_p = 0$). The beta value stands for the residual's percentage to be added to P_c (its initial value β is user-defined). It is initialized with another user-defined parameter $0 < \beta << 1$ and updated according to a user-defined learning rate $0 < \kappa <= 1$. Consequently, if ϑ_p or ϑ_n are incremented, the P_c and β' are updated as $P'_c = P_c \pm (\beta' \times P_c)$ and $\beta' = beta' + \vartheta * (1 + \kappa) * \beta$, respectively. If both ϑ stay the same, β' resumes its original value as $\beta' = \beta$. These updates are performed **incrementally** (i.e. whenever a real travel time for a given trip on one of its links arrives) to every trips available in the dataset. Note that the residuals are always calculated over the regression results P_c and not over the updated arrays P'_c. Thereby, its calculus is iterative but not recursive.

Given the updated predictions of two consecutive trips (P'_c, P'_{c+1}), it is possible to obtain the predicted headways $E_c = P'_{c+1} - P'_c$ while the real one is obtained as $H_c = T_{c+1} - T_c$. The calculus of E_c works as an *offline* prediction as it does not use information about the current headway experienced between the two trips. The second refinement uses the headway residuals $HR_c = H_c - E_c$ to update E_c stop-by-stop. Incrementally, we can obtain *online* headway predictions as $E'_{c,i} = H_{c,i-1} + E_{c,i} - E_{c,i-1}, \forall i \in \{2, s\}$. The problem is to update the headway online prediction for the next stop $E'_{c,i}$ given the value of $HR_{c,i-1}$. Let $\gamma' = \gamma$ be the residual's percentage to add to the prediction where its initial value for each trip $(0 < \gamma << 1)$ is an user-defined parameter. $E'_{c,i}$ can be updated as $E''_{c,i} = E'_{c,i} + (HR_{c,i-1} * \gamma')$. Finally, γ' is also updated by comparing the residuals of E_c and E'_c (HR_c and HR'_c, respectively). If $|HR_c| > |HR'_c|$, then $\gamma' = \gamma' * (1 - \gamma)$. Otherwise, $\gamma' = \gamma' * (1 + \gamma)$. The progression of γ' is bounded by an user-defined domain $[\gamma_{min}, \gamma_{max}]$. The value of $E''_{c,i}$ is also used to update the offline predictions for further stops as $E'_{c,j} = E''_{c,j-1} + E_{c,j} - E_{c,j-1} \wedge j = i + 1$ and $E'_{c,j} = E'_{c,j-1} + E_{c,j} - E_{c,j-1}, \forall j \in [i + 2, s]$. Again, whenever a newer headway value H_c, i arrives, the entire headway array $E'_{c,q}, q \in \{i + 1, s\}$ is updated accordingly. This scheme introduces a certain **flexibility** to handle the real-time stochastic usually associated to this variable.

By performing these two steps, it is possible to seize distinct levels of information to approximate the real-time link travel times incrementally. The propagation of our updates for further stops on the trip is the key to anticipate BB occurrences. The probabilistic framework devised to do so is detailed in the following section.

4 Event Detection

Let M denote a $l \times (s-1)$ matrix containing the l most recent residuals[2] for headway predictions from 1 to $s-1$ stops ahead of the current one (c) (where s is the number of stops) on a specific route, where l is an user-defined parameter to set the size of the sliding window to be employed. Consequently, $M[,i]$ represents a vector containing the most recent residuals on headway predictions i stops ahead. Departing from M, it is possible to build a rough approximation to the probability density function ($p.d.f.$) that describes the headway on a bus stop located i stops ahead. We do it so by assuming that all these distributions are **Gaussian**[3], being described by a function as $X_i = f_i(\mu, \sigma)$. μ is given by $E'_{c,i}$ or $E''_{c,1}$ while σ is given by the median value of $M[,i]$ (i.e. $\tilde{M[,i]}$). Considering the hypothesis of arising a BB event on this specific stop (i.e. H_i), we can express its likelihood as $Pr_i(X_i \leq f/4 \mid H_i)$. Such definition allows to quantify the statistical significance (i.e. p-value) of occurring a BB event on that specific stop. Using this framework, it is possible to quantify a Bunching likelihood for all the remaining stops in the route (and also to update them each time we obtain a newer value for the headway).

Using such estimations, it is possible to predict incrementally the BB occurrences in three simple steps: 1) calculate/update the Bunching likelihoods; 2) estimate a Bunching Score (BS) and 3) test if it is greater than the predefined threshold. These steps are performed each time a new headway value arrives (i.e. for each bus stop). Let j represent the latest bus stop for which the headway value is known. BS is calculated as follows: let m_j be an ordered vector (descendent) containing the likelihoods for the remaining bus stops and $n_j = 3 - ((j-1) \times 3/s) : n_j \in \mathbb{N}$ be the number of likelihoods to be used to compute BS. Finally, we have that $BS_j = m_j[\bar{1} : n_j]$ as the mean likelihood of the n_j greater ones. The BS threshold is defined in function of the frequency as $th_{BS} = 0.3 + [(f \mod \rho) * 0.1] : 0 < th_{BS} \leq 1$ where ρ is an user-defined parameter to set how many threshold levels should be defined for the frequency. Therefore, a BB event is detected if $BS_i \geq th_{BS}$. The alarm is triggered on the nearest stop where $m_i \geq th_{BS}$.

This probabilistic framework allows an *incremental* detection of the BB events by refining the headway predictions reusing not only its latest true values but also the most recent residuals. Experiments were conducted to validate this methodology. They are extensively described in the following section.

5 Experiments

On the offline regression problem, a state-of-art algorithms was employed: Random Forest (RF). We did so by following previous work on this topic which used

[2] E.g. given the newest headway value known, H_c, the residuals for the stops ranged between c and $c - l \geq 1$ are used

[3] a D'Agostino K-Squared test [12] was conducted on the headways experienced on every stop using previous data.

data from the same source [13]. The experiments were conducted using the R Software [14]. A sensitivity analysis was conducted on the regression parameters based on a simplified version of Sequential Monte Carlo method (the reader can consult the survey in [15] to know more about this topic) on previous data. The goal was to identify the best parameter setting to optimize the regression task. The best parameter setting was `mtry=3` and `ntrees=750`. The learning period used was $\theta = 7$ days. The error threshold to trigger the inter-trip update rule was set to $\alpha = 0.05$ while the initial value for the residual's percentage to be employed is $\beta = 0.01$. The learning rate *kappa* was set to 0.3. The initial residual's percentage employed on the stop-based update rule is $\gamma = 0.1$ while its domain is $\gamma \in [0.005, 0.3]$. Finally, the ρ was set to 360 seconds.

It is possible to divide the evaluation of our framework on two distinct contexts: (i) the mean absolute error and (ii) the BB detection accuracy. On the first one, we employed a prequential evaluation [16] by evaluating *just* the prediction made for the Link Travel Time performed for the next bus stop. We did so by using the Mean Absolute Error (MAE) on (1) the offline regression output and then on the (2) inter-trip and (3) intra-trip refinement. On the BB detection context, the Accuracy, the Precision and the Recall as evaluation metrics. An weighted accuracy was also employed by weighting the trips where a BB event emerge ten times more than the remaining ones. Such cost-based evaluation was done to address the different value on performing a false negative on detecting BB event - which is largely higher than raising a false positive. The Average Number of Stops Ahead is also displayed to show which is the forecasting horizon that this framework can reach. The results of these experiments are presented in the next section.

5.1 Results and Discussion

The results are presented on Table 2. More than identifying just a problematic link or stop, this framework also identifies which is the **vehicle pair** where a corrective action must be taken. In the current dataset, it was able to detect BB events thirteen stops ahead (in average), which gives more than enough room to perform any of the four possible corrective actions. Nevertheless its achievements, this framework also presents some limitations, namely, on the regression task and on the parameters employed. The regression task was tested using only one algorithm. Even considering that it presented good results in similar data [13], we do not know if there is another that could perform better using a similar computational effort. On the other hand, both the prediction refinements and the event detection framework rely on a large set of parameters. To get a fair parameter setting can be a hard task - especially if the user has no expertise on the case study approached. This issue can be specially relevant on the parameters defining the learning rates and the residual's percentages (β, κ and γ). A large-scale sensitivity analysis on these parameters must be carried out as future work.

On the first span of Table 2, it is possible to observe that the two update rules have a significant impact on reducing the MAE produced by the headway

prediction. The accuracy is high. However, the Precision is low (i.e. 52.51%). It demonstrates that our model triggers more BB alarms than necessary. This behavior can be partially justified by the **preventive** characteristics of this framework. Nevertheless its existence, it is not possible to quantify the negative impact it may have without regarding the corrective actions. By quantifying the BB probability along the route, our framework also quantifies the necessary *range* of the corrective action, which is given by $0 \leq BS_i - th_{BS} < 1$. This value can also be useful to determine which may be the corrective action to be applied in each case. The selection of a low-impact action can mitigate the effects of this over-prediction. However, such conclusions have to be validated by further experiments regarding such corrective actions (which are out of this paper scope).

6 Related Work

One of the first works to address the BB phenomenon was presented by Powell and Sheffi [17]. They devised a probabilistic model which built a set of recursive relationships to calculate the *p.d.f.* to validate the hypothesis of forming a platoon of vehicles on each stop. Nevertheless it has many similarities with the work presented here, both the relationships and the distributions were calculated based on a set of assumptions - and not on the real-time data. After this paper, many others works followed the *stability* concept (i.e. if we guarantee a stable headway, BB events will never emerge) by constantly introducing corrective actions on the system to avoid headway instability. Some examples are the work in [2], where each bus is an agent that negotiates with others the bus holding time on each station or in [4], where the negotiation is centered on the cruising speed. A more sophisticated approach to the *p.d.f.* estimation is done in [5] by accounting complex models to determine dwell times or even arrivals during such dwell times.

Table 2. Experimental results. The times are in seconds. The ALL column contains the average for the first two spans and the sum for the last one.

	A1	**A2**	**B1**	**B2**	**C1**	**C2**	**ALL**
MAE offline regression	1356.96	643.99	1475.22	1871.01	473.61	2776.57	1432.88
MAE inter-trip update	148.85	92.91	124.99	148.85	40.65	123.77	113.34
MAE incremental update	13.21	26.35	22.67	13.21	31.79	27.47	22.45
Accuracy	97.99%	96.34%	97.08%	97.83%	96.63%	93.83%	96.62%
Weighted Accuracy	93.97%	93.57%	94.57%	95.52%	95.73%	91.51%	94.14%
Precision	65.88%	40.85%	41.53%	45.70%	69.44%	51.67%	52.51%
Recall	81.81%	83.18%	83.07%	83.24%	94.48%	87.95%	85.62%
Avg. Nr. of Stops Ahead	11.85	14.78	13.88	15.01	12.96	14.52	13.83
Correct BB Predictions	558	460	363	303	1811	1497	4992
Real BB Events	682	553	437	364	1917	1702	5655

The employment of historical data to address this problem is very recent. In [3], a model to determine the optimal holding time in each station based on real-time location is presented. Delgado *et al.* [18] also suggested preventing passengers from boarding by establishing maximum holding times to maintain the headway stable. The efficiency of this type frameworks is usually demonstrated by simulations assuming i) stochastic demand and/or traffic events or 2) using historical data. Despite their usefulness, all these works do not account the historical and the real-time data. Moreover, they have a low interpretability because their outputs do not provide any clew on which is the best corrective action to took (usually, these works just pick one corrective action). The predictive method presented along this paper is able to deal with the network stochasticity, independently on which corrective action we want to take. Finally, it is important to highlight that the majority of the works on the literature try to maintain the headway stable at cost of some schedule uncertainty (introduced by the constant corrective actions), independently on the existing risk on forming a bus platoon on a further stop. By the abovementioned reasons, the authors believe that the proposed framework meets no parallel in the existing literature on this topic.

7 Final Remarks

In this paper, a probabilistic framework to anticipate the occurrence of BB events in real-time was presented. This framework employs Supervised Machine Learning techniques that incrementally refine predictions on the Link Travel Time of each bus trip. The residuals of such predictions are then used to build Gaussian Distributions on the headway values which can be used to estimate the Bunching likelihood on each bus stop. Experiments conducted on a real world data set of six bus routes running on the city of Porto, Portugal throughout an year validated this a framework as a step forward on automatizing the BB prediction task.

The present work is a proof of concept on the usefulness of predicting BB events instead of trying to maintain the headway stable at all cost. This work can be extended on three distinct axis: 1) the dataset, by including a larger dataset containing a set of lines representative of the entire network; 2) the parameter setting, by conducting a large-scale sensitivity analysis on their values and 3) on the corrective actions, by proposing a method to choose **where** and **when** a action should be took to avoid BB, as well as one to choose **which** is the best one to took in each case. Such issues comprise open research questions to be explored on future work.

Acknowledgments. The authors would like to thank STCP for the data supplied. This work was supported by the VTL: "Virtual Traffic Lights" (PTDC/EIA-CCO/118114/2010), by MAESTRA (ICT-2013-612944), by I-CITY - "ICT for Future Mobility" (NORTE-07-0124-FEDER-000064) and by ERDF - European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and also by the Portuguese Funds

through the FCT (Portuguese Foundation for Science and Technology) within project FCOMP-01-0124-FEDER-037281.

References

1. Powell, J. W., Huang, Y., Bastani, F., Ji, M.: Towards reducing taxicab cruising time using spatio-temporal profitability maps. In: Pfoser, D., Tao, Y., Mouratidis, K., Nascimento, M.A., Mokbel, M., Shekhar, S., Huang, Y. (eds.) SSTD 2011. LNCS, vol. 6849, pp. 242–260. Springer, Heidelberg (2011)
2. Gershenson, C., Pineda, L.: Why does public transport not arrive on time? the pervasiveness of equal headway instability. PloS One 4(10), 72–92 (2009)
3. Daganzo, C.: A headway-based approach to eliminate bus bunching. Transportation Research Part B 43(10), 913–921 (2009)
4. Daganzo, C., Pilachowski, J.: Reducing bunching with bus-to-bus cooperation. Transportation Research Part B: Methodological 45(1), 267–277 (2011)
5. Bellei, G., Gkoumas, K.: Transit vehicles' headway distribution and service irregularity. Public Transport 2(4), 269–289 (2010)
6. Moreira-Matias, L., Ferreira, C., Gama, J., Mendes-Moreira, J., de Sousa, J.F.: Bus bunching detection by mining sequences of headway deviations. In: Perner, P. (ed.) ICDM 2012. LNCS, vol. 7377, pp. 77–91. Springer, Heidelberg (2012)
7. Wang, F.: Toward intelligent transportation systems for the 2008 olympics. IEEE Intelligent Systems 18(6), 8–11 (2003)
8. Mishalani, R., McCord, M., Wirtz, J.: Passenger wait time perceptions at bus stops: empirical results and impact on evaluating real-time bus arrival information. Journal of Public Transportation 9(2), 89 (2006)
9. Strathman, J., Kimpel, T., Dueker, K.: Transportation Northwest: Bus transit operations control: review and an experiment involving tri-met's automated bus dispatching system. Technical report, Transportation Northwest, Department of Civil Engineering, University of Washington (2000)
10. Chen, G., Yang, X., An, J., Zhang, D.: Bus-arrival-time prediction models: Link-based and section-based. Journal of Transportation Engineering 138(1), 60–66 (2011)
11. Rosenblatt, F.: The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review 65(6), 386 (1958)
12. D'Agostino, R.B.: Transformation to normality of the null distribution of g1. Biometrika, 679–681 (1970)
13. Mendes-Moreira, J., Jorge, A., de Sousa, J., Soares, C.: Comparing state-of-the-art regression methods for long term travel time prediction. Intelligent Data Analysis 16(3), 427–449 (2012)
14. R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2012)
15. Cappé, O., Godsill, S., Moulines, E.: An overview of existing methods and recent advances in sequential monte carlo. Proceedings of the IEEE 95(5), 899–924 (2007)
16. Dawid, A.: Present position and potential developments: Some personal views: Statistical theory: The prequential approach. Journal of the Royal Statistical Society. Series A (General) 147, 278–292 (1984)

17. Powell, W., Sheffi, Y.: A probabilistic model of bus route performance. Transportation Science 17(4), 376–404 (1983)
18. Delgado, F., Muñoz, J.C., Giesen, R., Cipriano, A.: Real-time control of buses in a transit corridor based on vehicle holding and boarding limits. Transportation Research Record: Journal of the Transportation Research Board 2090(1), 59–67 (2009)