Online Repositioning in Bike Sharing Systems

Meghna Lowalekar, Pradeep Varakantham, Supriyo Ghosh, Sanjay Dominik Jena[†], Patrick Jaillet[‡]
School of Information Systems, Singapore Management University
[†]Département de management et technologie, École des sciences de la gestion (ESG) UQAM
[‡]Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, USA
meghnal.2015@phdis.smu.edu.sg, pradeepv@smu.edu.sg, supriyog.2013@phdis.smu.edu.sg,sanjay.jena@cirrelt.ca,jaillet@mit.edu

Abstract

Due to increased traffic congestion and carbon emissions, Bike Sharing Systems (BSSs) are adopted in various cities for short distance travels, specifically for last mile transportation. The success of a bike sharing system depends on its ability to have bikes available at the "right" base stations at the "right" times. Typically, carrier vehicles are used to perform repositioning of bikes between stations so as to satisfy customer requests. Owing to the uncertainty in customer demand and day-long repositioning, the problem of having bikes available at the right base stations at the right times is a challenging one. In this paper, we propose a multi-stage stochastic formulation, to consider expected future demand over a set of scenarios to find an efficient repositioning strategy for bike sharing systems. Furthermore, we provide a Lagrangian decomposition approach (that decouples the global problem into routing and repositioning slaves and employs a novel DP approach to efficiently solve routing slave) and a greedy online anticipatory heuristic to solve large scale problems effectively and efficiently. Finally, in our experimental results, we demonstrate significant reduction in lost demand provided by our techniques on real world datasets from two bike sharing companies in comparison to existing benchmark approaches.

1 Introduction

Bike Sharing systems (BSSs) are rapidly becoming a dominant mode of transportation for short distance trips in various cities. The use of bicycle as a mode of transportation helps in decreasing traffic congestion and carbon emissions that has been increasing due to excess private vehicle usage. Some of the popular bike sharing systems include Capital Bikeshare in Washington DC, Hubway in Boston, Bixi in Montreal, and Vélib in Paris. A bike sharing system typically has a set of base stations that are strategically placed throughout a city and each station has a fixed number of docks. At the start of the day, each station is stocked with a predetermined number of bikes. Customers can pick and drop bikes from any station and are charged depending on the hiring duration.

As the customers move according to their own needs, during the course of the day there will be starvation at some stations and congestion at some stations. To rebalance the availability of bikes, companies employ carrier vehicles to move bikes between stations. There has been extensive research on static repositioning approaches (Schuijbroek, Hampshire, and van Hoeve 2017; Chemla, Meunier, and Calvo 2013; Raviv, Tzur, and Forma 2013). These approaches perform repositioning only at the beginning of the day, when the movements of bikes by the customers are negligible, to ensure desired distribution of bikes across different stations. But they do not consider stations getting imbalanced during the day. Hence there is a need to focus on dynamic repositioning of bikes (Contardo, Morency, and Rousseau 2012; Ghosh et al. 2015; 2017) during the day.

Shu et al. (2013) provide an optimization model for dynamic repositioning of bikes, but they do not consider routing of carrier vehicles. Ghosh et al. (2015; 2017) consider the problem of dynamic repositioning of bikes along with the problem of finding the routing policy for carrier vehicles. They provide an offline policy generation approach based on mean demand computed from the historical data. To overcome the inherent complexity of the problem, they propose a decomposition and abstraction based approach. While the offline policy provides significant improvement over static repositioning approaches, it is unable to consider the changing demand scenarios in real time. Recently Ghosh and Varakantham (2017) propose a novel dynamic repositioning approach by combining optimization and mechanism design, to reduce carbon footprint using bike trailers. Ghosh, Trick, and Varakantham (2016) provide an online approach to compute dynamic repositioning and routing policy. The focus of their work is on providing a robust approach which optimizes for the worst case scenario. While robust policies provide guarantees for the adversarial inputs, they do not work well when demand follows an expected distribution and hence, we do not provide an experimental comparison.

In this work, we provide an online approach to compute dynamic repositioning and routing policy using demand samples from historical data. Due to its online nature, this approach can adapt and react to the changing demand patterns in real time. Two Stage stochastic models have been successfully used in various related domains such as online taxi matching (Lowalekar, Varakantham, and Jaillet 2016), online multi resource routing problem (Zhang, Smilowitz, and Erera 2011), etc. We specifically provide a multi-stage stochastic optimization formulation, which considers samples of future customer demand (typically obtained from historical data) for finding repositioning and routing policy. Given the large scale nature of the problem, we provide a Lagrangian decomposition approach (that decouples the global problem into routing and repositioning slaves and employs a novel DP approach to efficiently solve routing slave) and

a greedy online anticipatory heuristic approach to improve scalability. Finally, we provide an exhaustive comparison of our approaches against multiple benchmarks on two real world datasets.

2 Model

In this section, we provide a generic model for representing the problem of repositioning and routing while considering demand uncertainty in bike sharing systems. We extend the expected demand based model of Dynamic Repositioning and Routing Problem (DRRP) provided by Ghosh et al. (2015). Here, we provide the Dynamic Repositioning and Routing Problem with Demand Uncertainty **(DRRPDU)**, that considers multiple samples of demand at different stations and timesteps.

DRRPDU is formally defined using the following tuple:

$$\left\langle \mathcal{S}, \mathcal{V}, F, C^{\#}, C^{*}, d, \xi, \sigma, \mathcal{X}, \Delta, Q \right\rangle$$

- $\mathcal{S}$ denotes the set of stations.
- $\mathcal{V}$ denotes the set of carrier vehicles used for repositioning of bikes.
- F denotes samples of customer requests for the future timesteps with $F^{t,k}$ denoting the set of customer requests in demand sample k at decision epoch t. We use $F_{s,s'}^{t,t',k}$ to denote the number of customer requests between stations s and s' in sample k which start at decision epoch t and end at decision epoch t'. $F_s^{t,k}$ denotes the total number of customer requests originating at station s and decision epoch t in sample k.
- $C^{\#}$ denotes the capacity of stations with $C_s^{\#}$ denoting capacity of station s.
- C^{*} denotes the capacity of carrier vehicles with C_v^{*} denoting capacity of vehicle v.
- d denotes the initial distribution of bikes at stations with d_s denoting the initial number of bikes at station s.
- ξ denotes the initial distribution of bikes in vehicle with ξ_v denoting the initial number of bikes in vehicle v.
- σ denotes the distribution of carrier vehicles at stations. $\sigma_v^t(s)$ is set to 1, if vehicle v starts from station s at decision epoch t and is 0 otherwise.
- $\mathcal{X}_s^t$ denotes the additional number of bikes which will become available at station s at decision epoch t[1].
- Δ denotes the decision epoch duration in minutes.
- Q denotes the lookahead period, i.e., the number of decision epochs for which request samples are considered. $Q * \Delta$ gives the lookahead period duration in minutes.

The goal in DRRPDU is to minimize the expected lost demand over multiple samples of demand scenarios over the entire time horizon $Q*\Delta$. It should be noted that all elements of the DRRPDU tuple except Q and Δ can be populated directly from bike sharing datasets. Q and Δ are typically decided by management based on observing movement of bikes.

[1]This could be due to the bikes hired in previous decision epochs which are expected to return at decision epoch t

Variable	Description
$y_{s,v}^{+,0}$	Number of bikes picked up from station s by vehicle v at current decision epoch.
$y_{s,v}^{-,0}$	Number of bikes dropped at station s by vehicle v at current decision epoch.
$y_{s,v}^{+,t,k}$	Number of bikes picked up from station s by vehicle v at decision epoch t in sample k
$y_{s,v}^{-,t,k}$	Number of bikes dropped at station s by vehicle v at decision epoch t in sample k
$h_s^{t,k}$	Bikes hired in sample k at station s at epoch t
$g_s^{t,k}$	Bikes returned in sample k at station s at epoch t
$L_s^{t,k}$	Lost demand in sample k at station s at epoch t
$d_s^{t,k}$	Number of bikes present at station s at epoch t in sample k
u_v^0	Number of bikes present in vehicle v at current decision epoch.
$u_v^{t,k}$	Number of bikes present in vehicle v at epoch t in sample k
$z_{s,v}^{t,k}$	Indicates whether vehicle v is present at station s at decision epoch t in sample k
$m_{s,v}^{0,t}$	Indicates whether vehicle v moves towards s at current decision epoch and reaches s at t
$m_{s,v}^{t,t',k}$	Indicates whether vehicle v moves towards s at t and reaches s at decision epoch t' in sample k

Table 1: Notation

MSS ():

$$\min \quad \frac{1}{|F|} \sum_{t=0}^{Q-1} \sum_{k} \sum_{s} L_s^{t,k} \tag{1}$$

$$s.t. \quad L_s^{t,k} \geq F_s^{t,k} - h_s^{t,k} ::: \forall s,t,k \tag{2}$$

$$y_{s,v}^{+,t,k} + y_{s,v}^{-,t,k} \leq C_v^{*} * z_{s,v}^{t,k} ::: \forall s,v,t > 0,k \tag{3}$$

$$z_{s,v}^{t,k}, m_{s,v}^{0,t}, m_{s,v}^{t,t',k} \in \{0,1\} \tag{4}$$

$$Constraints \ (5) - (22)$$

Table 2: MSS Formulation

3 Multi-Stage Stochastic Optimization

We now formulate DRRPDU as a mixed integer linear optimization formulation. Unlike most current online repositioning approaches that are typically myopic (Pfrommer et al. 2014), this multi-stage stochastic (MSS) formulation considers future demand scenarios while optimizing repositioning of bikes. The variables used in MSS are described in Table 1. Table 2 presents the MSS formulation. Here are the key constraints that are not specific to only one of repositioning and routing:

***Lost demand*:** Constraints (2) along with the objective ensure that lost demand at any station, at any decision epoch is the difference between demand and hired bikes.

***Validity of Pickup/Dropoff*:** Constraints (3) ensure that a vehicle v picks or drops bikes from any station s at any decision epoch t in any sample k if and only if the station s is visited by vehicle at decision epoch t in sample k.

Rest of the constraints are specific to either **Repositioning**

Repositioning:

$$h_s^{t,k} \leq F_s^{t,k} ::: \forall s,t,k \quad (5)$$

$$h_s^{0,k} \leq d_s - \sum_v y_{s,v}^{+,0} + \sum_v y_{s,v}^{-,0} ::: \forall s,k \quad (6)$$

$$h_s^{t,k} \leq d_s^{t,k} - \sum_v y_{s,v}^{+,t,k} + \sum_v y_{s,v}^{-,t,k} ::: \forall s,t>0,k \quad (7)$$

$$g_s^{t,k} \leq \sum_{t'=0}^{t-1} \sum_{s'} h_{s'}^{t',k} * \frac{F_{s',s}^{t',t,k}}{F_{s'}^{t',k}} + \mathcal{X}_s^t ::: \forall s,t>0,k \quad (8)$$

$$d_s^{t,k} = d_s^{t-1,k} - \sum_v y_{s,v}^{+,t-1,k} + \sum_v y_{s,v}^{-,t-1,k} - h_s^{t-1,k} + g_s^{t,k} ::: \forall s,t>1,k \quad (9)$$

$$d_s^{1,k} = d_s - \sum_v y_{s,v}^{+,0} + \sum_v y_{s,v}^{-,0} - h_s^{0,k} + g_s^{1,k} ::: \forall s,k \quad (10)$$

$$d_s^{t,k} \leq C_s^{\#} ::: \forall s,t,k \quad (11)$$

$$\sum_v y_{s,v}^{+,0} \leq d_s^0; \sum_v y_{s,v}^{-,0} \leq C_s^{\#} - d_s^0 ::: \forall s \quad (12)$$

$$\sum_v y_{s,v}^{+,t,k} \leq d_s^{t,k}; \sum_v y_{s,v}^{-,t,k} \leq C_s^{\#} - d_s^{t,k} ::: \forall s,k,t>0 \quad (13)$$

$$u_v^{t,k} = u_v^{t-1,k} + \sum_s y_{s,v}^{+,t,k} - \sum_s y_{s,v}^{-,t,k} ::: \forall v,k,t>0 \quad (14)$$

$$u_v^{0,k} = \xi_v + \sum_s y_{s,v}^{+,0} - \sum_s y_{s,v}^{-,0} ::: \forall v,k \quad (15)$$

$$u_v^{t,k} \leq C_v^* ::: \forall v,k,t \quad (16)$$

$$y_{s,v}^{+,0} + y_{s,v}^{-,0} \leq C_v^* * \sigma_v^0(s) ::: \forall s,v \quad (17)$$

Table 3: Repositioning Constraints

or **Routing**. Table 3 presents the constraints related to the repositioning problem:

Hired bikes : Constraints (5)-(7) ensure that at any station at any decision epoch, the number of hired bikes is the minimum of available bikes at station and demand at station.

Returned bikes: Constraint (8) compute the number of returned bikes at any station and decision epoch as the sum of bikes returned due to hiring at previous decision epochs (within formulation) and bikes which will be returned due to previous hires.

Bike availability: Constraints (9)-(11) ensure that the number of available bikes at any station at any decision epoch is less than the capacity of that station. At any station, at any decision epoch, the number of bikes available is calculated by considering hired bikes at previous decision epoch, bikes picked up/dropped by vehicles in previous decision epoch and bikes returned by customers in current decision epoch.

Station capacity: Constraints (12)-(13) ensure that the number of bikes picked up from any station is less than the number of available bikes and the number of bikes dropped at any station is less than the number of available free slots.

Vehicle capacity: The number of bikes present in any vehicle v is calculated by considering bikes picked up and dropped by vehicle in current decision epoch and the number of bikes already present in the vehicle. Constraints (14)-(16) ensure that at any decision epoch, in any sample, the number of bikes in any vehicle v is less than capacity of vehicle.

Routing:

$$z_{s,v}^{t,k} = \sum_{t'=1}^{t-1} m_{s,v}^{t',t,k} + m_{s,v}^{0,t} + \sigma_v^t(s) ::: \forall s,v,t>0,k \quad (18)$$

$$m_{s,v}^{t,t',k} \leq \sum_{s'} z_{s',v}^{t,k} * \delta_{s',s,v}^{t,t'} ::: \forall s,v,t,k \quad (19)$$

$$m_{s,v}^{0,t} \leq \sum_{s'} \sigma_v^0(s') * \delta_{s',s,v}^{0,t} ::: \forall s,v,t \quad (20)$$

$$\sum_{t'=t+1}^{Q-1} \sum_s m_{s,v}^{t,t',k} = \sum_{s'} z_{s',v}^{t,k} ::: \forall v,t,k \quad (21)$$

$$\sum_{t'=1}^{Q-1} \sum_s m_{s,v}^{0,t'} = \sum_{s'} \sigma_v^0(s') ::: \forall v \quad (22)$$

Table 4: Routing Constraints

Table 4 contains the constraints related to routing problem. These constraints ensure that there exists a valid path between vehicle positions at different decision epochs. $m_{s,v}^{0,t}$ variables are used to ensure that at current decision epoch, vehicle moves towards same station across all samples. All the constraints in this case ensure that ***movement of vehicles*** are valid.

Constraint (18) ensure that a vehicle v is present at station s, at decision epoch t, in sample k, if and only if , either it is reaching station s at decision epoch t due to previous assignments (given by σ_v^t values) or if it is going to reach station s at decision epoch t due to assignments which are part of current formulation. Constraints (19)-(20) ensure that a vehicle v starts moving towards station s' at decision epoch t and reaches s' at decision epoch t', if at decision epoch t it was present at some station s, such that the distance between s and s' can be covered in time $t'-t$. We use binary constants $\delta_{s,s',v}^{t,t'}$ to indicate if vehicle v starting at station s at decision epoch t reaches station s' exactly at decision epoch t' or not. We assume a fixed travel time between stations based on average speed of vehicle so these binary constants can be calculated beforehand. Constraints (21)-(22) ensure that at any decision epoch, if a vehicle is present at some station, then it will start moving towards exactly one of the stations. A movement between the same station indicates vehicle is staying at the same station.

We solve the MSS optimization formulation online at each decision epoch to compute the repositioning and routing strategy. At each decision epoch, distribution of bikes at stations and vehicle positions are updated based on actual realized customer requests and repositioning strategy executed by vehicles.

4 Lagrangian Dual Decomposition

Since we have to make decisions online, the solution has to be generated quickly. With increasing number of stations and increasing number of samples, the number of variables and constraints increases in MSS which makes it difficult to

solve quickly. Decomposition across samples does not help in reducing computation time, as for problems with large number of stations, even for a single sample MSS takes a long time (thousands of seconds). Therefore, we extend the Lagrangian Dual Decomposition (LDD) (Fisher 1985) method proposed by Ghosh et al. (2015) for solving offline repositioning and routing problem in bike sharing. Our key contributions within this LDD approach when applied to DRRPDU are two fold:(i) We update LDD to account for multiple demand samples; (ii) We significantly improve the computational complexity of solving the routing problem by using dynamic programming.

In our MSS formulation, as we can see in Table 2, only constraints (3) link the routing and repositioning variables across samples. Therefore, we dualize constraint (3) using price variables $\alpha_{s,v}^{t,k}$ and obtain Lagrangian as follows:

$$\mathcal{L}(\alpha) = \min \Big[\frac{1}{|F|}\sum_{t=0}^{Q-1}\sum_{k}\sum_{s} L_s^{t,k} + \sum_{t=1}^{Q-1}\sum_{v}\sum_{k}\sum_{s}\alpha_{s,v}^{t,k} * (y_{s,v}^{+,t,k} + y_{s,v}^{-,t,k} - C_v^* * z_{s,v}^{t,k})\Big] \quad (23)$$

$$= \min \Big[\frac{1}{|F|}\sum_{t=0}^{Q-1}\sum_{k}\sum_{s} L_s^{t,k} + \sum_{v}\sum_{t=1}^{Q-1}\sum_{k}\sum_{s}\alpha_{s,v}^{t,k} * (y_{s,v}^{+,t,k} + y_{s,v}^{-,t,k})\Big] - \min\Big[\sum_{t=1}^{Q-1}\sum_{v}\sum_{k}\sum_{s}(C_v^* * \alpha_{s,v}^{t,k} * z_{s,v}^{t,k})\Big] \quad (24)$$

In equation (24) the first two terms correspond to the repositioning problem and last term corresponds to the routing problem. Therefore, we have a decomposition of dual problem into repositioning and routing slaves. The repositioning slave minimizes the first two terms of equation (24) subject to constraints (5)-(16). The routing problem minimizes the last term of equation (24) subject to constraints in Table 4.

As the task of routing constraints is to ensure the presence of a valid path between vehicle positions at different timesteps and the objective of routing slave is to minimize the weights of visited station timestep pairs, instead of solving it as an integer optimization problem, we can also solve it using dynamic programming to significantly improve efficiency. We can observe in the routing constraints (Table 4) that vehicles are independent of each other. Therefore, if we have a single sample, for each vehicle the routing problem can be solved separately. The routing problem can be viewed as node weighted shorted path problem with each station at each decision epoch as graph node and node weights as $-1 * C_v^* * \alpha_{s,v}^{t,k}$.

In case of multiple samples, once the $m_{s,v}^{0,t}$ variables are fixed, samples are independent of each other. Therefore, for each vehicle and each sample we can still solve using dynamic programming and at t=0, instead of taking minimum for individual sample, we take the minimum of sum of weights for all samples. Algorithm 1 provides the detailed steps. Steps 3-9 identify the starting station and timestep for the vehicle. We then use $w_{s,v}$ variables to store the weight at the vertex and $a_{s,v}$ variables to store the path. Steps 15-18 are the key dynamic programming steps that update the weight at the vertices using backward induction. Steps 29-37 update the variables of the optimization formulation using the stored path in $a_{s,v}$ variables.

To obtain the solution to MSS,we optimize $\max_\alpha \mathcal{L}_\alpha$. Given an α, the dual value corresponding to the MSS is obtained by adding the solution from both slaves. The master optimization problem is solved iteratively using subgradient descent on price variables α as described in Algorithm 2. Convergence in the process is detected when difference between primal solution (defined as p in Algorithm 2) and dual solution (defined as sum of objective values of repositioning and routing slaves) is lesser than a small predetermined value (ϵ).

Algorithm 1 SolveRouting(α)

1: $obj = 0$
2: **for** $v \in \mathcal{V}$ **do**
3: $startts = -1$
4: **for** t=0 to Q-1 **do**
5: **if** $\sigma_v^t(s) == 1$ **then**
6: $startts \leftarrow t$
7: **if** $t > 0$ **then**
8: **for** $k = 1$ to $|F|$ **do**
9: $z_{s,v}^{t,k} \leftarrow 1$
10: $t_1 = startts > 0 ? startts : startts + 1$
11: **for** $k = 1$ to $|F|$ **do**
12: **for** $t = Q-1$ to t_1 **do**
13: **for** $s \in \mathcal{S}$ **do**
14: **if** t==Q-1 **then**
15: $w_{s,v}^{t,k} \leftarrow -1 * C_v^* * \alpha_{s,v}^{t,k}$
16: **else**
17: $w_{s,v}^{t,k} \leftarrow \min_{s',t'}((w_{s',v}^{t',k} - C_v^* * \alpha_{s,v}^{t,k}) * \delta_{s,s',v}^{t,t'})$
18: $a_{s,v}^{t,k} \leftarrow \operatorname{argmin}_{s',t'}((w_{s',v}^{t',k} - C_v^* * \alpha_{s,v}^{t,k}) * \delta_{s,s',v}^{t,t'})$
19: **if** $startts == 0$ **then**
20: $w_{s,v}^0 \leftarrow \min_{s',t'}(\sum_k (w_{s',v}^{t',k}) * \delta_{s,s',v}^{0,t'})$
21: $a_{s,v}^0 \leftarrow \operatorname{argmin}_{s',t'}(\sum_k (w_{s',v}^{t',k}) * \delta_{s,s',v}^{0,t'})$
22: $obj \leftarrow w_{s,v}^0$
23: **else**
24: **for** $k = 1$ to $|F|$ **do**
25: $obj += w_{s,v}^{startts,k}$
26: $s \leftarrow startstation$
27: **if** $startts == 0$ **then**
28: $s', t' \leftarrow a_{s,v}^0$
29: $m_{s',v}^{0,t'} \leftarrow 1$
30: **for** $k = 1$ to $|F|$ **do**
31: $z_{s,v}^{t',k} \leftarrow 1$
32: $s \leftarrow s', t_1 \leftarrow t'$
33: **for** $k = 1$ to $|F|$ **do**
34: $t = t_1$
35: **while** $t < Q-1$ **do**
36: $s', t' \leftarrow a_{s,v}^{t,k}$
37: $m_{s',v}^{t,t',k} \leftarrow 1, z_{s',v}^{t',k} \leftarrow 1$
38: $s \leftarrow s', t \leftarrow t'$
39: **return** obj, z, m

We need to extract a feasible primal solution from the solution obtained from slaves. The solution obtained from repositioning slave may not be consistent with the routes computed by routing slave but the solution obtained by routing slave is always feasible solution to the original MSS op-

timization. Therefore, to extract a feasible primal solution, we solve the MSS optimization by fixing the routing variables to the values obtained from routing slave.

Algorithm 2 SolveLDD()

$\alpha \leftarrow 0, iter \leftarrow 0$
repeat
  $o_1, y^+, y^- \leftarrow$ SolveRepositioning(α^{iter})
  $o_2, z, m \leftarrow$ SolveRouting(α^{iter})
  $p \leftarrow$ ExtractPrimal(z, m)
  $\alpha_{s,v}^{t,k,iter+1} = \left[\alpha_{s,v}^{t,k,iter} + \gamma * (y_{s,v}^{+,t,k} + y_{s,v}^{-,t,k} - C_v^* * z_{s,v}^{t,k})\right]_+$
  $iter \leftarrow iter + 1$
until $p - (o1 + o2) \leq \epsilon$

5 Greedy Online Anticipatory Heuristic

The Lagrangian decomposition approach described in Section 4 helps in scaling MSS to larger problems. However, in some cases, we may need to run a large number of iterations (around 100 iterations that can take up to 10 minutes) to get a high quality solution. In online settings, to get a solution within reasonable time, we execute both MSS and LDD with a time-limit of 1 minute. Therefore, as the number of stations and vehicles increases, we may not be able to get a high quality solution within the time-limit.

In this section, we provide Greedy Online Anticipatory Heuristic (GOAH) approach based on online anticipatory algorithms (Mercier and Van Hentenryck 2007) that can quickly provide solution for large scale problems. Typically, online anticipatory algorithms are used to solve large scale online stochastic integer programs. These algorithms optimize for each sample scenario and then select the best solution over all samples. We use a similar idea to develop our approach but instead of optimally solving each sample, we approximate the value obtained for each sample due to scalability issues. In our case, solution for a sample would correspond to a set of repositioning and routing decisions for each vehicle. Each vehicle v has maximum $|\mathcal{S}|$ routing choices where $|\mathcal{S}|$ is the number of stations and it has C_v^* repositioning choices where C_v^* denotes the capacity of vehicle v. As vehicle v already has ξ_v bikes in vehicle, it can pick at most $C_v^* - \xi_v$ bikes or it can drop at most ξ_v bikes. So all possible solutions for each vehicle are $C_v^* * |\mathcal{S}|$.

Unlike in typical anticipatory algorithms where there is only one entity, here, we have multiple carrier vehicles and therefore, the space of possible joint solutions grows exponentially. To address this, we consider one vehicle at a time and use the greedy algorithm (Algorithm 3) to pick the best vehicle policy in each iteration and execute that policy. To pick the best vehicle policy in each iteration, we can use MSS/LDD for a single vehicle to compute individual vehicle's policy. But using MSS/LDD for a single vehicle will not provide desirable gain in runtime (as this will not reduce the complexity in the MSS/LDD formulations due to presence of multiple samples). Therefore, we use a heuristic approach, which computes policy for individual vehicle in two steps: (i) Approximate computation of repositioning decisions (extra bikes available for pickup/dropoff) at each station, timestep in each sample. (ii) Compute policy (repositioning and routing decisions) for a vehicle across all samples by using approximate repositioning decisions calculated in previous step.

Algorithm 3 GOAH()

$e \leftarrow$ComputeApproxRepositioningValues()
$V' = \phi$
while $|V'| < |\mathcal{V}|$ **do**
  for $v \in \mathcal{V} \backslash V'$ **do**
    $val_v = 0$
    $val_v, a_v \leftarrow$ GetVehiclePolicy(v, e)
  $v' \leftarrow \underset{v}{\operatorname{argmin}}\, val_v$
  $e \leftarrow$ ExecutePolicy($v', a_{v'}, e$)
  $V' \leftarrow V' \cup v'$

Approximate computation of repositioning decisions: We use $e_s^{t,k}$ to denote the extra bikes available for pickup or drop off at station s timestep t in sample k. A positive $e_s^{t,k}$ value indicates availability of extra bikes and a negative value indicates the number of bikes which should be dropped to meet the lost demand. These values can be used by vehicles to decide the number of bikes to pick up/drop off at a station timestep pair.

For each sample, we execute "no repositioning" strategy, i.e., simulate the hiring and return of bikes (according to demand observed in the sample) on the current distribution of bikes at the stations, to calculate the available bikes, lost demand and hired bikes at each station timestep pair.

Since the bikes which are not required at timestep t can be used to meet demand at timestep $t+1$, if we use the current computed $e_s^{t,k}$ value, it can be a wrong indicator for vehicle to pick bikes. Therefore, we ensure that $e_s^{t,k}$ values are positive if and only if bikes are not required at any future timesteps. Similarly, unallocated bikes at time t can remain unallocated at time $t+1$, so same bikes will contribute to the $e_s^{t,k}$ values for two different timesteps. To forbid vehicle from considering same bike as available for pickup at two different timesteps, we assume that a station will be visited at most once. This is a valid assumption if the lookahead period is small (i.e., less than one hour).

As the station is visited only once, when vehicle visits any station at any timestep, the decision of the number of bikes picked up/dropped by vehicle should consider future timesteps as well. Therefore, we update the $e_s^{t,k}$ values for each timestep to account for lost demand at future timesteps or requirement of extra bikes at future timesteps. The updated $e_s^{t,k}$ values are used in next step, to guide the policy computation for vehicle.

Computing policy for vehicle given $e_s^{t,k}$ values: In this step, for each sample, we construct a graph. The nodes in the graph correspond to s, c, t, where s is the station id, c is the number of bikes in the vehicle and t is the timestep. An edge is created between node $\{s, c, t\}$ and node $\{s', c', t'\}$ if and only if $\delta_{s,s'}^{t,t'} = 1$ (i.e., s' is reachable from s). The edge between nodes $\{s, c, t\}$ and $\{s', c', t'\}$ indicates that at timestep t vehicle is moving from station s towards station s' and reaches station s' at timestep t'. Positive value

of $c - c'$ indicates vehicle dropped $c - c'$ bikes at station s and negative value indicates vehicle picked $|c - c'|$ bikes from station s. Therefore, an edge in the graph represents the routing and repositioning decision of vehicle. We create an additional sink node T. The policy for a single vehicle is computed by finding the maximum weighted path between vehicle start position and node T.

The weight of edges is defined as follows:

$$R^k(\{s,c,t\},\{s',c',t'\}) =$$
$$\begin{cases} min(0, e_s^{t,k} + c - c') & if\ c <= c' \\ min(c - c', -e_s^{t,k}) & if\ c > c'\ and\ e_s^{t,k} < 0 \\ 0 \quad otherwise \end{cases}$$

$$R^k(\{s,c,t\},T) = \begin{cases} min(c, -e_s^{t,k}) & if\ e_s^{t,k} < 0 \\ 0 \quad otherwise \end{cases}$$

i.e., edges have positive weight on dropping bikes at station with lost demand, negative weight on picking bikes from a station with lost demand and 0 weight otherwise. In other words, weight of edge indicates reduction in lost demand. To incorporate the assumption of visiting a station only once (for picking up or dropping off bikes) in the graph, we need to consider constrained graphs where a set of edges can not be part of a path. This problem is NP-hard (Ziegelmann 2001). However, for Q=3, we can easily incorporate this assumption without using constrained graphs. This is because at t=0, vehicle position is known, so we can avoid creating edges between same stations at different timesteps.

Once we compute policies for individual vehicle, in each iteration, we exeucte the policy for vehicle which maximizes the marginal reduction. As a result of executing vehicle policy, the $e_s^{t,k}$ values are updated to take into account the pickup/drop of bikes at stations by vehicle.

6 Experiments and Results

In this section, we compare our approaches Multi Stage Stochastic Optimization (MSS), Lagrangian Dual Decomposition (LDD) and Greedy Online Anticipatory Heuristic (GOAH) with the following approaches:

1. Static Repositioning (STREP) - In this approach, stations are rebalanced at the end of the day. That is to say, no repositioning is performed during the planning period.
2. Expected Sample Offline Policy Generation (ESOF): In this case, mean demand between stations from past 30 days of data is used as a demand sample to generate offline repositioning policy. We execute MSS/LDD for this expected sample with Q as evaluation decision epochs for different Δ values to generate an offline policy[2].
3. Expected Sample Offline Policy Generation with revenue as objective (ESOF-Rev) (Ghosh et al. 2015): As the objective of the formulation is to maximize the revenue, the approach tries to minimize the cost of vehicle movement in addition to minimizing lost demand. We compare the lost demand values and fuel cost with this approach.
4. Expected Sample Online Policy Generation (ESON): In this case we use the mean demand sample online in the MSS/LDD approaches.

We use MSS($\Delta = x, Q = y$), LDD($\Delta = x, Q = y$), GOAH($\Delta = x, Q = y$) and ESON($\Delta = x, Q = y$) to refer to our approaches when the time interval Δ is set to x and lookahead Q is set to y. We compare the value of lost demand for all the approaches. For MSS, LDD and ESON the time limit to compute solution is set as 1 minute.

Setup: We conducted our experiments by taking the demand distribution over 3 months from 2 real world bike sharing datasets. The first dataset is from Hubway BSS[3] which has 95 stations and the second dataset is from Capital BikeShare BSS[4] which has 305 stations. For the Hubway dataset, we use 3 vehicles for repositioning of bikes and for the Capital BikeShare 6 vehicles are used.

As the historical trip data only contains successful bookings and does not capture the unobserved lost demand, we employ a micro-simulation model with 1 minute of timestep to identify the duration when a station got empty and introduce artificial demand at the empty station based on the observed demand at that station in previous timestep. In the 3 months trip data of both datasets, we have data for 60 weekdays. We use the first 30 weekdays to compute the mean demand sample which is used by ESOF and ESOF-Rev approaches. All the approaches are evaluated on remaining 30 weekdays and the average lost demand is computed over these 30 days.

While computing the repositioning and routing policy at decision epoch t, for our approaches MSS, LDD and GOAH, we consider k samples of customer requests at decision epoch t,t + 1,..,t+Q-1 from past k days (from the evaluation day at the same time). Once the repositioning and routing policy is computed, we evaluate the policy on realized customer demand.

We evaluate all the approaches using the simulation model used in Ghosh, Trick, and Varakantham (2016). The simulation model is run every one minute to serve the customer demand. In contrast, repositioning/routing strategy is computed at an interval of Δ minutes. After the repositing/routing policy is obtained by algorithms at time Δ, $2 * \Delta$,... minutes, the availability of bikes at station and in vehicles is updated. In case there is no free slot at the station available while returning of bikes, the bikes are distributed in the nearby stations. The simulator is run for 6 hours for each day starting at different time of the day. We experimented with starting time as 6:00AM and 03:00PM. At the start of the experiment starting position of vehicles is randomly chosen. The objective of all the algorithms (except ESOF-Rev) is to minimize the lost demand[5].

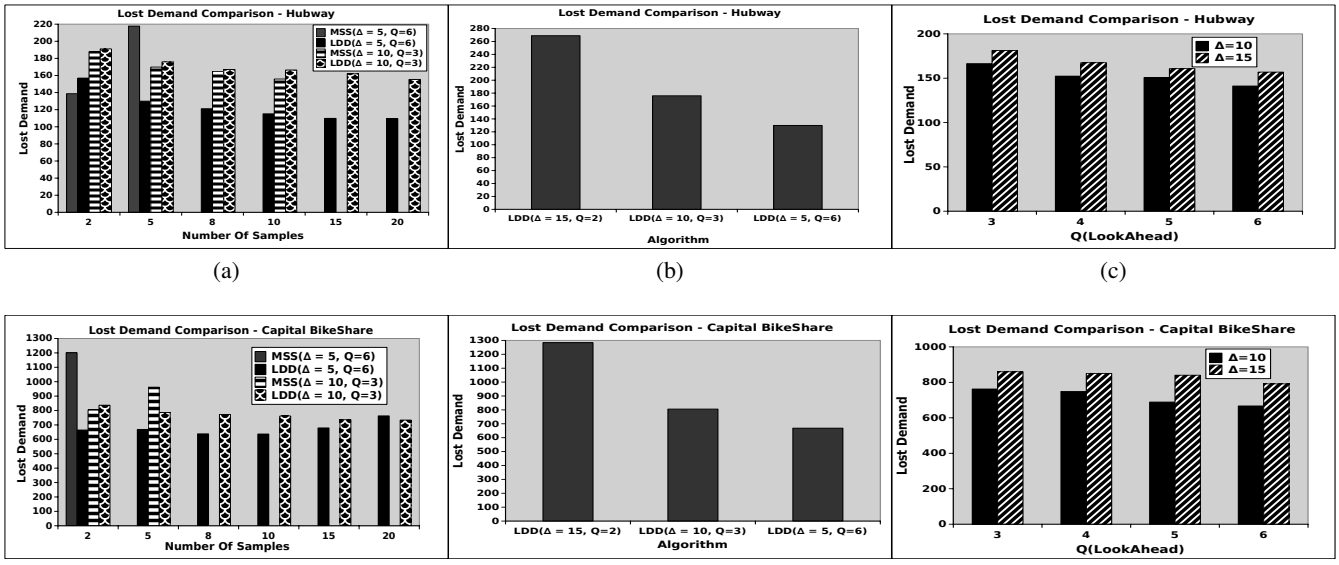
Results: We first show results for MSS and LDD by varying the values of Δ and Q. For GOAH, as described in Section 5, Q is fixed to 3. Therefore, we do not show results for GOAH for different Q values.

[2]For $\Delta = 10$ (in minutes) and evaluation period of 6 hours, we use Q as 36.

[3]http://hubwaydatachallenge.org/trip-history-data/

[4]http://www.capitalbikeshare.com/system-data

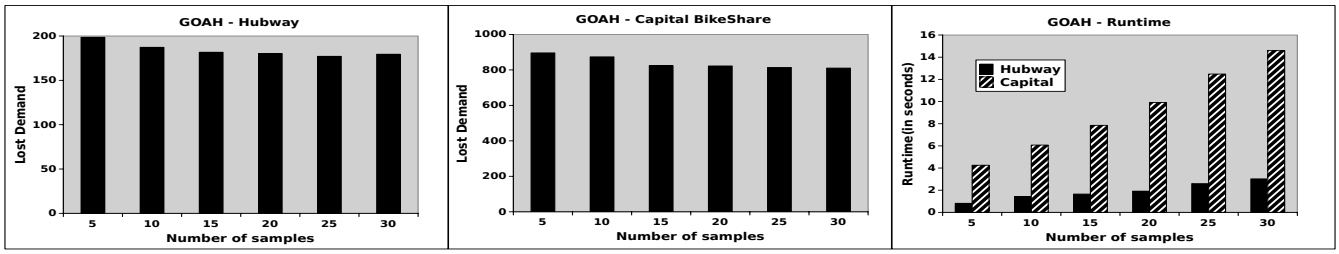
[5]All approaches have been implemented in Java using the IBM CPLEX 12.6.0.



(a) (b) (c)

(d) (e) (f)

Figure 1: (a) and (d) Performance comparison of MSS and LDD. (b) and (e) Lost demand comparison of LDD for different Δ and Q values. (c) and (f) Lost demand comparison of LDD for different Q values.



(a) (b) (c)

Figure 2: Performance comparison of GOAH

	STREP	**ESOF-Rev**	**ESOF**($\Delta = 10$)	**ESON** ($\Delta = 10, Q = 6$)	**LDD** ($\Delta = 10, Q = 6$)	**GOAH** ($\Delta = 10, Q = 3$)
Hubway(3PM)	278.32	225.95	225.93	172.11	141.16	181.67
Capital BikeShare(3PM)	1320.93	960.30	795.28	745.61	666.78	824.03
Hubway(6AM)	184.82	130.30	122.69	98.63	56.85	92.75
Capital BikeShare(6AM)	508.27	456.96	410.60	323.17	288.87	341.71

Table 5: Lost demand reduction

	ESOF-Rev	**ESOF**($\Delta = 10$)	**ESON**($\Delta = 10, Q = 6$)	**LDD**($\Delta = 10, Q = 6$)	**GOAH**($\Delta = 10, Q = 3$)
Fuelcost	7.84	32.64	31.26	30.60	26.60
Revenue	431.10	422.87	461.11	488.58	441.72
Gain	423.25	390.23	429.85	457.98	415.11

Table 6: Fuel cost comparison-Hubway 3pm

Trip Duration	**ESOF-Rev**	**ESOF**($\Delta = 10$)	**ESON**($\Delta = 10, Q = 6$)	**LDD**($\Delta = 10, Q = 6$)	**GOAH**($\Delta = 10, Q = 3$)
0-30	1166.74	1169.26	1212.17	1232.34	1205.55
30-60	93.43	91.24	102.84	112.51	97.73
60-90	11.27	11.11	11.62	12.50	11.39
$>$90	7.88	7.69	8.11	8.22	7.95

Table 7: Number of bikes hired for different trip duration (in minutes)- Hubway 3pm

Number of Samples: In the first set of experiments on MSS and LDD, we experiment with different numbers of samples. Figure (1a) shows results for the Hubway dataset and Figure (1d) shows results for the Capital BikeShare dataset. The X-axis represents the number of samples and Y-axis represents the lost demand values. On the Hubway dataset, MSS could not compute a reasonable quality solution (the optimality gap remained at 80%) within 1 minute for more than 5 samples for $\Delta = 5$ and for more than 10 samples for $\Delta = 10$. On the Capital Bikeshare data set, for Δ = 5 and a time limit of 1 minute, MSS did not find a solution of reasonable quality even when only a single sample was used. On both the datasets, we observe that for LDD, lost demand reduces on increasing the number of samples, but the reduction in lost demand is less than 5% on increasing samples beyond 10. For $\Delta = 5$, on the Capital BikeShare, LDD could not complete even 10 iterations within 1 minute and hence, the lost demand increases on increasing samples from 10 to 15. As the lost demand reduction is not significant after 10 samples and the problem also becomes complex , we use 10 samples for LDD and MSS in the next experiments.

Decision Epoch Duration(Δ): In the next set of experiments, we fix the lookahead period for approaches to 30 minutes and experiment with different decision epoch duration. As $\Delta * Q$ =30 minutes, the value of Q is taken as $\frac{30}{\Delta}$. Figure (1b) shows results on the Hubway dataset and Figure (1e) shows the results on the Capital BikeShare dataset. We fix the number of samples as 10. We only show results for LDD as MSS was not able to compute a reasonable quality solution within 1 minute for 10 samples for majority of scenarios. On both datasets, lost demand decreases on decreasing the value of Δ. This is because vehicle is allowed to make more movements and also because we are looking at demand values at smaller intervals which allows making better online decisions. On decreasing Δ from 10 to 5, lost demand reduces by nearly 15% on both datasets.

Lookahead Period: For a fixed value of Δ, we experiment with different lookahead period durations. We show the results for $\Delta = 10$ with lookahead period between 30 minutes and 1 hour. For $\Delta = 15$, we show the results for lookahead period between 30 minutes and 1.5 hours. Figures (1c) and (1f) show the results for different Q values. On increasing the look ahead period lost demand reduces for both Δ values but the reduction is more with $\Delta = 10$. With $\Delta = 10$ and $Q = 6$ (i.e., lookahead period of 1 hour), the lost demand values are comparable to $\Delta = 5$ and $Q = 6$ (lookahead period of 30 minutes). With $\Delta = 10$, lost demand reduces by 12% on increasing Q value from 3 to 6. But the rate of reduction is low with higher Δ value. As the lost demand reduction provided by $\Delta = 10, Q = 6$ is comparable to lost demand reduction with $\Delta = 5$, we use $\Delta = 10, Q = 6$ for further comparison as this involves lesser vehicle movement.

GOAH Performance: We experiment with different number of samples for GOAH for $\Delta = 10, Q = 3$. In case of GOAH, along with lost demand we also compare the runtime on both datasets with different number of samples. Figure (2a) and (2b) show the lost demand comparison on increasing the number of samples. On both datasets, on increasing samples from 5 to 15, lost demand reduces by 8% but on increasing beyond 15 samples reduction is 2%. With 15 samples, on both datasets, GOAH obtains a runtime of less than 8 seconds (Figure (2c)). Therefore, it is possible to execute GOAH on larger bike sharing systems where it is even difficult for offline approaches to compute a solution. As described later, GOAH provides nearly 35% reduction in lost demand as compared to no repositioning strategy.

Comparison Of Different Algorithms: Next, we compare the reduction in lost demand values obtained by different algorithms. We compare LDD($\Delta = 10, Q = 6$), GOAH($\Delta = 10, Q = 3$) with ESOF($\Delta = 10$), ESOF-Rev, ESON($\Delta = 10, Q = 6$) and STREP on both datasets. For LDD we use 10 samples and for GOAH 15 samples. Table 5 shows the lost demand values on both datasets for 6AM and 3PM. As we can see on both datasets, LDD reduces the lost demand by nearly 50% as compared to STREP and provides 20% gain over ESOF and ESON. The lost demand reduction by GOAH is comparable to ESOF and ESON but it provides improvement in runtime which is the main advantage of using it against other approaches.

Comparison Of Fuel cost: Finally we compare the fuel cost incurred by different algorithms. We use the cost of diesel as 1.5 USD per litre and assume that the vehicle can travel 12 kilometer with 1 litre of fuel. Here, we show the results on the Hubway dataset at 3pm. We obtained similar results on the other dataset. We then compare the fuel cost incurreed by various algorithms in Table 6. As we do not consider the fuel cost in our objective, fuel cost of our algorithm is nearly 4 times the cost of fuel consumed by ESOF-Rev. We also compute the revenue obtained by bike sharing company by using the standard price model where only rides greater than 30 minutes are charged[6]. The revenue increase compensates for the additional fuel cost in case of LDD. With GOAH, both the fuel cost and revenue gain are less than LDD. Overall gain provided by GOAH is less than ESOF-Rev.

As the rides having travel time less than 30 minutes are included in the subscription cost, we also compare the number of bikes hired for different trip duration by various algorithms (Table 7). Once again, LDD provides the best results. As the major percentage of bikes are hired for duration 0-30 minutes, the lost demand reduction of these rides does not directly contribute to daily revenue. But this reduction will help in increasing the number of new subscribers which will provide additional profit to bike sharing companies.

7 Conclusion

We develop a multi-stage stochastic optimization formulation for computing online repositiong and routing policy for vehicles in bike sharing systems. We also provide a lagrangian based decomposition and greedy based heuristic approach to efficiently reduce lost demand for large scale bike sharing systems. In future, this work can be extended to compute online policies which can simultaneously and efficiently optimize reduction of lost demand and fuel cost of vehicles. Another possible direction is to include uncertainty in travel time of vehicles, so as to compute robust policies.

[6]https://www.thehubway.com/pricing/day

8 Acknowledgements

This work was partially supported by the Singapore National Research Foundation through the Singapore-MIT Alliance for Research and Technology (SMART) Centre for Future Urban Mobility (FM).

References

Chemla, D.; Meunier, F.; and Calvo, R. W. 2013. Bike sharing systems: Solving the static rebalancing problem. *Discrete Optimization* 10(2):120–146.

Contardo, C.; Morency, C.; and Rousseau, L.-M. 2012. *Balancing a dynamic public bike-sharing system*, volume 4. Cirrelt.

Fisher, M. L. 1985. An applications oriented guide to lagrangian relaxation. *Interfaces* 15(2):10–21.

Ghosh, S., and Varakantham, P. 2017. Incentivising the use of bike trailers for dynamic repositioning in bike sharing systems. In *International Conference on Automated Planning and Scheduling (ICAPS)*.

Ghosh, S.; Varakantham, P.; Adulyasak, Y.; and Jaillet, P. 2015. Dynamic redeployment to counter congestion or starvation in vehicle sharing systems. In *International Conference on Automated Planning and Scheduling (ICAPS)*.

Ghosh, S.; Varakantham, P.; Adulyasak, Y.; and Jaillet, P. 2017. Dynamic repositioning to reduce lost demand in bike sharing systems. *Journal of Artificial Intelligence Research* 58:387–430.

Ghosh, S.; Trick, M.; and Varakantham, P. 2016. Robust repositioning to counter unpredictable demand in bike sharing systems. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Lowalekar, M.; Varakantham, P.; and Jaillet, P. 2016. Online spatio-temporal matching in stochastic and dynamic domains. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Mercier, L., and Van Hentenryck, P. 2007. Performance analysis of online anticipatory algorithms for large multistage stochastic integer programs. In *IJCAI*, 1979–1984.

Pfrommer, J.; Warrington, J.; Schildbach, G.; and Morari, M. 2014. Dynamic vehicle redistribution and online price incentives in shared mobility systems. *IEEE Transactions on Intelligent Transportation Systems* 15(4):1567–1578.

Raviv, T.; Tzur, M.; and Forma, I. A. 2013. Static repositioning in a bike-sharing system: models and solution approaches. *EURO Journal on Transportation and Logistics* 2(3):187–229.

Schuijbroek, J.; Hampshire, R.; and van Hoeve, W.-J. 2017. Inventory rebalancing and vehicle routing in bike sharing systems. *European Journal of Operational Research* 257(3):992–1004.

Shu, J.; Chou, M. C.; Liu, Q.; Teo, C.-P.; and Wang, I.-L. 2013. Models for effective deployment and redistribution of bicycles within public bicycle-sharing systems. *Operations Research* 61(6):1346–1359.

Zhang, G.; Smilowitz, K.; and Erera, A. 2011. Dynamic planning for urban drayage operations. *Transportation Research Part E: Logistics and Transportation Review* 47(5):764–777.

Ziegelmann, M. 2001. *Constrained shortest paths and related problems*. Ph.D. Dissertation, Universitätsbibliothek.