Transportation Research Part E 72 (2014) 192–209

Contents lists available at ScienceDirect

Transportation Research Part E

journal homepage: www.elsevier.com/locate/tre

Proactive vehicle routing with inferred demand to solve the bikesharing rebalancing problem

Robert Regue *, Will Recker

Department of Civil and Environmental Engineering, and Institute of Transportation Sciences, University of California, Irvine, 4000 Anteater Instruction and Research Building (AIRB), Irvine, CA 92697-3600, USA

ARTICLE INFO

Article history:
Received 13 February 2014
Received in revised form 30 September 2014
Accepted 9 October 2014

Keywords:
Bikesharing
Dynamic rebalancing problem
Proactive vehicle routing
Gradient boosting
Simulation

ABSTRACT

Bikesharing suffers from the effects of fluctuating demand that leads to system inefficiencies. We propose a framework to solve the dynamic bikesharing repositioning problem based on four core models: a demand forecasting model, a station inventory model, a redistribution needs model, and a vehicle-routing model. The approach is proactive instead of reactive, as bike repositioning occurs before inefficiencies are observed. The framework is tested using data from the Hubway Bikesharing system. Simulation results indicate that system performance improvements of 7% are achieved reducing the number of empty and full events by 57% and 76%, respectively, during PM peaks.

Published by Elsevier Ltd.

1. Introduction

Bikesharing is a sustainable and environmentally friendly transportation mode that offers bikes "on-demand" to improve daily urban mobility. A typical current bikesharing system operates as follows: a member can pick up a bike from any of the stations available in the system and must return it before a predefined time period to any other station that has empty docks available. Stations have a fixed capacity and a time limit is imposed to ensure high bike usage and bike rotation.

Bikesharing systems compete with other forms of public transportation in urban environments. In the United States, as of 2012 there were 15 IT-based bikesharing programs (Shaheen et al., 2012) and major US cities, such as New York, San Francisco, Chicago, Forth Worth or Columbus launched their own bikesharing programs during 2013. Similar trends are observed around the world (Meddin and DeMaio, 2007; Shaheen et al., 2012).

Although bikesharing systems potentially offer a viable alternative for enhancing urban mobility, they suffer from the effects of fluctuating spatial and temporal demand that inherently lead to severe system inefficiencies; e.g. having empty or full stations for long periods of time. These inefficiencies are embedded in the fabric of bikesharing because one-way trips are allowed and the operator has little control over the behavior of the users. As a result, some stations are empty and some others are full, impeding potential users to either pick up or drop off bikes at their desired stations, degrading the level of service, system performance and causing disappointment that may result in loss of users. To resolve these inefficiencies, bikesharing operators are compelled to reposition bikes dynamically to avoid the system from collapsing (Fricker et al., 2012). It has also been demonstrated that knowledge of future demands can aid in these repositioning tasks, reducing relocation costs and increasing the system performance (Barth and Todd, 1999).

* Corresponding author. Tel.: +1 949 824 5989; fax: +1 949 824 8385.
E-mail addresses: rreguegr@uci.edu (R. Regue), wwrecker@uci.edu (W. Recker).

http://dx.doi.org/10.1016/j.tre.2014.10.005
1366-5545/Published by Elsevier Ltd.

The research presented in this paper outlines a comprehensive framework to solve the dynamic bikesharing rebalancing problem—finding the optimal routes and inventory levels to keep the bikesharing system balanced while it is in operation (Caggiani and Ottomanelli, 2012; Contardo et al., 2012; Rainer-Harbach et al., 2013; Raviv et al., 2013; Schuijbroek et al., 2013). The framework is based on four core models: (1) a demand forecasting model at the station level, (2) a station inventory model, (3) a redistribution needs model, and (4) a vehicle routing model.

The dynamic bikesharing rebalancing problem can be seen as a One-Commodity Pickup-and-Delivery Vehicle Routing Problem (1-PDTSP) (Hernandez-Perez and Salazar-Gonzalez, 2004) with the added complexity that the inventory at the stations is flexible (Schuijbroek et al., 2013). We see the methodology presented in the paper as an heuristic to solve such a problem that on its core uses anticipated future demands to decouple the inventory and the routing problem, reducing the complexity, making it scalable, proactive instead of reactive and allowing for real time decision-making. Vehicle routes are built dynamically based on current and expected events in a proactive manner, as inefficiencies are resolved before they actually occur, increasing customer satisfaction. As routes are being built periodically, operator interaction is permitted, overriding current routing decisions.

The routing problem maximizes the utility gained by removing inefficiencies from the system, it is selective (not all stations are visited), keeps track of the vehicle inventory, can handle a non-homogenous fleet and allows for pick ups or drop offs at buffering stations—stations that are in a balanced state but some bikes can be removed or added without causing future inefficiencies—solving the issue of having an empty or full vehicle that is not able to respond to existing inefficiencies.

The proposed predictive model has the potential to help determine more efficient user-based relocation policies by means of incentives or dynamic pricing policies. It is also self-adaptive, as it is regularly retrained as new system data are being acquired. Further enhancements to the predictive module can be made if bikesharing system users data were available—for example, offering discounts to users that express the need of a bike at a given station using a mobile application. Doing so can improve the predictive model and lead to better routing decisions.

Although the models developed in this paper pertain specifically to the dynamic rebalancing problem in bikesharing, the general framework has application in dynamic logistics resource operations and management in various other demand–supply operational scenarios; e.g., directly in the balancing of electric vehicles in the ZEV-NET shared-use station car system (City of Irvine, 2014), or with some modification to the problem of distribution of emergency medical personnel (by type of specialty) in such disasters as earthquakes and hurricanes. Furthermore, models developed on the framework could also be implemented to efficiently distribute vehicles in a hypothetical one-way carsharing system using a fleet of autonomous vehicles.

The framework has been tested under various simulation scenarios with variable time steps using data from The Hubway Bikesharing system in Boston (Hubway, 2011). The simulation results show that level of service can be improved compared to the "do nothing" scenario, especially in reducing the observed number of full and empty events. Managerial decisions are also simulated, testing for the impact of the number and the capacity of the vehicles used for rebalancing operations.

The structure of the paper is as follows. Section 2 reviews current literature on solving the bikesharing rebalancing problem. Section 3 describes the framework, methodology behind each model and data used. Section 4 outlines the simulation procedure. Section 5 reports the results under different simulation scenarios, and conclusions are drawn in Section 6.

2. Related work

Bikesharing-related literature has been growing as more and more systems are being implemented. Relative to issues addressed in this paper, there are two main areas of interest: forecasting future demands in shared ride systems, and approaches to formulate and solve the dynamic bikesharing rebalancing problem.

Concerning forecasting future demands a variety of techniques have been explored. Initial insights can be found in the carsharing literature, which has a longer history of investigation. Barth and Todd (1999) show under a simulation framework that for a one-way carsharing system the knowledge of future demands significantly impacts performance measures. Four different predictive techniques are tested on real data from the Honda Intelligent Community Vehicle System (ICVS) in Kek et al. (2005): Neural Networks (NN), regression, selective moving average and Holt's model. The results indicate that NN has the best performance. Based on this research, Cheu et al. (2006) ran tests on an expanded dataset of ICVS comparing NNs and Support Vector Machines (SVM). Their results also show that NNs lead to better performance and it is argued that they can better capture nonlinearities in the system. These results motivated the later implementation of a decision support system to optimize operator-based relocation operations in carsharing systems (Kek et al., 2009), which is modeled as a variation of a pick-up and delivery problem.

In the bikesharing literature, Froehlich et al. (2009) and Kaltenbrunner et al. (2010) use data from the *Bicing*, the bikesharing system in Barcelona (Spain). Froehlich et al. (2009) test four different predictive techniques: last value, historic mean, historic trend and a Bayesian Network (BN). The best results are obtained with the BN, with an average error of 8%, averaged over all days and prediction windows used (10, 20, 30, 60, 90 and 120 min). As expected, prediction errors increase with the prediction window. Kaltenbrunner et al. (2010) implement an Auto-Regressive Moving Average (ARMA) model with an FIR low-pass filter to predict station states. Mean absolute errors in a 60-min prediction window of 1.39 bikes with a maximum error of 6 bikes are reported.

Borgnat et al. (2011) use data from Vélo'v, Lyon's (France) bikesharing system to predict hourly rentals. The problem is decomposed into first predicting the non-stationary amplitude in a day and then adding hourly fluctuations. The amplitude is modeled as a Linear Regression using external factors to account for time and season. Hourly fluctuations are modeled as an autoregressive process of order 1 with exogenous inputs. Apparently, they are the first to include explanatory variables other than those related to historical system data in their models.

Caggiani and Ottomanelli (2012) use NN to forecast arrivals and departures for their decision support system that solves the bikesharing rebalancing problem. NNs are selected on the grounds that previous studies (Cheu et al., 2006; Kek et al., 2005) demonstrated its applicability. The independent variables set includes weather conditions.

More recently, Henderson and Fishman (2013) use Poisson regressions to predict arrivals and departures for the next 60 min and estimate the probability of stations being empty or full in the future. The model is tested on Divvy Bikes, the bikesharing system from Chicago.

The above approaches to predict bikesharing station states do not include external factors as independent variables. Only two studies, (Borgnat et al., 2011; Caggiani and Ottomanelli, 2013) test for weather and holidays, showing improvements of more than 50% in predictive power when those factors are included. Gebhart and Noland (2013) study the impact of weather on trips made at Capital Bikeshare, in Washington DC, considering the precise weather conditions when the trip is made and conclude that fewer trips are made in rainy, humid, windy and cold conditions.

Another common assumption is that the same explanatory variables work for all stations, meaning that the same model can be used to make forecasts for individual stations. However, our tests indicate that in bikesharing systems the behavior of the stations is extremely dynamic, and that behavior need not be correlated among them. As a result, either a variable selection step specific to each station should be introduced or the predictive technique should internally account for that.

Additionally, most of the current predictive models do not attempt to be used as inputs to the bikesharing rebalancing problem, but rather attempt either to explain bikesharing usage and riders' behavior (Borgnat et al., 2011; Froehlich et al., 2009) or to provide information to the users and operators about the system (Kaltenbrunner et al., 2010) such that better design and planning policies can be implemented.

The bikesharing–rebalancing problem is typically decomposed into the static and the dynamic rebalancing and can be seen as a variation of the pickup and delivery problem (PDP). Static rebalancing occurs during periods where demand is negligible (overnight) and the main goal is to set the stations to the optimal inventory level such that dynamic repositioning is minimized when the system is in operation. Most of the focus in the literature has been on formulating and solving the static rebalancing problem (Chemla et al., 2012; Nair et al., 2013; Rainer-Harbach et al., 2013).

In terms of finding optimal inventory levels, Raviv and Kolka (2013) introduce a dissatisfaction measure based on the number of renters and returners that abandon the system. The measure is a function of the initial inventory level at the beginning of each study period and the goal is to find the optimal inventory level such that the dissatisfaction is minimized. The dynamics of the station inventory are modeled as a continuous time Markov chain. The estimated inventory levels are tested on the Tel-O-Fun bikesharing system in Tel Aviv (Israel) by computing the initial inventory of each station at the beginning of the day. The performance of the proposed inventory levels is measured based on the number of shortages observed during the day and compared to current practices. Results indicate a 17% reduction of observed shortages during the day, reducing the cost of the dynamic repositioning.

A different approach to determine optimal inventory levels is presented in Lu (2013). A bikesharing system is represented using a time–space network flow model and the goal is to minimize the system total cost, including holding cost at station, bike supply costs, redistribution costs and the cost of losing customers due to unmet demand. The solution to the problem is the optimal bike allocation and the redistribution flows. To account for demand uncertainty, robust optimization techniques are introduced. Numerical tests are carried out using the bikesharing system in Banciao District, in New Taipei City (Taiwan).

Schuijbroek et al. (2013) combine both the inventory and the routing problems into a single framework. Optimal inventory levels are first estimated from historical data. Each station is individually modeled as a M/M/1/K queuing model. The inventory is then fed to the routing problem, minimizing the maximum tour length of the vehicles. The problem is solved in a rolling horizon fashion using a cluster-first, route-second heuristic. The model performance is tested on The Hubway and Capital Bikeshare bikesharing systems from Boston and Washington, respectively.

Caggiani and Ottomanelli (2013) also address both problems proposing a decision support system that considers stochastic demand and minimizes vehicle reposition cost. The day is divided into fixed time intervals. At the beginning of each time interval future demands are updated and the decision support system is launched. The outputs of the model are the relocation matrix and the relocation path of the redistribution vehicles that minimize the total operator costs, including relocation costs and lost users costs. The model is tested on a 5-station bikesharing system discretizing the day into 5-min intervals under three different demand scenarios. The results indicate that significant reductions of lost users can be achieved.

Pfrommer et al. (2013) use model-based receding horizon optimization techniques to combine operators' relocation operations and user-based relocations by means of a reward system. The goal is to use users to do the repositioning instead of the operator. Truck routes and incentives are computed online at periodic time instances and a predictive model is introduced to estimate the expected evolution of the system in the near future. Truck routes are also modeled under a time-expanded network and the solution approach is based on a greedy heuristic. The model was tested on data from the London's Barclays Cycle Hire (UK) and results indicate that price incentives can be used to reduce system inefficiencies, but that they are not enough to increase service levels considerably during weekday operations.

In this paper we focus on the dynamic problem. We propose to find the optimal inventory levels at each station and the repositioning vehicles optimal routes to redistribute bikes across the system.

Current methodologies rely on solving a variation of the pickup and delivery problem (PDP) to compute optimal vehicle routes. However, PDP is an NP hard problem that becomes intractable for large bikesharing systems. As a result, current practice consists of finding heuristics that can quickly find a near-optimal solution to the routing problem for a given time period. Here, we propose a different approach to reduce the complexity of the problem building on the idea of proactive dynamic vehicle routing (Ferrucci, 2013) and the use of machine learning techniques. We first combine different datasets to infer future station inventory levels. Based on the inferred inventory levels, a stochastic linear integer problem is solved to determine the estimated number of bikes required at the stations with inefficiencies. Once pickup and drop off events are identified, they are treated as deterministic and are the input to a vehicle routing model that optimally routes vehicles based on anticipated events. The vehicle routing problem is solved for a single vehicle at a time and is limited to those stations that have inefficiencies and are within threshold distance of the current station, bounding the size of the problem and allowing the use of traditional solvers to find a solution.

The proposed proactive vehicle routing approach differs from current proactive routing approaches such as the hybrid predictive control (Núñez et al., 2013) or the real-time control and request-forecasting (Ferrucci, 2013). The previous approaches focus on real time deliveries and paratransit type problems and incorporate into the objective function expected future events. Vehicles are routed through areas with higher probabilities for events to occur in future time steps so that when an event occurs, vehicles are able to respond faster. In our approach we also include this idea into the objective function, but the bikesharing rebalancing problem requires the solution of an inventory model first. Therefore, we not only use future expected demands in the routing problem but also to anticipate inventory needs, successfully combining machine learning techniques with mathematical programming. In this manner, we have a truly proactive approach as vehicles are routed before the events occur, not only focusing on minimizing the routing costs because the vehicles are closer to an event that is about to happen.

3. Framework

The framework we propose has four main models: (1) a demand forecasting model at the station level, (2) a station inventory model, (3) a redistribution needs model, and (4) a vehicle routing model. The relationship between the various models and the available data in most new-generation bikesharing systems is shown in Fig. 1. Dashed boxes represent available data that serves as an input to the models.

Historical data is fed into the demand forecasting model and the inventory levels model. By historical data we refer to the inventory level time series prior to the current time and other such related data as weather and transit data that may improve the accuracy of the forecasting model.

The goal of the forecasting model is to anticipate inventory levels at different time windows from the current time; for example, after 20, 40 and 60 min, based on current system states. The inventory levels model uses historical data to determine the optimal initial inventory level such that in the next time period (next 20 min) system inefficiencies—defined by either pickup or drop off events—will not occur.

The output of these two models allows us to compute the system redistribution needs. In other words, for each station in the system we determine the number of bikes that should be picked up or dropped off such that at the beginning of the next time period the station will have the initial optimal inventory level. The final step of the process is to dispatch and route the available vehicles to address system inefficiencies. In this latter step, system operators can interact, altering

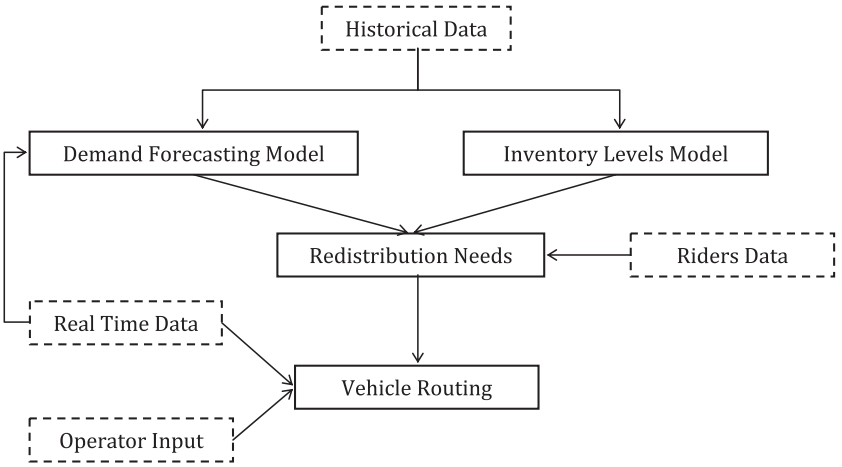

Fig. 1. Model interactions. — Available Data. – Models.

vehicle-dispatching decisions based on their own expertise or unpredicted events that the demand-forecasting model cannot anticipate.

The demand-forecasting and inventory levels models are periodically retrained, as system data are being acquired. For example, the predictive model can be updated weekly using data from the past week to train the new model. Additionally, if riders' data were available by matching user ids with bike travel patterns, a user-based relocation procedure could be incorporated into the redistribution needs model. As future inefficiencies are anticipated, it should be possible to contact users in advance and offer them an incentive to change their behavior.

We understand the proposed framework as an heuristic by itself that successfully decouples the inventory and the routing problem and that intrinsically handles the vehicle decomposition in the routing problem. As a result, the framework is scalable, both, in terms of the number of stations and vehicles used for the repositioning. As detailed in coming sections, the routing problem is bounded and the other three models, the rebalancing needs model (P1), the station inventory model and the predictive model are defined at the station level and solved for a single station at a time.

Details on each model and the data available are given in the following subsections.

3.1. Data description

We use a dataset from The Hubway Bikesharing system in Boston. The Hubway was launched in July 28, 2011 with 600 bikes and 61 stations (Hubway, 2011). The data were made publically available for The Hubway Data Visualization Challenge (Hubway, 2012) and include:

- *Trip history data:* all trips made by users detailing length, departure and arrival times and origin and destination stations.
- *Rebalancing operations aggregated at the daily level:* the number of bikes that were picked up and dropped off for rebalancing purposes in a given day.
- *Station snapshot data:* number of available bikes and docks per station in 1-min intervals.
- *Other related data:* census information, neighborhoods, elevation, employment characteristic, population, vehicle miles traveled, etc.

In our study we use only the trip history data and the station snapshot data from Sunday May 6th to Sunday July 29th 2012, a period in which the number of working stations was constant at 61 stations (during the following months the system gradually expanded, having 108 stations and more than 1000 bikes before it was shut down for the winter season in October 2012).

It should be noted that the station snapshot data already contains the rebalancing operations undertaken by the operator.

The dependent variable represents the number of available bikes and it is drawn from the station snapshot data and grouped into 10-min intervals. The mean of the observations that fall within the 10-min intervals is then taken.

The Hubway data are merged with hourly weather data for station 14739 at Logan International Airport from the National Climatic Data Center (National Climatic Data Center, 2012), and with sunset and sunrise data from (Time and Date AS, 1995)—in Gebhart and Noland (2013) the authors claim that bike riders behavior is affected by daylight and darkness.

As a result, all of the available variables that can be used in the demand-forecasting model are shown in Table 1.

Table 1
Variable definition. NCDC: National Climatic Data Center.

Variable	Description	Source
Number of bikes	Number of bikes at the station at current time t	Hubway
Number of bikes at Station X	Number of bikes at surrounding stations X at current time t	Hubway
Number of bikes at Delta Y	Number of bikes that were present at current station, Y hours before the prediction (i.e. -1 h, -24 h, -168 h)	Hubway
Hourly Rain	Dummy variable (0,1) that indicates if it rained during current time t	NCDC
Hourly Drizzle	Dummy variable (0,1) that indicates drizzling during current time t	NCDC
Hourly Shower	Dummy variable (0,1) that indicates if it shower during current time t	NCDC
Hourly Temp	Mean temperature in Fahrenheit for current time t	NCDC
Snowfall	Dummy variable (0,1) that indicates if it snowed during the day	NCDC
Hourly Humidity	Relative humidity for current time t	NCDC
Wind Speed	Average wind speed in miles per hour	NCDC
Visibility	Statute visibility in miles	NCDC
Daylight	Dummy variable (0,1) that indicates if it was daylight or dark	Time & date
Weekday	Categorical variable representing the weekday (1-Mon...7-Sun). It can optionally be set to 0 or 1 to disaggregate between week days or weekends	
Hour	Hour of the current time t	
Minute	Minute of the interval for the current time t	
US Holidays	Dummy variable (0,1) that indicates if a given day was an official holiday	Google Calendar
Station Activity	Standard deviation of the number of bikes at the current station during the last 6 time intervals	

3.2. Demand forecasting model

The demand forecasting model is based on a relatively new prediction method—gradient boosting machines (GBM), introduced by Friedman (2001, 2002). Gradient boosting is a supervised and regression-based machine learning method that repeatedly fits a weak classifier (typically a decision tree) and ensembles them to make the final prediction.

The use of GBM in the transportation literature is limited. However, it has been successfully implemented to enhance reliability in real-time risk assessment (Ahmed and Abdel-Aty, 2013) and to forecast traffic flow under abnormal conditions (Wu et al., 2012).

The main advantages of using GBM, as detailed in Friedman (2001), are that its performance is invariant to transformations in the explanatory variables and insensitive to outliers. Furthermore, variable selection is internalized in the decision tree, making the algorithm robust toward irrelevant input variables, and there is no need to impute missing values since the decision tree can also handle them. Also, overfitting is avoided by carefully selecting the regularization parameters: learning rate (v) and number of iterations (M). Another advantage is that there is no need to retrain the entire model when new data are acquired, as the boosting procedure can continue from the previous model, saving computational time and making GBM attractive for online applications. Finally, compared to such other black box techniques as Neural Networks, the importance of the explanatory variables can be measured, easing the interpretation of model results and coefficients.

As in any regression method, the goal of GBM is to find the model $F(\tilde{\boldsymbol{x}})$ that minimizes the expectation of the loss function $\Psi(y, F(\tilde{\boldsymbol{x}}))$ given a dataset $D = \{(\tilde{\boldsymbol{x}}_1, y_1), \ldots, (\tilde{\boldsymbol{x}}_N, y_N)\}$, with input, or explanatory, variables $\tilde{\boldsymbol{x}}$ and output variable, or prediction, y (Eq. (1)). The estimated model $\widehat{F}(\tilde{\boldsymbol{x}})$ is found by the weighted sum of weak models $h(\tilde{x}_s, a_m)$. In this study h is a decision tree in which the input variable $\tilde{\boldsymbol{x}}_s$ is a sub sample of $\tilde{\boldsymbol{x}}$ drawn at random and a_m are the parameters of the mth decision tree (Eq. (2)). Random sampling introduces stochastic behavior, which reduces computational times and improves accuracy (Friedman, 2002).

$$F^*(\tilde{\boldsymbol{x}}) = \arg\min_{F(\tilde{\boldsymbol{x}})} E_{y,\tilde{\boldsymbol{x}}}[\Psi(y, F(\tilde{\boldsymbol{x}}))] \tag{1}$$

$$\widehat{F}(\tilde{\boldsymbol{x}}) = \sum_{m=1}^{M} \beta_m h(\tilde{\boldsymbol{x}}_s, a_m) \tag{2}$$

The optimal weights β_m are derived in a greedy approach, minimizing (Eq. (3)) following a forward stagewise procedure. Starting from a constant function, $F_0(\tilde{\boldsymbol{x}})$, the new model is updated using (Eq. (4)). The shrinkage parameter, v, is a regularization parameter that controls for overfitting. Typically, smaller values of v lead to better estimates, but there is a tradeoff with the number of iterations M. Both parameters need to be calibrated using cross-validation or using an independent dataset not used in training the algorithm.

$$(\beta_m, a_m) = \arg\min_{\beta,a} \sum_{i=1}^{N} \Psi(y_i, F_{m-1}(\tilde{\boldsymbol{x}}_l) + \beta h(\tilde{\boldsymbol{x}}_l, a)) \tag{3}$$

$$F_m(\tilde{\boldsymbol{x}}) \leftarrow F_{m-1}(\tilde{\boldsymbol{x}}) + v\beta_m h(\tilde{\boldsymbol{x}}_s, a_m) \tag{4}$$

In this implementation, the R package *gbm* by Ridgeway (2012) and the Gaussian loss function have been used. The subsample size for the stochastic GBM is set to 0.4.

For training, validating, and testing the models the original dataset was first split into training, validation and test datasets. The validation data are used to validate the model trained on the training data set. Data from Sunday July 15th to Saturday July 21st (1 week) are considered for validation purposes. The test dataset, which has been excluded from previous model calibration steps, goes from Sunday July 22nd to Saturday July 28th (1 week).

The details of the calibration procedure and variable selection are shown in Regué and Recker (2014). Prediction accuracies are compared to those obtained using Neural Networks (NN), Linear Regression (LR) and Last Known Value (LKV). The results show that GBM does not need an extensive calibration procedure, meaning that the same set of parameters, including algorithmic parameters and those altering the independent variables set, can be used for all stations, reducing computational time.

Three different prediction time windows are tested—20, 40 and 60 min. Using the root mean squared error as an error measure, GBM models without calibration perform 1.33%, 8.7% and 13.27% better for the 20-, 40- and 60-min predictions, respectively, than the equivalent Neural Network model.

The mean error for the GBM model is 3.6%, 9.5% and 11.2% of the capacity of the station for the 20-, 40- and 60-min time windows, respectively. For example, the mean error of predictions for a station with a capacity of 15 bikes, would be expected to be 0.55, 1.42 and 1.68 bikes, for the respective time windows.

An important advantage of GBM over other models is that the validation dataset is not required. This is shown in Regué and Recker (2014), where the tradeoffs between using more recent data to train the model or fine tuning the parameters by means of the validation data set are analyzed.

3.3. Inventory levels model

Initial inventory levels are computed borrowing the concept of level of service described by Schuijbroek et al. (2013). The underlying idea is to determine whether or not the initial inventory level in a given station is enough to serve the future demand for a certain time period. Each station is modeled as a M/M/1/K process, K being the capacity of the station. Arrivals and departures are assumed Poisson and its parameters are found from historical data. The outputs are $s_{i,t}^{min}$ and $s_{i,t}^{max}$, which represent the minimum and maximum inventory levels at station i and time t such that if the initial inventory level $s_{i,t}$ falls between that range, no action is required with some level of confidence for pickups β_i^+ and for drop offs β_i^-. Table 2 summarizes the different events that can occur.

Compared to Schuijbroek et al. (2013) where data from the early stages of the Hubway are used (November 1st, 2011 to May 31st, 2012) and the time interval is set to 1 h, here we set the time interval to 20 min and use the same dataset for predictions. Note that in our dataset the activity in the system is greater, as the system was well established and more trips are observed, allowing for a shorter time window. Service level requirements parameters (β_i^-, β_i^+) are set to 95% for all stations.

As an example of the disparate nature of activity among the various stations, Fig. 2 shows inventory levels as computed in Schuijbroek et al. (2013) relative to the station capacity at stations 14 and 38, together with the observed number of bikes on July 20th 2012. For the computation of inventory levels, only weekday data between 6:00 and 23.59 have been used. Outside this interval there are not enough trip observations to properly estimate the arrival and departure pattern distributions. Note the different behavior of each station and how throughout the day, pickup and drop off events should occur to bring the station into a balanced state. Station 14 has most of the arrivals during the AM peak where the maximum inventory threshold is lower. Inventory levels for station 38 vary during the day due to its higher activity. Despite the large number of docks

Table 2
Initial inventory levels and event types.

Initial inventory level	Event type
$s_{i,t} \leqslant s_{i,t}^{min}$	Drop off
$s_{i,t}^{min} < s_{i,t} < s_{i,t}^{max}$	Self-sufficient station
$s_{i,t} \geqslant s_{i,t}^{max}$	Pickup

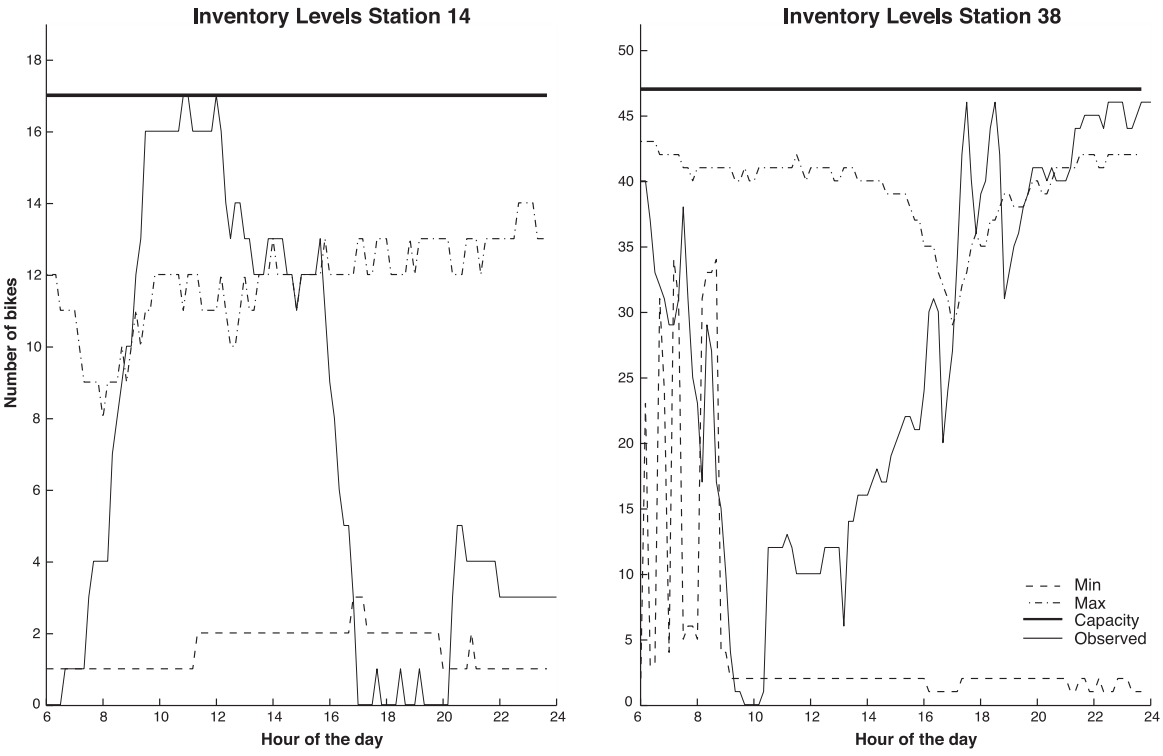

Fig. 2. Inventory levels for station 14 and station 38.

available, station 38 gets empty at the end of the morning peak (9:30 AM) and full during the PM peak. As a result, the minimum inventory threshold is higher during the AM peak and relatively constant the rest of the day. The maximum inventory level is lower at the beginning of the PM peak, so that future arrivals can be accommodated.

As discussed in Schuijbroek et al. (2013), although none is observed in their data set, three different types of infeasibilities can occur. In our data we do observe situations where the maximum inventory level is smaller than the minimum, $s_{i,t}^{max} < s_{i,t}^{min}$. This implies that given the current arrival and departure patterns the station cannot be balanced. In such cases operators need to prioritize. From an operational point of view, operators tend to focus on ensuring that docks are available instead of bikes, as riders may incur late fees if they go over the maximum riding time. These singularities are observed in stations with high activity during peak hours and small capacities (stations 35, 41, 47 and 58). In such events we set $s_{i,t}^{min} = s_{i,t}^{max}$.

3.4. Redistribution needs

Decisions on the number of bikes to be picked up $(y_{i,t}^{-})$ or dropped off $(y_{i,t}^{+})$ are made solving a stochastic linear integer program that takes as input inventory levels and predictions at different time windows, both being the outputs of the two previous models. Predictions $(\hat{s}_{i,t})$ are treated as normal distributed variables with mean the prediction and variance equal to the variance of the error made in the predictions for the test set.

The logic behind this step is to ensure that by the beginning of the next time window (in this case 20 min) all pickups and drop offs such that all of the stations are in balance are known (i.e., the initial inventory level $\hat{s}_{i,20}$ falls between the range $s_{i,20}^{min} < \hat{s}_{i,20} < s_{i,20}^{max}$ for all stations i). Note that in here we do not use the current observed inventory at the stations nor the arrival or departure patterns. We are only interested in making sure that at the beginning of the next time period, the number of bikes at the stations will fall between the optimal inventory ranges.

Instead of simply checking predictions $\hat{s}_{i,20}$ that do not fall in the optimal range and setting them to the boundary (i.e., $y_{i,t}^{+} = s_{i,20}^{min} - \hat{s}_{i,20}$ or $y_{i,t}^{-} = \hat{s}_{i,20} - s_{i,20}^{max}$ for all stations i), a linear problem is formulated in such a way that future predictions $\hat{s}_{i,40}$ and $\hat{s}_{i,60}$ are considered in determining $y_{i,t}^{-}$ or $y_{i,t}^{+}$. In this manner, we make a better use of the information gained from the predictive model. This can be observed in Fig. 3, where in the particular case represented, we want to find the number of bikes to be removed (y^{-}) from the station before t equals 20 min such that system inefficiencies in later time steps do not occur. In this process we are assuming that demand is constant during the study period. As a result, the number of bikes to be picked up or dropped off during the time period of analysis is minimized together with the probability to revisit one station in future time steps.

The problem is formulated as a chanced constrained problem with stochastic constraints having a confidence level α and independently solved for each station. As a result, for each station i in the set of stations $\mathcal{S}$ problem P1 is solved.

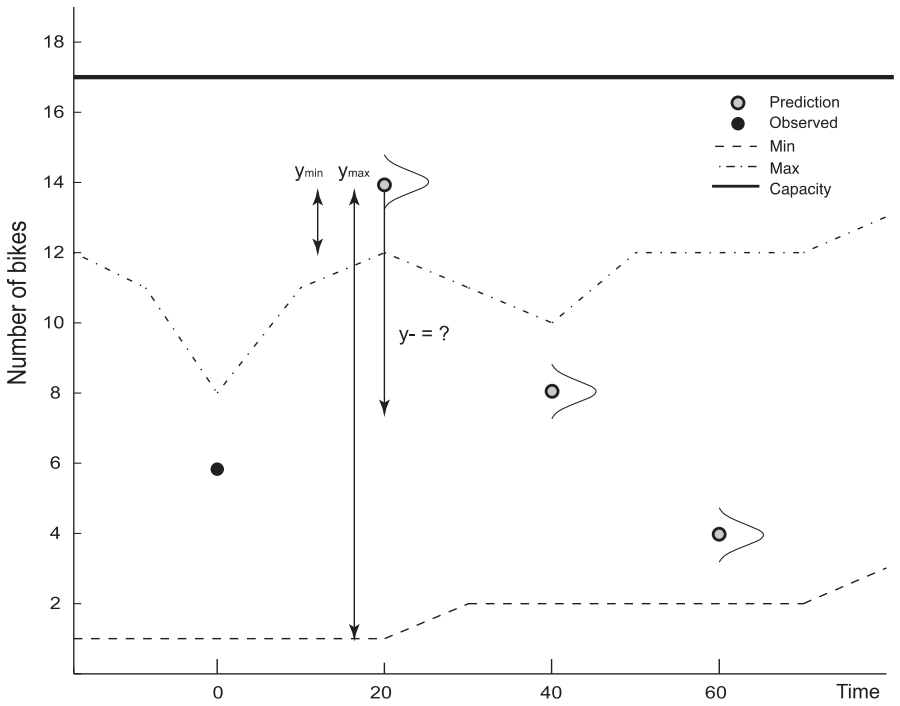

Fig. 3. Redistribution needs example.

P1:

$$\min \sum_{t\in T}\omega_t \cdot \left(y^+_{i,t}+y^-_{i,t}\right)$$

st.

$$Pr\left\{s^{min}_{i,t}-\hat{s}_{i,t}-\sum_{k=1}^{t}\left(y^+_{i,k}-y^-_{i,k}\right)\le 0\right\}\geqslant \alpha,\quad \forall t\in T \tag{1}$$

$$Pr\left\{-s^{max}_{i,t}+\hat{s}_{i,t}+\sum_{k=1}^{t}\left(y^+_{i,k}-y^-_{i,k}\right)\leqslant 0\right\}\geqslant \alpha,\quad \forall t\in T \tag{2}$$

$$Pr\left\{y^-_{i,t}-\hat{s}_{i,t}\leqslant 0\right\}\geqslant \alpha,\quad \forall t\in T \tag{3}$$

$$Pr\left\{y^+_{i,t}-C_i+\hat{s}_{i,t}\leqslant 0\right\}\geqslant \alpha,\quad \forall t\in T \tag{4}$$

$$y^-_{i,t}, y^+_{i,t}\in \mathbb{N}$$

where the set $T = \{20, 40, 60\}$ is the set of time windows and C_i is the capacity of station i. Weights (ω_t) are introduced in the objective function to penalize early deliveries, being larger for closer time windows ($\omega_1 > \omega_t > \omega_T$). This is to avoid picking up or dropping off bikes in earlier time steps than when they are needed. For example, if $s^{max}_{i,60}-\hat{s}_{i,60}=-3$, the desired solution would be $y^-_{i,60}=3$, but $y^-_{i,20}=3$ and $y^-_{i,40}=3$ are also feasible solutions. If all weights are equal to one, they all have the same objective function value, 3. Introducing the weights pushes the solution to be $y^-_{i,60}=3$.

The first two constraints ensure that after picking up or dropping off bikes the number of remaining bikes will be between the maximum and minimum inventory ranges. The last two constraints ensure that we do not pick up more bikes than the ones that are currently in the stations and that we do not drop off more bikes than docks available.

The above chanced constrained problem can be turned into its deterministic equivalent and solved with traditional linear programming solvers.

Under some circumstances, especially when a station has a highly dynamic behavior, a solution may not exist to problem P1. If this is the case, we set $y^-_{i,20}$ to $\hat{s}_{i,20}-s^{max}_{i,20}$ or $y^+_{i,20}$ to $s^{min}_{i,20}-\hat{s}_{i,20}$.

Using the above concept we can also determine the number of bikes that can safely (i.e., without causing the station to become 'inefficient') be picked up or dropped off from a station that is not expected to have inefficiencies. This set of stations is referred to as the buffering stations set S^B—the set of stations that can be visited by the relocation vehicles if they run out or have an excess of bikes. For each station in S^B, the maximum number of bikes that can be drop off z^+_h and the maximum number of bikes that can be picked up z^-_h are defined.

3.5. Vehicle routing

Vehicle routing is done solving the problem P2 for a single vehicle each time it completes the previous tour. We take as an input the results from the redistribution needs model to define a set of candidate stations that can be visited. We select the stations that: (1) are within TT_{max} minutes reach of the current station, (2) where the expected shortage or excess of bikes is larger than a specified threshold, and (3) that are not being served by another vehicle.

Preprocessing the station set and solving the vehicle routing problem for a single vehicle reduces the problem size, allowing the use of traditional solvers to find the solution in real time.

The problem is formulated using an arc and sequence-indexed formulation to keep track of vehicle inventory:

P2:

$$\max\ U = U_I + U_N + U_B \tag{5}$$

$$U_I = \sum_{i\in\mathcal{S}, j\in\mathcal{S}\setminus\{S^B,S^0\},k\in\mathcal{K}} \frac{|u_j|}{C_j}\cdot x^k_{ij} \tag{6}$$

$$U_N = \sum_{i\in\mathcal{S}, j\in\mathcal{S}\setminus\{S^B,S^0\},k\in\mathcal{K}} \beta_j \cdot x^k_{ij} \tag{7}$$

$$\beta_j = \sum_{n\in\mathcal{N}_j}\sum_{m\in T}\frac{y^+_{n,m}+y^-_{n,m}}{C_n} \tag{8}$$

$$U_B = \frac{\gamma}{TT_{max}}\cdot\left(-\sum_{i\in\mathcal{S}, j\in S^B,k\in\mathcal{K}} tt_{i,j}\cdot x^k_{ij} - \sum_{i\in S^B, j\in\mathcal{S},k\in\mathcal{K}} tt_{i,j}\cdot x^k_{ij}\right) \tag{9}$$

$$\text{s.t.} \sum_{ij\in\mathcal{S},k\in\mathcal{K}} (tt_{i,j}+c)\cdot x_{i,j}^{k} \leqslant TT_{max} \tag{1}$$

$$\sum_{j\in\mathcal{S}} x_{S^0,j}^{1} = 1 \tag{2}$$

$$\sum_{i\in\mathcal{S},k\in\mathcal{K}} x_{i,S^0}^{k} = 0 \tag{3}$$

$$\sum_{i\in\mathcal{S},j\in\mathcal{S}} x_{i,j}^{k} \leqslant 1 \quad \forall k\in\mathcal{K} \tag{4}$$

$$\sum_{i\in\mathcal{S}} x_{i,j}^{k} \geqslant \sum_{i\in\mathcal{S}} x_{j,i}^{k+1} \quad \forall j\in\mathcal{S}, k\in\mathcal{K}\setminus\{\tau\} \tag{5}$$

$$\sum_{i\in\mathcal{S},k\in\mathcal{K}} x_{i,j}^{k} \leqslant 1 \quad \forall j\in\mathcal{S} \tag{6}$$

$$\sum_{i\in\mathcal{S},k\in\mathcal{K}} x_{i,i}^{k} = 0 \tag{7}$$

$$D^{k} = D^{k-1} - \sum_{\forall i\in\mathcal{S},j\in\mathcal{S}\setminus\{S^0,S^B\}} u_j\cdot x_{i,j}^{k} - \sum_{h\in S^B} z_h^{k} \quad \forall k\in\mathcal{K} \tag{8}$$

$$-\sum_{i\in\mathcal{S},j\in\mathcal{S}\setminus\{S^0,S^B\}} x_{i,j}^{k}u_j - \sum_{h\in S^B} z_h^{k} + D^{k-1} \leqslant V_{cap} \quad \forall k\in\mathcal{K} \tag{9}$$

$$\sum_{i\in\mathcal{S},j\in\mathcal{S}\setminus\{S^0,S^B\}} x_{i,j}^{k}u_j + \sum_{h\in S^B} z_h^{k} \leqslant D^{k-1} \quad \forall k\in\mathcal{K} \tag{10}$$

$$\sum_{n\in\mathcal{S}} x_{n,h}^{k} - \sum_{n\in\mathcal{S}} x_{h,n}^{k+1} = 0 \quad \forall h\in S^B, k\in\mathcal{K}\setminus\{\tau\} \tag{11}$$

$$\sum_{n\in\mathcal{S}} x_{n,h}^{\tau} = 0 \quad \forall h\in S^B \tag{12}$$

$$z_h^{k} \geqslant z_h^{-}\cdot\sum_{i\in\mathcal{S}} x_{i,h}^{k} \quad \forall h\in S^B, k\in\mathcal{K} \tag{13}$$

$$z_h^{k} \leqslant z_h^{+}\cdot\sum_{i\in\mathcal{S}} x_{i,h}^{k} \quad \forall h\in S^B,\ k\in\mathcal{K} \tag{14}$$

$$D^{k} \geqslant 0 \quad k\in\mathcal{K} \tag{15}$$

$$x_{i,j}^{k}\in\{0,1\} \quad \forall i,j\in\mathcal{S}, k\in\mathcal{K}$$

$$D^{k}\in\mathbb{Z} \quad \forall k\in\mathcal{K}$$

$$z_j\in\mathbb{Z} \quad \forall j\in S^B$$

In the formulation above:

x_{ij}^{k}: Binary variable that indicates if the vehicle traverses the arc i,j at time period k.
z_h^{k}: Integer variable representing the number of bikes picked up or dropped off at buffering stations $h\in S^B$ at time k, and is bounded by z_h^{+} and z_h^{-}, determined in the rebalancing needs model.
D^{k}: Integer variable to keep track of vehicle load at time k.
$tt_{i,j}$: Travel time from i to j.
c: Fixed loading and unloading cost.
C_i: Station capacity.
u_j: Difference between number of bikes drop off and picked up. It is defined as $u_j = y_{j,20}^{+} - y_{j,20}^{-}$, being $y_{j,20}^{+}$ and $y_{j,20}^{-}$ the outputs of the previous model. Note that u_j can be positive or negative, depending on if it is a drop off event or a pick up event, respectively and that a simultaneous pick up or drop off cannot occur at the same station.
τ: Parameter that defines the number of visited stations. Setting τ to 1 makes P2 a purely dispatching problem, where the vehicle will be dispatched to the station that maximizes the utility. Under the current scenario, we set τ to 3, as it is unlikely that a vehicle can visit more than 3 stations in TT_{max} minutes.
TT_{max}: Maximum travel time for a vehicle when is being routed. It is currently set to 20 min.
γ: Dimensionless parameter that accounts for the relative importance of the travel time utility term.

The following sets are defined:

$\mathcal{K}$: Set of time steps $\{1,\ldots,k,\ldots,\tau\}$.
S^{+}: Set of stations with drop off events.
S^{-}: Set of stations with pick up events.
S^B: Set of buffering stations.
S^0: Current station.

S: Set of all stations $\left\{S^+ \cup S^- \cup S^B \cup S^0\right\}$.
$\mathcal{N}_j$: Set of neighboring stations of station j.
T: Set of time windows {20, 40, 60}.

As an objective function U (Eq. (5)) we propose a combination of three elements: (1) the utility gained by visiting a station with a large inefficiency U_I (Eq. (6)), (2) the utility gained by visiting a station with a neighborhood of stations ($\mathcal{N}_j$) that is expected to have inefficiencies in future time steps U_N (Eq. (7)), and (3) in the case that buffering stations need to be visited, minimize the travel time involved in going to those stations, U_B (Eq. (9)).

The utility of visiting a station with a large inefficiency is measured as the ratio between the inefficiency and the station capacity. It has been observed that using a ratio leads to better repositioning results rather than using the inefficiency measured in number of bikes. The second utility term aims to route the current vehicle to a region that is expected to have inefficiencies in the future. To measure it, we have defined the neighborhood of station j, $\mathcal{N}_j$, as all the stations that are within 0.5 miles from the current station j. We then sum over all time windows $T = \{20, 40, 60\}$ the expected inefficiencies $(y^+_{i,t}, y^-_{i,t})$ and we divide them by the capacity of station (C_i).

The last term is introduced as a penalty for visiting buffering stations. If a penalty were not added, vehicles may be routed through buffering stations without any need to. We normalize the travel time by the maximum travel time permitted TT_{max}, which in our problem equals 20 min, and we introduce a coefficient, γ, to change the relative importance of the penalty. For example, setting γ to TT_{max} makes the penalty one order of magnitude larger than the other utility terms, meaning that if there is any other alternative rather than visiting a buffering station, independent of the magnitude of the inefficiency, the buffering station will not be visited. We have tested for the sensitivity of γ in the numerical examples.

Constraint (1) ensures the routing is completed by the next simulation time. Constraints (2) to (7) take care of the routing. Constraints (2) and (3) ensure that the vehicle leaves the depot (its location at the start of the period) and that it will not come back to it in later time steps k. Constraint (4) enforces that only one trip per time step is allowed, constraint (5) is the connectivity constraint and we also enforce that stations cannot be revisited in (6).

Constraints (8)–(10) keep track of vehicle load and make sure that more bikes than the ones available are not dropped off or that vehicle capacity is not exceeded.

Constraint (11) forces buffering stations to be transshipment nodes and constraint (12) does not allow buffering stations to be demand nodes. Constraints (13) and (14) set the bounds on z^k_h if the corresponding buffering station is visited.

Problem P2 can be infeasible, a condition that occurs most frequently when the travel time constraint is violated because the vehicle does not have either enough bikes or empty spaces to serve an inefficiency and has to first visit a buffering station to either pick up or drop off bikes. Under such circumstance, the vehicle remains idle at its current station. The idle time is set to 10 min in current simulation scenarios.

Note that P2 is not formulated as a multi-period problem that uses information from previous models regarding future time steps. In other words, u_j is defined using only the estimated inefficiencies for the first time window. The underlying reasons to do so are because there is a trade off to be made which is due to the accuracy of the predictions and because future information is already accounted for when solving P1 to determine the inefficiencies for the first time window $(y^+_{i,20}, y^-_{i,20})$.

Regarding the accuracy of the predictions, the longer the time horizon, the less accurate are the predictions. In fact, if the predictions were to be 100% accurate, the preferred approach would be to solve a multi-period problem using all the information from previous models. However, as the predictions are not 100% accurate, a heuristic would be needed to make sure that current routes are the best routes possible given current, up to date, information. Instead of proposing such heuristic and the more complex multi-period problem, we have opted by solving P2 more frequently with more recent information. In addition, the current approach may be more appealing and flexible from an operational perspective, as it is likely more responsive to re-routings implemented by the operations center and can better capture the "true" travel times observed from the network in real time, given that travel times can be obtained every time a tour ends (limited to 20 min) rather than after the 3 periods (limited to 60 min), if information from previous models were used.

To demonstrate the impact of the utility terms in the objective function of the routing problem we have built a toy example. This toy example takes a subset of The Hubway system where stations 10, 46 and 27 have an estimated inefficiency of $u_{10} = -2$, $u_{46} = 4$ and $u_{27} = 6$ bikes respectively. Each station with inefficiencies has been assigned a neighboring set, being $\mathcal{N}_{10} = \{9, 33\}, \mathcal{N}_{46} = \{21, 57\}$ and $\mathcal{N}_{27} = \{30\}$ and they all have the same capacity $C_{10} = C_{46} = C_{27} = 20$. The vehicle is initially empty, $D^0 = 0$, to force visiting buffering stations. We also set $\tau = 3$ and $TT_{max} = 20$.

We solve the routing problem P2 under three different scenarios:

Scenario 1: Inefficiencies at later time windows are zero for all stations and $\gamma = TT_{max}$.
Scenario 2: Inefficiencies at later time windows are zero for all stations and $\gamma = 1$.
Scenario 3: Station 57 has an expected inefficiency $y^-_{57,40} = 4$ at time window 40 min and $\gamma = 1$.

Fig. 4 depicts the different routing solutions obtained for each scenario. Note that when $\gamma = TT_{max}$, the vehicle is routed to station 10, where we need to remove 2 bikes, even though the inefficiencies at other stations are larger. Other stations are not visited because the vehicle does not have enough bikes and it does not have time to go through a buffering station. In scenario 2, instead, when setting γ to 1 we reduce the importance of visiting buffering stations and the vehicle is now routed

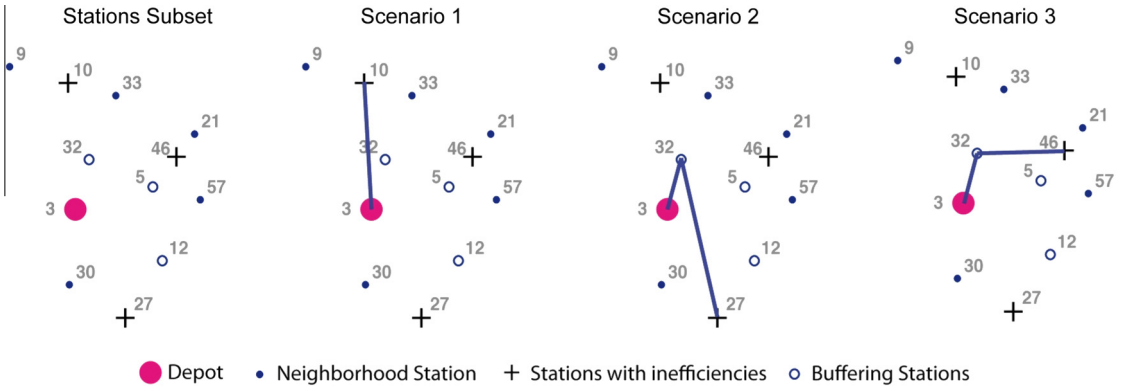

Fig. 4. Vehicle routing toy example.

to station 27, which has the largest inefficiency ($u_{27} = 6$). It first visits the buffering station 32 to pick up the required bikes. Even though it is not apparent from the figures, going to station 32 minimizes the buffering stations travel time.

Finally, we set $y^{-}_{57,40} = 4$ in scenario 3, and now the neighborhood utility term comes into play. Under this case, we visit station 46, as it is likely that in the next iteration, station 57 would need to be visited.

3.6. Performance measures

The performance measures used to compare before and after simulation results are: system performance after (Eq. (10)), system performance before (Eq. (11)), system performance increase (Eq. (12)), relative performance increase of empty events (Eq. (13)), relative performance increase of full events (Eq. (14)), the total number of bikes picked up, the total bikes dropped off, the total distance traveled and the duration of the inefficiencies.

We use subscript A to indicate "after" simulation and B for "before" simulation. *Empty* indicates the number of time periods where a station was empty and *Full* the number of time periods when it was full. D refers to the mean duration of consecutive empty or full events.

$$SysPerf_A = \frac{(\#Observations - (Empty_A + Full_A))}{Total\ Observations} \tag{10}$$

$$SysPerf_B = \frac{(\#Observations - (Empty_B + Full_B))}{\#Observations} \tag{11}$$

$$SysPerf_{INC} = \frac{SysPerf_A - SysPerf_B}{SysPerf_B} \tag{12}$$

$$relEmpty = -\frac{Empty_A - Empty_B}{Empty_B} \tag{13}$$

$$relFull = -\frac{Full_A - Full_B}{Full_B} \tag{14}$$

$$D_{Full} = \frac{D_{Full_A} - D_{Full_B}}{D_{Full_B}} \tag{15}$$

$$D_{Empty} = \frac{D_{Empty_A} - D_{Empty_B}}{D_{Empty_B}} \tag{16}$$

4. Simulation procedure

A MATLAB Graphical User Interface (GUI) has been created that takes the parameters listed in Table 3 as inputs to test for different scenarios and settings. A snapshot of the GUI is shown in Fig. 5.

In the application, travel times are computed using the Google Distance Matrix API. For simplicity, a static OD distance matrix has been computed and a fixed vehicle speed is assumed. In online applications, observed travel times can be queried

Table 3
Simulation Parameters.

Parameter	Description	Default value
technique	Forecasting method	'GBM'
timeWindows	Time windows for predictions in minutes	[20,40,60]
nv	Number of relocation vehicles	2
Vcap	Relocation vehicles' capacity	[20,20]
inLoad	Initial load of each vehicle	[10,10]
Depot	Station serving as a depot for each vehicle (selected at random)	[3,3]
Day Start	Starting time for the vehicle	[7,7]
Day End	Ending time for the vehicle	[21,21]
minTreshold	If the redistribution need is below the threshold, it won't be considered in the dispatching algorithm	1
idelTime	Maximum time a vehicle can remain idling before a new dispatching decision is made in minutes	10
speed	Relocation vehicles speed (mph)	15
loadTime	Loading and unloading time of bikes in minutes per event	5 min
Utility	Combinations of names for the routing utility function	'max-ratio'
Tau	Maximum number of stations visited when routing	3
nStationDistance	Threshold distance to determine neighborhood stations in miles	0.5
inTime	Start time for the simulation	'2012-07-23 07:30'
finTime	End time for the simulation	'2012-07-28 00:00'
Stochastic	Redistribution needs are solved using stochastic approach	1
PlotRoutes	Routes are plotted in the UI	0
TT_{max}	Maximum travel time allowed per time period	20 min
γ	Relative importance of visiting buffering stations utility	TT_{max}

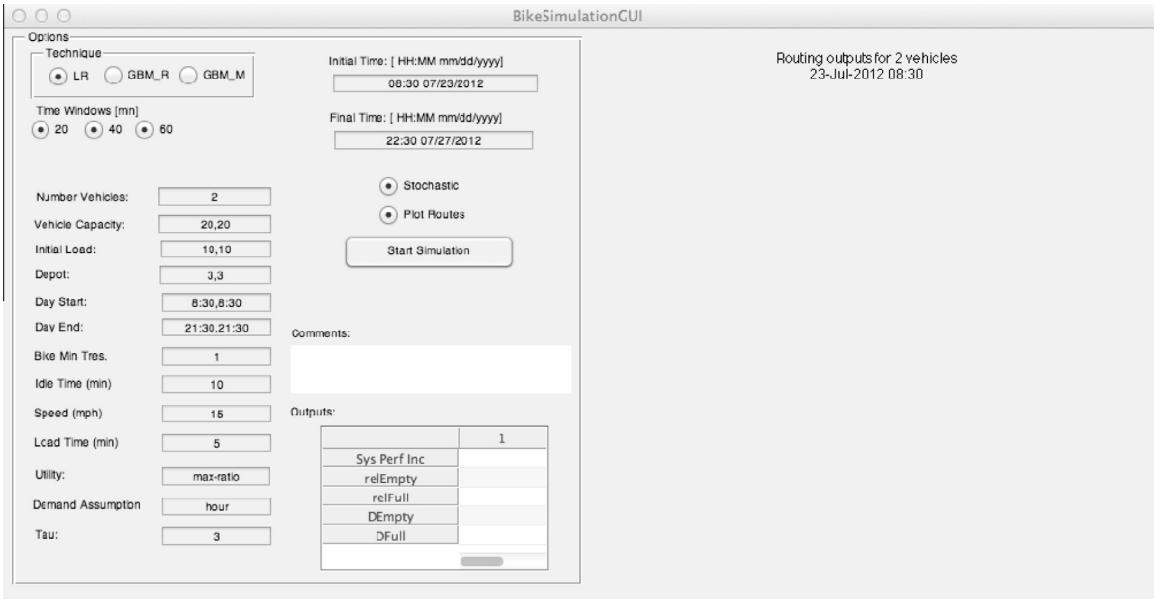

Fig. 5. Software user interface.

in real time and a dynamic OD travel time matrix that would reflect any congestion in the network could replace this static assumption.

Due to the different nature of pick up and drop of events when making predictions, combined with the relatively small $s_{i,t}^{min}$ threshold, the *minThreshold* parameter plays a significant role in the simulation outputs. Setting *minThreshold* to a large value considerably reduces performance in dealing with empty events. After sensitivity testing, we set it to 1.

The simulation flow goes as follows: (1) set simulation parameters, (2) read the current system state and make predictions at 20, 40 and 60 min, (3) solve the rebalancing needs model to find station inefficiencies and select buffering stations, (4) solve the routing problem, (5) update system states, and (6) check for convergence: if convergence is not obtained go to step 2, otherwise dispatch vehicle to the initial depot and compute performance measures.

The convergence check consists of checking if the current simulation time of the vehicle has reached the end of the simulation period. There is also an end of day convergence check if multiple day simulations are considered, where the vehicle is dispatched to the initial depot and initial inputs are reset for the next day to start.

Table 4
PM peak period simulation results.

Performance measure	Unit	GBM	LR	Random
$SysPerf_A$	%	96.33	95.59	88.52
$SysPerf_B$	%	90.16	90.16	90.16
$SysPerf_{INC}$	%	6.84	6.24	−1.82
relEmpty	%	57.41	48.28	−5.56
relFull	%	76.19	76.74	−45.24
D_{Empty}	%	45.79	40.00	27.78
D_{Full}	%	44.44	53.49	27.78
# Bikes picked up	#	72	75	78
# Bikes dropped	#	54	58	59
Total distance traveled	Miles	70.43	64.09	63.93

In order to update the state of the system based on the rebalancing events, an assumption about how to treat the latent demand is needed. This is because we are limited in the application of the models to existing Hubway data (rather than to outputs from a real-time closed loop control system under actual operation., For demonstration purposes, we consider here that one hour after the relocation event has occurred, the demand (or number of bikes) will be the same as that observed in the original Hubway dataset. Note that this assumption may lead to sudden jumps in the number of bikes observed in a station and therefore trigger a relocation event. Note also that in a real-time online scenario this assumption is not needed, as the input to the forecasting model would be the current state of the system.

5. Simulation results and discussion

To demonstrate the potential of the modeling framework to improve current system performance, we first run a simple test case based on the Hubway dataset. The study period goes from 16:30 to 20:30 on July 24th, 2012. We have deliberately selected a time period during the PM peak, as the effects of the static rebalancing are no longer present.

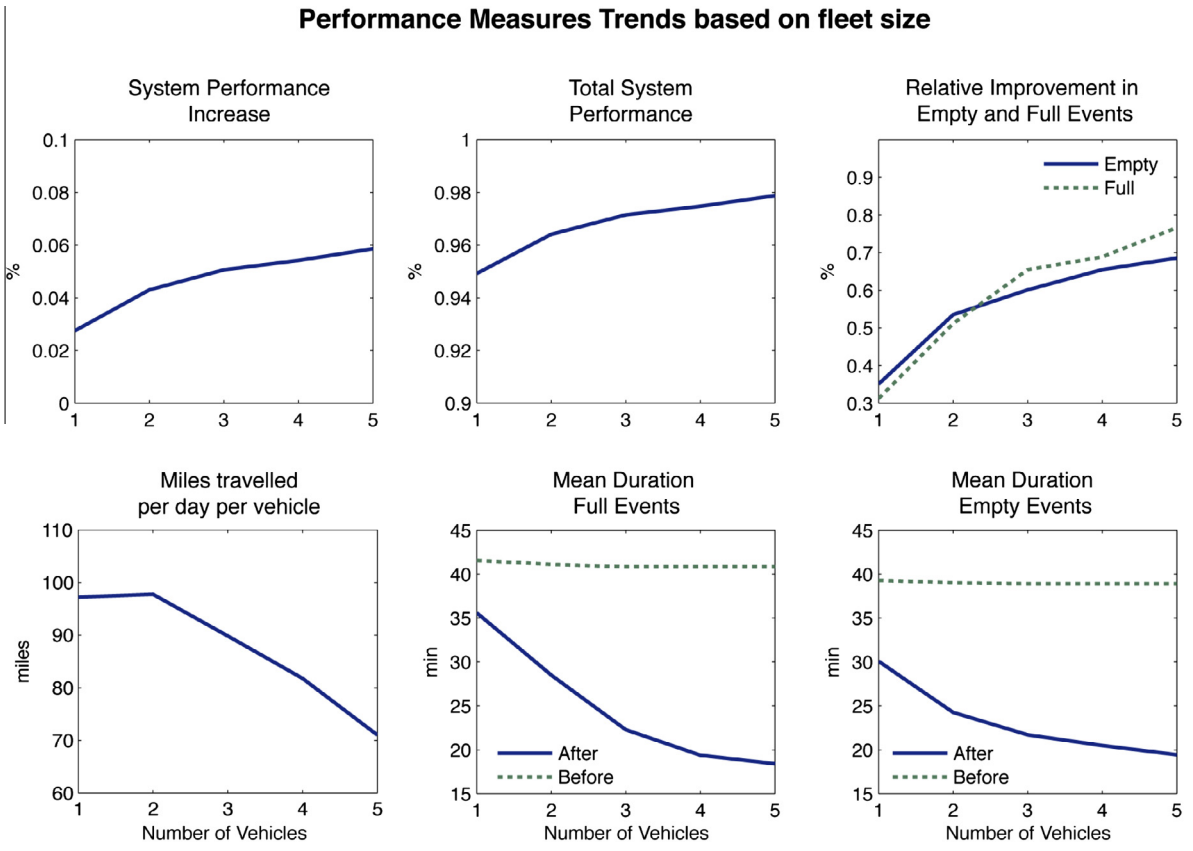

Fig. 6. Performance trends depending on the fleet size.

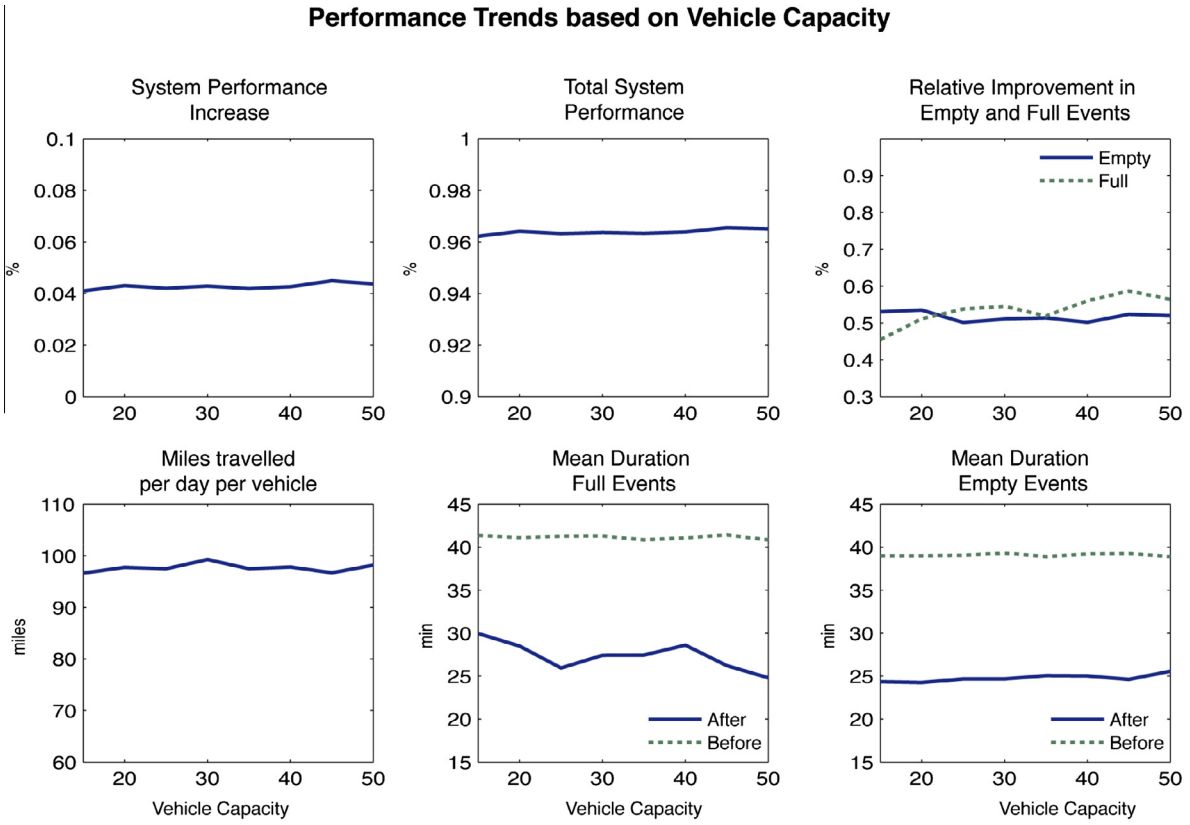

Fig. 7. Performance trends depending on the vehicle capacity.

The settings for this case are the default settings, where 2 vehicles with capacity of 20 bikes and with an initial load of 10 bikes are considered. The simulation is run under three different predictive techniques: GBM, LR, and Random. This latter case tests for the impact of having inaccurate (random) demand predictions in the overall framework; for this case we sample uniform random numbers between 0 and the station capacity as the demand at that station for the next time period.

The results of the simulation are shown in Table 4. From the table it can be observed that the system performance was already at $SysPerf_B$ = 90.16% and after the relocation operation it was boosted $SysPerf_{INC}$ = 6.84% and 6.02% using GBM and LR, respectively.[1] Under the random case, the system performances is degraded, which demonstrates the ability of the predictive model to anticipate events.

If we look at the relative performance regarding empty and full events, using GBM we are able to decrease the number of empty and full events by 57% and 76%, respectively. Note that empty events performance is significantly increased under GBM due to the ability to better predict empty events. Furthermore, the total duration of empty and full events was reduced—by 46% and 44%, respectively—as well as the average duration per empty event—average decrease from 30 min to 17 min per event—and the average duration per full event—average decrease from 39 min to 21 min per event.

A second, more extensive set of simulations was run for the time period from 8:30 to 21:30 for an entire week—from Monday July 23rd to Friday July 27th, 2012—using GBM as the predictive technique. In this set of simulations we tested (separately) for the impacts on the various performance measures of using different fleet size, from 1 vehicle to 5 vehicles, different vehicle capacities, ranging from 15 to 50 bikes (in intervals of 5 units) and different γ parameters (0, 0.5, 1, 2, 5, 10, 20). In all the cases all of the other parameters are set to the default values, and the initial load of the vehicle is set to be half of the vehicle capacity.[2]

Fig. 6 compares different performance measures for variable fleet size. As expected, the larger the fleet size, the better the performance. Using a single vehicle, system performance increases by 2.75% and with five vehicles the performance increase

[1] It should be noted that the current Hubway dataset already contains the rebalancing operations that were made by the operators of the system, meaning that what is being addressed are the remaining inefficiencies that the operator was not able to handle.

[2] Under the scenario in which vehicles can visit buffering stations, the overall results are not sensitive to the initial load parameter, as it only has an effect during the initial simulation steps.

tops 5.86%, reaching an overall system performance of 97.88%. The largest jump is observed in going from 1 to 2 vehicles. A similar trend is observed in terms of empty and full events. We also observe a significant increase in the number of time steps that vehicles remain idle as the fleet size increases, as shown by the average miles traveled per vehicle per day; for the case of 1 or 2 vehicles, they travel on average about 100 miles per day, whereas as fleet increases, average distance traveled drops to 70 miles per day. In all cases the duration of empty and full events is significantly reduced, leveling off at around 20 min for 4 and 5 vehicles. Even with 5 vehicles the system cannot be projected to full performance using the proactive approach, which is not surprising due to the inaccuracies inherent to the predictions.

Fig. 7 shows the trends in performance with gradually increasing the vehicle capacity (and initial load of the two vehicles, which is set at half of the vehicle capacity). As can be observed, all performance measures remain relatively unaffected by the vehicle capacity.

Fig. 8 depicts the results for a varying γ. Similarly to what occurs when changing vehicle capacity, performance measures remain relatively flat, only showing incremental increases when $\gamma > 0$. It can also be noted that when $\gamma = 0$ the miles traveled per day per vehicle are minimum, as well as the mean duration of full events. This behavior can be explained because if the penalty is not imposed on visiting buffering stations, larger inefficiencies that require visiting buffering stations have preference towards smaller inefficiencies that do not require visiting buffering stations. Larger inefficiencies tend to be related with longer and pick up events as the range $C_i - s_{i,t}^{max}$ is often larger than the range $s_{i,t}^{min} - 0$ (Fig. 2). Furthermore, if a vehicle goes through a buffering station, fewer stations with inefficiencies are visited due to the travel time constraint. Overall, because fewer stations with inefficiencies are visited, the system performance is reduced, but miles traveled and mean duration of full events decreases.

A conclusion that could be drawn from these simple results is that, from an operational perspective, a better strategy for increasing system performance (independent of the costs) would be to use more vehicles with smaller capacities rather than fewer vehicles with greater capacity and set γ to a value of 5 as a compromise solution between system performance and travel time. This result is intuitive for this particular application of the methodology to the Hubway dataset since the former strategy affords greater flexibility in addressing what likely are relatively small "residual" system inefficiencies in an operation that has already been tuned to be efficient by the system operators.

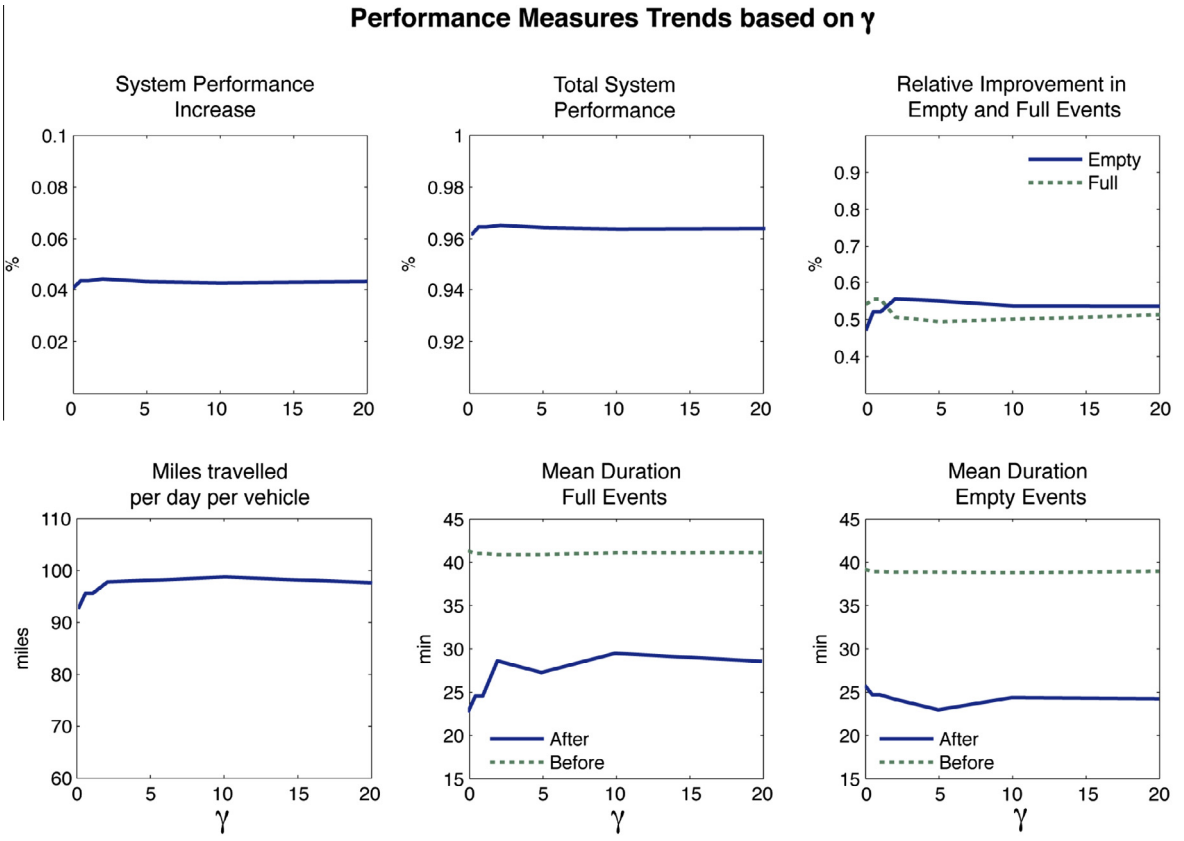

Fig. 8. Performance trends depending on γ.

6. Conclusions and future work

In this paper we have proposed a methodological framework to solve the bikesharing repositioning problem based on four core models: (1) a demand forecasting model at the station level, (2) a station inventory model, (3) a redistribution needs model, and (4) a vehicle-routing model. The novelty of this approach is that it is proactive instead of reactive, as the bike redistribution occurs before inefficiencies are observed, increasing system performance and, potentially, customer satisfaction, and uses the outputs of a machine learning technique to decompose the inventory and the routing problem. The decomposition approach proposed: (1) makes the problem scalable to large bikesharing systems, (2) allows for real time implementation, making routing decisions every time a vehicle completes a limited tour, (3) is responsive to operator inputs, and (4) can accommodate user-specific models. In addition, the underlying models based on historical data are self-adaptive, as they are constantly being retrained using the most recent data available.

Simulation results based on data associated with the Hubway Bikesharing system show that significant improvements to the overall system performance could have been made (over and above that being achieved under current operation) using the proposed modeling approach—achieving improvements of nearly 7% in the afternoon peak. As expected, performance measures are better when the predictive model makes better predictions. More comprehensive tests using a full week of data demonstrate how the methodology could be used to evaluate such decisions as fleet size and vehicle capacity. To test the full potential of the framework, tests using real-time data under closed-loop control should be used (i.e., data that are not already the outcomes of operational decisions) and compared to the current rebalancing decisions made by the operator. However, access to such data was not possible.

Having outlined the fundamental philosophy behind, and demonstrated the applicability of, the framework, further research should focus on: (1) fine tuning each model, (2) incorporating a user-based relocation model, (3) building it as a web application tool for bikesharing operators to use, and (4) testing its transferability to other bikesharing systems. In terms of improving the various models, the forecasting model should incorporate more independent variables such as large events (e.g., basketball games, concerts, etc.), transit arrivals and departures in areas nearby bikesharing stations or real time distribution of people in the city, which could be gathered from cellphone towers or social media data. For the routing model, different objective functions can be proposed that incorporate the concept of maximizing a cumulative reward instead of the immediate benefit gained by routing a vehicle to a station. Additionally, some of the input parameters, such as τ, could be made self-adaptive to better handle idling situations.

Acknowledgements

The authors gratefully acknowledge the The Hubway Bikesharing system for making data publically available. This research was supported, in part, by grants from the University of California ITS Multi-Campus Research Program and Initiative on Sustainable Transportation, the *Balsells-Generalitat de Catalunya Fellowship* and the *Fundación Caja Madrid Fellowship*. Their support is gratefully acknowledged.

References

Ahmed, M.M., Abdel-Aty, M., 2013. Application of a stochastic gradient boosting (SGB) technique to enhance the reliability of real-time risk assessment using AVI and RTMS data. In: Presented at the Transportation Research Board 92nd Annual Meeting.

Barth, M., Todd, M., 1999. Simulation model performance analysis of a multiple station shared vehicle system. Transp. Res. C 7, 237–259.

Borgnat, P., Abry, P., Flandrin, P., Robardet, C., Rouquier, J.-B., Fleury, E., 2011. Shared bicycles in a city: a signal processing and data analysis perspective. Adv. Complex Syst. 14, 415–438. http://dx.doi.org/10.1142/S0219525911002950.

Caggiani, L., Ottomanelli, M., 2012. A modular soft computing based method for vehicles repositioning in bike-sharing systems. Proc. – Social Behav. Sci. 54, 675–684. http://dx.doi.org/10.1016/j.sbspro.2012.09.785.

Caggiani, L., Ottomanelli, M., 2013. A dynamic simulation based model for optimal fleet repositioning in bike-sharing systems. Proc. – Social Behav. Sci. 87, 203–210.

Chemla, D., Meunier, F., Wolfler Calvo, R., 2012. Bike sharing systems: solving the static rebalancing problem. Discr. Optim. 10, 120–146. http://dx.doi.org/10.1016/j.disopt.2012.11.005.

Cheu, R., Xu, J., Kek, A., Lim, W.P., Chen, W.L., 2006. Forecasting shared-use vehicle trips with neural networks and support vector machines. Transp. Res. Rec. 40–46. http://dx.doi.org/10.3141/1986-13.

City of Irvine, 2014. Irvine Transportation Network [WWW Document]. cityofirvine.org. URL: <http://www.cityofirvine.org/cityhall/pw/itn_new/transit/routes/irvine.asp> (accessed 6.14).

Contardo, C., Morency, C., Rousseau, L.M., 2012. Balancing a Dynamic Public Bike-Sharing System. CIRRELT.

Ferrucci, F., 2013. Pro-active Dynamic Vehicle Routing. Real-Time Control and Request-Forecasting Approaches to Improve Customer Service. Physica-Verlag. http://dx.doi.org/10.1007/978-3-642-33472-6.

Fricker, C., Gast, N., Mohamed, H., 2012. Mean field analysis for inhomogeneous bike sharing systems. DMTCS Proc., 365–376.

Friedman, J.H., 2001. Greedy function approximation: a gradient boosting machine. Ann. Stat. 29, 1189–1232.

Friedman, J.H., 2002. Stochastic gradient boosting. Comput. Stat. Data Anal. 38, 367–378.

Froehlich, J., Neumann, J., Oliver, N., 2009. Sensing and predicting the pulse of the city through shared bicycling. In: International Joint Conference on Artificial Intelligence (IJCAI-09).

Gebhart, K., Noland, R.B., 2013. The impact of weather conditions on capital bikeshare trips. In: Presented at the Transportation Research Board 92nd Annual Meeting.

Henderson, J., Fishman, A., 2013. Divvy: Helping Chicago's New Bike Share Find Its Balance [WWW Document]. Data Science for Social Good. URL: <http://dssg.io/2013/08/09/divvy-helping-chicagos-new-bike-share.html> (accessed 1.14).

Hernandez-Perez, H., Salazar-Gonzalez, J.-J., 2004. Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. Transp. Sci. 38, 245–255. http://dx.doi.org/10.1287/trsc.1030.0086.

Hubway, 2011. Hubway [WWW Document]. The Hubway. URL: <http://www.thehubway.com/> (accessed 7.12).

Hubway, 2012. Hubway Data Visualization Challenge [WWW Document]. The Hubway. URL: <http://hubwaydatachallenge.org/> (accessed 5.13).
Kaltenbrunner, A., Meza, R., Grivolla, J., Codina, J., Banchs, R., 2010. Urban cycles and mobility patterns: exploring and predicting trends in a bicycle-based public transport system. Pervasive Mobile Comput. 6, 455–466. http://dx.doi.org/10.1016/j.pmcj.2010.07.002.
Kek, A.G.H., Cheu, R.L., Xu, J., 2005. Trip forecasting models for intelligent community vehicle systems. In: Presented at the Transportation Research Board 84th Annual Meeting, Washington, pp. 1–17.
Kek, A.G.H., Cheu, R.L., Meng, Q., Fung, C.H., 2009. A decision support system for vehicle relocation operations in carsharing systems. Transp. Res. Part E: Logist. Transp. Rev. 45, 149–158. http://dx.doi.org/10.1016/j.tre.2008.02.008.
Lu, C.-C., 2013. Robust multi-period fleet allocation models for bike-sharing systems. Netw. Spat. Econ., 1–22. http://dx.doi.org/10.1007/s11067-013-9203-9.
Meddin, R., DeMaio, P., 2007. The Bike-Sharing World Map [WWW Document]. URL: <http://www.bikesharingworld.com> (accessed 5.8.13).
Nair, R., Miller-Hooks, E., Hampshire, R.C., Bušić, A., 2013. Large-scale vehicle sharing systems: analysis of Vélib'. Int. J. Sustain. Transp. 7, 85–106, 10.1080/15568318.2012.660115.
National Climatic Data Center, 2012. Quality Controlled Local Climatological Data (QCLCD) [WWW Document]. NOAA. URL: <http://www.ncdc.noaa.gov/land-based-station-data/quality-controlled-local-climatological-data-qclcd> (accessed 9.12).
Núñez, A.A., Sáez, D.A., Cortes, C.E., 2013. Hybrid Predictive Control for Dynamic Transport Problems. Springer.
Pfrommer, J., Warrington, J., Schildbach, G., Morari, M., 2013. Dynamic vehicle redistribution and online price incentives in shared mobility systems. arXiv.
Rainer-Harbach, M., Papazek, P., Hu, B., Raidl, G.R., 2013. Balancing bicycle sharing systems: a variable neighborhood search approach. In: Evolutionary Computation in Combinatorial Optimization, Lecture Notes in Computer Science. Springer, Berlin Heidelberg, Berlin, Heidelberg, pp. 121–132. doi:http://dx.doi.org/10.1007/978-3-642-37198-1_11.
Raviv, T., Kolka, O., 2013. Optimal inventory management of a bike-sharing station. IIE Trans. 45, 1077–1093. http://dx.doi.org/10.1080/0740817X.2013.770186.
Raviv, T., Tzur, M., Forma, I.A., 2013. Static repositioning in a bike-sharing system: models and solution approaches. Euro J. Transp. Logist. 2, 187–229. http://dx.doi.org/10.1007/s13676-012-0017-6.
Regué, R., Recker, W., 2014. Using gradient boosting machines to predict bikesharing station states. In: Presented at the Transportation Research Board 93rd Annual Meeting, Washington.
Ridgeway, G., 2012. GBM R Package [WWW Document]. cran.r-project.org. URL: <http://cran.r-project.org/web/packages/gbm/gbm.pdf> (accessed 11.12).
Schuijbroek, J., Hampshire, R., van Hoeve, W.-J., 2013. Inventory Rebalancing and Vehicle Routing in Bike Sharing Systems. Tepper School of Business.
Shaheen, S.A., Martin, E., Cohen, A., 2012. Public Bikesharing in North America: Early Operator and User Understanding. San José.
Time and Date AS, 1995. timeanddate.com [WWW Document]. timeanddate.com. URL: <http://www.timeanddate.com/> (accessed 11.12).
Wu, T., Xie, K., Xinpin, D., Song, G., 2012. A online boosting approach for traffic flow forecasting under abnormal conditions. In: Presented at the 2012 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), IEEE, pp. 2555–2559. doi:http://dx.doi.org/10.1109/FSKD.2012.6234335.