Dynamic Redeployment to Counter Congestion
or Starvation in Vehicle Sharing Systems

Supriyo Ghosh, Pradeep Varakantham, Yossiri Adulyasak†, Patrick Jaillet‡

School of Information Systems, Singapore Management University, Singapore 178902
†Singapore-MIT Alliance for Research and Technology (SMART), Massachusetts Institute of Technology
‡Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology
supriyog.2013@phdis.smu.edu.sg, pradeepv@smu.edu.sg, yossiri@smart.mit.edu, jaillet@mit.edu

Abstract

Extensive usage of private vehicles has led to increased traffic congestion, carbon emissions, and usage of non-renewable resources. These concerns have led to the wide adoption of vehicle sharing (ex: bike sharing, car sharing) systems in many cities of the world. In vehicle-sharing systems, base stations (ex: docking stations for bikes) are strategically placed throughout a city and each of the base stations contain a pre-determined number of vehicles at the beginning of each day. Due to the stochastic and individualistic movement of customers, there is typically either congestion (more than required) or starvation (fewer than required) of vehicles at certain base stations. As demonstrated in our experimental results, this happens often and can cause a significant loss in demand. We propose to dynamically redeploy idle vehicles using carriers so as to minimize lost demand or alternatively maximize revenue for the vehicle sharing company. To that end, we contribute an optimization formulation to jointly address the redeployment (of vehicles) and routing (of carriers) problems and provide two approaches that rely on decomposability and abstraction of problem domains to reduce the computation time significantly. Finally, we demonstrate the utility of our approaches on two real world data sets of bike-sharing companies.

1 Introduction

Shared Transportation Systems (STS) offer attractive alternatives to deal with serious concerns of private transportation such as increased carbon emissions, traffic congestion and usage of non-renewable resources. Popular examples of STS are bike sharing (ex: Capital Bikeshare in Washington DC, Hubway in Boston, Bixi in Montreal, Velib in Paris, Wuhan and Hangzhou Public Bicycle in Hangzhou) and car sharing (ex: Car2go in Seattle, Zipcar in USA) systems, which are installed in many major cities around the world. Bike sharing systems are widely adopted with 747 active systems, a fleet of over 772,000 bicycles and 235 systems in planning or under construction (Meddin and DeMaio 2015). A bike-sharing system (BSS) typically has a few hundred base stations scattered throughout a city. At the beginning of the day, each station is stocked with a pre-determined number of bikes. Users with a membership card can pickup and return bikes from any designated station, each of which has a finite number of docks. At the end of the work day, carrier vehicles (ex: trucks) are used to move bikes around so as to return to some pre-determined configuration at the beginning of the day.

Due to the individual movement of customers according to their needs, there is often congestion (more than required) or starvation (fewer than required) of bikes on aggregate at certain base stations. Figure 1 provides the number of instances when stations become empty or full during the day (summed over each month) for a leading bike-sharing company which performs redeployment of bikes to stations only once at the end of the day. A full station can be considered as being indicative of congestion and an empty station can be considered as being indicative of starvation. At a minimum, there are around 100 cases of empty stations and 100 cases of full stations per day and at a maximum there are about 750 cases of empty stations and 330 cases of full stations per day. This serves as the motivation for this paper, where we employ dynamic redeployment during the day to better match demand with supply.

As demonstrated in (Fricker and Gast 2012) and our experimental results, this (particularly starvation) can result in a significant loss of customer demand. Such loss in demand can have two undesirable outcomes: (a) loss in revenue; (b) increase in carbon emissions, as people can resort to fuel burning modes of transport. So, there is a practical need to minimize the lost demand and our approach is to dynamically redeploy bikes with the help of carriers (typically medium to large sized trucks) during the day. However, because carriers incur a cost in performing redeployment, we have to consider the trade-off between minimizing lost demand (alternatively maximizing revenue) and cost of using carriers for redeployment. Henceforth, we refer to this problem as the Dynamic Redeployment and Routing Problem (DRRP).

Minor variations of DRRP are applicable to more general shared transportation systems, empty vehicle redistribution in Personal Rapid Transit (PRT) (Lees-Miller, Hammersley, and Wilson 2010) and dynamic redeployment of emergency vehicles (Yue, Marla, and Krishnan 2012; Saisubramanian, Varakantham, and Chiu 2015).

Given the practical benefits of bike sharing systems and
the challenging nature of setting up such systems to operate efficiently, there have been a wide variety of research papers addressing the problem of lost demand and other issues pertinent to it. The key distinction from existing research on bike sharing is that we consider the dynamic redeployment of bikes in conjunction with the routing problem for carriers.

DRRP is an NP-Hard problem and therefore, we focus on principled approximations. Specifically, our key contributions are as follows:

1. A mixed integer and linear optimization formulation to maximize profit for the bike sharing company by trading off between:
   - computing the optimal re-deployment strategy (i.e., how many vehicles have to be picked or dropped from each base station and when) for bikes; and
   - computing the optimal routing policy (i.e., what is the order of base stations according to which redeployment happens) for each of the carriers.
2. A Lagrangian dual decomposition method to exploit the weak dependency between the component which computes re-deployment strategy for bikes and the component which computes routing policy for carriers.
3. An abstraction mechanism that groups nearby base stations to reduce the size of the decision problem and consequently, improve scalability.

Extensive computational results on real-world datasets of two bike-sharing companies, namely Capital Bikeshare (Washington, DC) and Hubway (Boston, MA) demonstrate that our techniques improve revenue and operational efficiency of bike-sharing systems.

2 Related Work

Given the practical benefits of bike sharing systems, they have been studied extensively in the literature. We focus on three threads of research that are of relevance to this paper. First thread of papers focus on the problem of finding routes at the end of the day for a fixed set of carriers to achieve the desired configuration of bikes across the base stations.

1. A mixed integer and linear optimization formulation to maximize profit for the bike sharing company by trading off between:
   - computing the optimal re-deployment strategy (i.e., how many vehicles have to be picked or dropped from each base station and when) for bikes; and
   - computing the optimal routing policy (i.e., what is the order of base stations according to which redeployment happens) for each of the carriers.
2. A Lagrangian dual decomposition method to exploit the weak dependency between the component which computes re-deployment strategy for bikes and the component which computes routing policy for carriers.
3. An abstraction mechanism that groups nearby base stations to reduce the size of the decision problem and consequently, improve scalability.

Extensive computational results on real-world datasets of two bike-sharing companies, namely Capital Bikeshare (Washington, DC) and Hubway (Boston, MA) demonstrate that our techniques improve revenue and operational efficiency of bike-sharing systems.
\( S \) represents the set of base stations. \( V \) represents the set of carrier vehicles that can be employed to redeploy bikes. \( C\# \) is a vector representing docking capacity of all stations, with \( C\#_s \) representing docking capacity of a station \( s \in S \). Similarly, \( C^* \) is a vector representing storage capacity of different carriers. Note that we assumed a fixed set of carriers in our model. However, in deciding the optimal number of carriers, our optimization approach can be run multiple times with different numbers of carriers to perform sensitivity analysis.

Distribution of bikes at a base station \( s \) at any time \( t \) is given by \( d^s_{#,t} \). Hence, initial distribution at any station \( s \) (provided as input) is denoted by \( d^s_{#,0} \). Similarly, total number of bikes present in a carrier \( v \) at any time \( t \) is given by \( d^*_{v,t} \) while the initial allotment of bikes \( d^*_{v,0} \) is provided as input. \( \sigma^v_s(t) \) captures the initial distribution of a carrier and is set to 1 if carrier \( v \) is stationed at station \( s \) initially. For ease of notation in the optimization formulation, we use the generic \( \sigma^v_s(t) \) and set it to 0, if \( t > 0 \).

\( F^{t,k}_{s,s',v} \) represents the expected demand at time step \( t \) going out from station \( s \) and reaching station \( s' \) after \( k \) time steps. For instance, to compute a redeployment and routing strategy for Mondays, this would be computed by averaging demand over all the Mondays. \( R^{t,k}_{s,s'} \) represents the revenue obtained by the company if a bike is hired at time \( t \) from station \( s \) and returned at station \( s' \) after \( k \) time steps. \( P_{s,s',v} \) represents the penalty for any carrier vehicle to travel from \( s \) to \( s' \).

Given the customer demand of bikes between different stations, the goal is to maximize profit or minimize loss of the bike sharing company by redeploying bikes using carrier vehicles (to satisfy customer demand). Notice that, minimizing lost demand would also minimize number of stations that are empty or full. As indicated earlier, we refer to this as the Dynamic Redeployment and Routing Problem (DRRP).

**Proposition 1** Solving DRRP is an NP-Hard problem.

**Proof Sketch.** We show that DRRP is a generalisation of 3-set partitioning problem, a known NP-Hard problem.

### 4 Solution Approach

We employ a data driven approach to solve DRRP. That is to say, we compute redeployment and routing strategies for a given training data set of demand values and evaluate the performance of the computed redeployment and routing strategies on a testing data set. Specifically, we provide a Mixed Integer Linear Problem (MILP) formulation for solving DRRP with expected demand values, \( F \) that are obtained from the training data set. For ease of understanding, the decision and intermediate variables employed in the formulation are provided in Table 1.

---

1. While we provide a generalized model for initial location of carriers, our model can easily be extended such that carriers are enforced to start and return in a particular depot.

2. We do not directly minimize lost demand, because that can result in a significant cost due to carrier vehicles. Profit provides the right trade-off between minimizing lost demand (maximizing revenue) and reducing cost due to carriers.

---

<table>
<thead>
<tr>
<th>Category</th>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision</td>
<td>( y^+_{s,v,t} )</td>
<td>Number of bikes picked from ( s ) by carrier ( v ) at time ( t )</td>
</tr>
<tr>
<td>Decision</td>
<td>( y^-_{s,v,t} )</td>
<td>Number of bikes dropped at ( s ) by carrier ( v ) at time ( t )</td>
</tr>
<tr>
<td>Decision</td>
<td>( z^v_{s,s',v,t} )</td>
<td>Set to 1 if carrier ( v ) has to move from ( s ) to ( s' ) at time ( t )</td>
</tr>
<tr>
<td>Intermediate</td>
<td>( x^{t,k}_{s,s',v} )</td>
<td>Number of bikes moving from ( s ) at time ( t ) to ( s' ) at ( t+k )</td>
</tr>
<tr>
<td>Intermediate</td>
<td>( d^v_{s,t} )</td>
<td>Number of bikes present in station ( s ) at time-step ( t )</td>
</tr>
<tr>
<td>Intermediate</td>
<td>( d^*_{v,t} )</td>
<td>Number of bikes present in carrier ( v ) at time ( t )</td>
</tr>
</tbody>
</table>

Table 1: Decision and Intermediate Variables

\[
\begin{align*}
\min \sum_{t,k,s,s',v} R^{t,k}_{s,s'} x^{t,k}_{s,s',v} + \sum_{t,v,s,s',v} P_{s,s'} z^v_{s,s',v,t} + \sum_{t,k,s,s'} d^k_{s,t} z^v_{s,s',v,t} \\
\text{s.t. } d^v_{s,t} + \sum_{k,v,s,s'} (x^{t-k,k}_{s,s',v} - x^{t,k}_{s,s',v,t}) + \sum_{v,s,s'} (y^+_{s,v,t} - y^-_{s,v,t}) = d^v_{t+1}, \forall t,s \quad (1) \\
x^{t,k}_{s,s',v} &\leq d^k_{s,t}, \forall t,k,s,s',v \quad (2) \\
d^v_{t+1} + \sum_{s \in S} (y^+_{s,v,t} - y^-_{s,v,t}) &\leq d^v_{t+1}, \forall t,v \quad (3) \\
\sum_{k \in S} z^v_{s,s',v,t} &\leq 1, \forall t,v \quad (4) \\
\sum_{j \in S, v \in V} z^v_{s,j,v,t} &\leq 1, \forall t,v \quad (5) \\
\sum_{k \in S} a^v_s(s) &\leq C^*, \forall v \quad (6) \\
y^+_{s,v,t} + y^-_{s,v,t} &\leq C^*, \forall t,v \quad (7) \\
0 &\leq x^{t,k}_{s,s',v} \leq F^{t,k}_{s,s',v}, 0 \leq d^k_{s,t} \leq C\#, 0 \leq y^+_{s,v,t} \leq C^*, \quad (8) \\
0 &\leq y^-_{s,v,t} \leq C^*, 0 \leq d^v_{t} \leq C^* \quad (9)
\end{align*}
\]

Table 2: SolveDRRP()

We have access to flow of bikes in \( F \). Note that we only have the information about successful bike trips, thus we employ a standard method adopted in (Shu et al. 2013) to predict the actual demand, where the demand is represented using a Poisson distribution with mean computed from historical data. One of our goals is to compute a redeployment of bikes and it should be noted that because of redeployment, the number of bikes at a station will be different to what was observed in the training dataset. Hence, flow of bikes between station \( s \) and \( s' \) at time step \( t \) for \( k \) time steps suggested by our MILP will be different from the observed flow of bikes in the data, i.e., \( F^{t,k}_{s,s',v} \). To represent this, we introduce a proxy variable, \( x^{t,k}_{s,s',v} \) for \( F^{t,k}_{s,s',v} \) that is set based on \( F \) and the number of bikes available in the source station after redeployment. \( x \) is included in the objective to ensure most of the expected demand is satisfied. For this reason, \( x \) is only an intermediate variable that is proxy to expected demand, \( F \). The lost demand is calculated as the difference between
demand sample \( (F) \) and the served demand \( (x) \).

To represent the trade-off between lost demand (or equivalently revenue from bike jobs) and cost of using carrier vehicles accurately, we employ the dollar value of both quantities and combine them into overall profit. This objective is represented in Equation 1 of the MILP in \textbf{SOLVE\text{-}DRRP}().

The key to this formulation for solving DRRP are the flow preservation, movement and capacity constraints for bikes, stations and carriers. Here, we provide a brief description of these constraints:

1. **Flow of bikes in and out of stations is preserved**: Constraints (2) enforces this flow preservation by ensuring equivalence of the number of bikes at station, \( s \) at time, \( t \) at time, \( t+1 \) (i.e., \( d^F_{s,t} \)) to the sum of bikes at station, \( s \) at time, \( t \) (i.e., \( d^F_{s,t+1} \)) and net number of bikes arriving at station \( s \) after time step \( t \) (i.e., \( \sum x_{s,k}^t - \sum x_{s,k'}^t \)), minus the net number of bikes picked up from station \( s \) by all carriers \( v \) (i.e., \( \sum y_{s,v}^+ - y_{s,v}^- \)).

2. **Flow of bikes between any two stations follows the transition dynamics observed in the data**: As a subset of arrival customers can be served if number of bikes present in the station is less than arrival demand, constraints (3) ensures that flow of bikes between any two station \( s \) and \( s' \) should be less than the product of number of bikes present in the source station \( s \) (i.e. \( d^F_{s,t} \)) and the transition probability that a bike will move from \( s \) to \( s' \) (according to customer demand (i.e. \( F_{s,k,s'} \)\( F_{s,k,s}\))

3. **Flow of carriers in and out of stations is preserved**: Since \( \sigma_{s} = 0 \) for all \( t > 0 \), constraints (4) ensures that flow out of a station \( s \) for a carrier \( v \) at time \( t \) (i.e., \( \sum_{k \in S} z_{s,k,v}^t \)) is equivalent to flow of \( v \) into the station \( s \) at time \( t - 1 \) (i.e., \( \sum_{k \in S} z_{s,k,v}^{t-1} \)). For \( t = 0 \), depending on \( \sigma_{s} \) is given as input, this constraint will ensure carrier flow moves appropriately out of the initial locations.

5. **Only one carrier can be in one station at a time step**: Constraints (6) ensures this by restricting the maximum carrier flow in a station as one.

6. **Carrier can pick up or drop off bikes from a station by being at the station**: Constraints (7) enforces that the number of bikes picked up or dropped off at a time is bounded by whether the station is visited at that time step.

7. **Station capacity is not exceeded when redeploying bikes**: Constraints (8) ensures that the number of bikes at a station, \( s \) is lower than the number of docks available at that station (i.e., \( C^D_{s} \)).

8. **Carrier capacity is not exceeded when redeploying bikes**: Constraints (9) ensures that the number of bikes dropped off or picked up from any station at every time step and in aggregation is always less than the carrier capacity.

## 5 Decomposition Approach for Solving DRRP

We now provide a decomposition approach to exploit the minimal dependency that exists in the MILP of \textbf{SOLVE\text{-}DRRP}() between the routing problem (how to move carrier vehicles between base stations to pick up or drop off bikes) and the redeployment problem (how many bikes and from where to pick up and drop off bikes). The following observation highlights this minimal dependency:

**Observation 1** In the MILP of Table 2:

- \( y^+ \) and \( y^- \) variables capture the solution for the redeployment problem.
- \( z \) variables capture the solution for the routing problem.

These sets of variables only interact due to constraints (7). In all other constraints of the optimization problem, the routing and redeployment problems are completely independent.

In order to exploit Observation 1, we use the well known Lagrangian Dual Decomposition (Fisher 1985; Gordon et al. 2012) technique. While this is a general purpose approach, its scalability, usability and utility depend significantly on whether the right constraints are dualized\(^{3}\) and if primal solution can be extracted from an infeasible dual solution\(^{4}\). If the right constraints are not dualized or primal solution is not extracted, then the underlying Lagrangian based optimization may not provide desired scalability and solution quality. In order to provide a sense of the overall flow, the pseudo code for DRRP is provided in Algorithm 1. We first identify the decomposition of the optimization problem into a master problem and slaves (\textbf{SOLVE\text{-}REDEPLOY}() and \textbf{SOLVE\text{-}ROUTING}()). As highlighted in observation 1, only constraints (7) contains a dependency between routing and redeployment problems. Thus, we dualize constraints (7) using the price variables, \( \alpha_{s,t,v} \) and obtain the Lagrangian as follows:

\[
\mathcal{L}(\alpha) = \min_{x,y,s} \left[ - \sum_{t,k,s',s} R_{s,s'}^t \cdot x_{s,k,s'}^t + \sum_{t,v,s'} P_{s,v}^t \cdot z_{s,v,s'}^t \right] + \sum_{s,t,v} \alpha_{s,t,v} \cdot \left( y_{s,v}^+ - y_{s,v}^- - C^v_{s} \cdot z_{s,v,s'}^{t-1} \right)
\]

\[
\min_{y,s} \left[ - \sum_{t,k,s',s} R_{s,s'}^t \cdot x_{s,k,s'}^t + \sum_{s,t,v} \alpha_{s,t,v} \cdot \left( y_{s,v}^+ + y_{s,v}^- \right) \right] + \sum_{s,t,v} \alpha_{s,t,v} \cdot \left( C^v_{s} \cdot z_{s,v,s'}^t \right)
\]

In Equation (2), the first two terms correspond to the redeployment problem and the second two terms correspond to the routing problem. Thus, we have a decomposition of the dual problem into two slaves. Specifically, the slave optimization formulations corresponding to the redeployment

\(^{3}\)We do not consider the labor/capital cost in the optimization model as they are constant (decided by strategic planning) and would not alter the results. However, we have a buffer on fuel cost to account for any other costs pertaining to day-to-day operations.

\(^{4}\)Note that this constraint does not preclude a carrier from staying in the same station.
\[
\min_{y^+, y^-} - \sum_{t,k,s,s'} R_{t,k}^s \cdot (y^+_s + y^-_{s'}) + \sum_{s,t,v} \alpha_{t,s,v} \cdot (y^+_s + y^-_t)
\]
\[\text{s.t. } \text{Constraints 2, 3, 4 & 8 hold}\]

**Table 3: SolveDeploy()**

\[
\min_{x, y} \sum_{s,t,v} P_{s,t,v} \cdot x_{s,t,v} - \sum_{s,t,v} \alpha_{s,t,v} \cdot C_v^t \cdot \sum_i z_{i,s,t,v}
\]
\[\text{s.t. } \text{Constraints 5, 6 & 9 hold}\]

and routing problems after decomposition are given in Table 3 and Table 4 respectively.

To obtain the final solution for the original optimization problem of Table 2, at the master, we optimize \(\max_\alpha L(\alpha)\). From Equation (2), given an \(\alpha\), the dual value corresponding to the original problem is obtained by adding up the solution values from the two slaves. This master optimization problem is solved iteratively using sub-gradient descent on price variables, \(\alpha\):

\[
\alpha_{s,t,v}^{k+1} = \left[\alpha_{s,t,v}^k + \gamma \cdot (y^+_s + y^-_t - C_v^t \cdot \sum_i z_{i,s,t,v})\right]_+
\]  

(3)

where \([\cdot]_+\) notation indicates that if the value within square brackets is less than 0, then we consider it as 0 and if it is positive, we take that value as it is. This is so, because we have dualized a less than equal to constraint and a value of less than zero indicates there is no violation of the constraint. \(\gamma\) corresponds to step parameter that is derived using standard strategies highlighted in (Bertsekas 1999, Section 6.3.1). The value within parenthesis () in Equation (3) is computed from the solutions of the two slaves.

Convergence in the process is detected when the difference between the primal objective (defined as \(p\) in Algorithm 1) and the dual objective (the sum of the slave’s objectives \(o_1, o_2\)) is less than a predetermined threshold value \(\delta\). In order to determine convergence of the algorithm and also understand the progress towards computing the optimal solution, we need to solve the primal problem in conjunction with the dual solution. Therefore, extracting the best primal solution after each iteration of solutions from slaves is critical. This is also challenging because the solution obtained from slaves may not always be feasible for the original problem in Table 2.

**Observation 2** The infeasibility in the dual solution arises because routes of the carriers (computed by routing slave) may not be consistent with redeployment of bikes (computed by redeployment slave). However, solution of the routing slave is always feasible and can be fixed in the optimization problem of Table 2 to obtain a feasible primal solution.

Let \(Z_{s,t}^t = \sum_{s'} z_{s,s',t}^t\). We extract the primal solution by solving the following optimization problem provided in Table 5 and subtract the routing cost from the objective to get the primal solution.

**Algorithm 1: SolveLDD(drrp)**

1. Initialize: \(\alpha^0, i \leftarrow 0\);
2. repeat
   1. \(o_1, x, y^- \leftarrow \text{SolveDeploy}(\alpha^i, drrp)\); \(o_2, z \leftarrow \text{SolveRouting}(\alpha^i, drrp)\);
   2. \(\alpha^{i+1} \leftarrow \left[z_{s,t,v}^i + \gamma \cdot (y^+_s + y^-_t - C_v^t \cdot \sum_i z_{i,s,t,v}^i)\right]_+\);
   3. \(p, x, y^+_t, y^-_t \leftarrow \text{ExtractPrimal}(z, drrp)\);
   4. \(i \leftarrow i + 1\);
3. until \(p - (o_1 + o_2) < \delta\);
4. return \(p, x, y^+_t, y^-_t\)

**Table 5: ExtractPrimal()**

**Proposition 2** (Fisher 1985) Error in solution quality obtained by Lagrangian dual decomposition method in Algorithm 1 is bounded by \(p - (o_1 + o_2)\).

It should be noted that MILP in Table 2 provides the optimal solution.

**6 Abstraction Approach for Solving DRRP**

Even after applying LDD, we can only scale to problems with at most 60 stations, 38 time-step and 6 carriers. However, in bigger cities, the number of base stations is in the order of hundreds. To ensure scalability to bigger cities, we propose a heuristic approach that employs abstraction. Specifically, we have the following key steps:

- Create an abstract DRRP with abstract stations, each of which is a grouping of original base stations.
- Solve the abstract DRRP using LDD and obtain routing and redeployment strategy over abstract stations.
- Derive the routing and redeployment strategies for the original DRRP from the routing and redeployment strategies for abstract DRRP.

**Create abstract DRRP:** Concretely, the first step in this approach is to generate the abstract DRRP, \(\langle \tilde{S}, V, \tilde{C}^t, \tilde{d}^t, \tilde{d}^*, 0 \rangle, \tilde{\delta}_V, \tilde{F}, \tilde{R}, \tilde{P} \rangle\) from the original DRRP. Everything related to carriers in the abstract DRRP remains the same as in the original DRRP. In practice, revenue, \(F_{s,k}^t\), is only dependent on the time of the day, \(t\) and number of time steps, \(k\) for which the bike is hired and not on \(s\) and \(s'\). Hence, we can assume that the revenue model also remains the same in the abstract DRRP.

We outline below how the other elements of the abstracted DRRP tuple are computed from the original DRRP:

- Stations in abstracted DRRP, \(\tilde{S}\): Grouping of stations, \(S\) into abstract stations can either be provided by an expert or
\[
\max \sum_{t,k,s,s'} R_t^{k,s} \cdot x_t^{k,s} - \sum_{t,v,s,s'} P_{s,v} \cdot z_t^{s,v}
\]
\[\text{s.t. } \text{Constraints 2-9 hold and } \quad \sum_{s \in \tilde{s},s' \in \tilde{s}'} z_t^{s,v} = \tilde{z}_t^{s,v} \forall \tilde{s},s',t,v \quad (14)\]

Table 6: RETRIEVEDRRP() 

done manually or computed through clustering approaches (ex: k-means clustering). Thus, each abstracted station, \( \tilde{s} \) is a set of original base stations.

- Capacity of an abstract station, \( C^\#_\tilde{s} = \sum_{s \in \tilde{s}} C^\#_s \). The capacity of a abstract station \( \tilde{s} \) is the sum of capacities of all the stations \( s \in \tilde{s} \).
- Initial distribution of the abstracted station: \( d^\#_{\tilde{s},0} = \sum_{s \in \tilde{s}} d^\#_{s,0} \). Initial distribution of a abstract station \( \tilde{s} \) is the sum of initial distribution of all the stations \( s \in \tilde{s} \).
- Initial distribution of carrier: \( d^\#_{s,0} = 1 \), if \( \exists s \in \tilde{s} \) and \( \sigma^0_{t,s} = 1 \). That is to say, the carrier \( v \) is initially located in abstract station \( \tilde{s} \) if its original location (station \( s \)) belongs to the abstract station \( \tilde{s} \).
- Flow in abstracted DRRP: \( F^t_{\tilde{s},k} = \sum_{s \in \tilde{s},s' \in \tilde{s}'} F^t_{s,s'} \). Flow from abstracted station \( \tilde{s} \) to \( \tilde{s}' \) is calculated as the sum of flows between any station \( s \in \tilde{s} \) to \( s' \in \tilde{s}' \) in the original DRRP.
- Cost model for the carriers in abstract DRRP: \( P_{\tilde{s},s'} = \max \{ s \in \tilde{s}, s' \in \tilde{s}' \} P_{s,s'} \). While there are multiple models possible, we consider the conservative option of taking the worst case penalty. Specifically, we take the maximum penalty for traveling between any pair of stations \( s \in \tilde{s} \) and \( s' \in \tilde{s}' \).

Solve abstract DRRP: We use LDD from previous section to solve the abstract DRRP. There are two key outputs: (a) Redeployment strategy, \( \tilde{y} \) for moving bikes between abstract stations; and (b) Routing strategy, \( \tilde{z} \) for moving carriers between abstract stations, \( \tilde{s} \) at different time steps.

Derive strategies for original DRRP: We retrieve the solution of DRRP from solutions obtained for abstract DRRP. \( \tilde{z} \) is the routing strategy obtained by solving the abstract DRRP, where \( \tilde{z}_{t,v}^{s,s'} = \sum_{s' \in \tilde{s}',s \in \tilde{s}} z_{t,v}^{s,s'} \) is present in abstract station \( \tilde{s} \) at time step \( t \). Table 6 provides the optimization model to retrieve the solution for DRRP with the assumption of carrier travels one station in each time step. We solve the global MIP SolveDRRP() provided in Table 2 with an additional set of constraints (14) that ensure a carrier can only be present in a base station at any time step if the station belongs to the abstract station where the carrier is located in the abstract DRRP solution. It also enforces that the decision variable \( z_{t,v}^{s,s'} \) can only be 1 if \( s \in \tilde{s}, s' \in \tilde{s}' \) and \( \tilde{z}_{t,v}^{s,s'} = 1 \). In the MILP of Retrieve-DRRP(), we explicitly set the decision variables \( z_{t,v}^{s,s'} \) to 0 if \( s \in \tilde{s}, s' \in \tilde{s}' \) and \( \tilde{z}_{t,v}^{s,s'} = 0 \). Thus RetrieveDRRP() becomes an easier optimization problem than SolveDRRP().

\[
\min \sum_{s,v} P_{s,v} \cdot z_{s,v}
\]
\[\text{s.t. } d_{t}^{*,k} + \sum_{s} (Y_{s}^+ - Y_{s}^-) \cdot z_{t,v}^{s,s'} = d_{t}^{*,t+1}, \forall t \in \tilde{T} \quad (22)\]
\[\sum_{s,v} z_{t,v}^{s,s'} = 1, \forall s \in \tilde{s} (Y_{s}^+ + Y_{s}^-) > 0 \quad (23)\]
\[\sum_{v} z_{t,v}^{s,s'} - \sum_{v} z_{t+1,v}^{s,s'} = \sigma^t (s), \forall t \in \tilde{T}, s \in \tilde{s} \quad (24)\]

Table 7: GETSTATIONREDEPLOY(v, Z, d^*_v)

Table 8: GETINTRAROUTING(\tilde{s}, Y)

6.1 Discussion

As we are abstracting the base stations based on their relative distance, all the base stations within an abstract station are located nearby. So, in reality it is possible for a carrier to visit all the base stations of an abstract station within one time-step. In this case, the mechanism to retrieve the DRRP solution from abstract station level solution changes and is outlined here.

We first compute redeployment strategy, \( z \) at the level of base stations for each carrier over the entire horizon and then compute the routing strategy within each abstract station. The optimization problem of Table 7 employs the constants, \( Z \) to obtain a base station level redeployment strategy, \( y \). If \( s \) is an original station and \( \tilde{s} \) is an abstract station, then let \( Z_{\tilde{s}}^s = 1 \) if \( s \in \tilde{s} \) and \( \sum_{s'} z_{t,v}^{s,s'} = 1 \). Our objective function specified in (15) is to maximize the overall revenue of the agency that implies maximum number of customers are served. One of the key differentiating constraints that has not been used earlier is constraints (19). This ensures that total number of bikes picked up or dropped off from all base stations in an abstract station is equal to the number of bikes picked up or dropped off in the abstract station level redeployment strategy.

Given the base station level redeployment strategy, \( Y \), we now compute the best route within the stations of an abstract station, \( \tilde{s} \). This problem can be solved locally for each ab-
strat station, $s$, where carrier $v$ is redeploying at time step $t$. If we have $|T|$ time-steps and $|V|$ carriers, we can solve $|T| \cdot |V|$ subproblems separately. To figure out the initial location, we find a station within the abstract station which is nearest to the station from where the carrier has exited in the previous time-step. Since the position of carriers is known at the first time step, we know the starting location. Such an approach automatically minimizes the inter-cluster routing. Table 8 provides the optimization formulation to solve each subproblem. The objective specified in (21) is to minimize the routing cost. Constraints (22) ensure the redeployment number from each station ($Y_j$) is satisfied and Constraints (23) ensure that each base station where a redeployment is required is visited only once.

7 Experimental Settings

We evaluate our approaches with respect to run-time, revenue for company and lost demand on real world and synthetic data sets. These data sets contain the following data: (1) Customer trip records that are indicative of successful bookings. We predict demand from these trip records. (2) Number of active docks in each station (i.e. station capacity) and initial distribution of bikes in the station at the beginning of a day. (3) Geographical locations of base stations. From the longitude and latitude information of stations, we calculate the relative distance between two stations. (4) Revenue model of the agency. (5) Cost of fuel for carriers.

We generated our synthetic data set as follows: (a) We take a subset of the stations from the real world data set (b) Customer demands, station capacity, geographical location of stations and initial distribution are drawn from the real world data for those specific stations. (c) We take the same revenue and cost model discussed earlier from real datasets. Because of limited scalability of MILP and LDD without abstraction, we are only able to evaluate run-time performance on small scale synthetic problems.

To the best of our knowledge, there is no other approach that addresses this problem nor does there exist an approach that can easily be adapted to solve our problem. Hence we compare our approaches against current practice of redeploying at the end of the day (in which user activities during the rebalancing period are negligible) with respect to: (a) overall revenue generated for the agency; and (b) lost demand. We compute the outcome of current practice by simulating flow of bikes between stations to be proportional to the flow observed in the data. With the flow, we compute distribution of bikes and based on that, we can compute the revenue and loss in demand.

Synthetic Dataset: We have three sets of results on the synthetic data set. Firstly, we compare the runtime performance of LDD (SOLVELDD()) with the global MILP (SOLVEDRRP()) in Figure 2(b). X-axis denotes the scale of the problem where we varied the number of stations from 5 to 50. Y-axis denotes the total time taken in seconds on a logarithmic scale. LDD significantly outperforms global MILP once the number of stations is above 10. Specifically, global MILP was unable to finish within a cut-off time of 6 hours for any problem with more than 20 stations, while LDD was able to solve problems with 50 stations in an hour.

In the second set of results we demonstrate the convergence of LDD. LDD can achieve the optimal solution if the duality gap i.e. the gap between primal and dual solution becomes zero. Figure 2(a) shows that the duality gap for a 20 station problem is only 1%. While, we do not show the results here, on larger problems we are able to get a solution with duality gap of less than 0.5%.

Finally, we demonstrate the performance of abstraction in comparison with optimal on a problem with 30 base stations. We grouped those 30 base stations into 8 abstract stations. Then we run the LDD based optimization on both the base station and abstraction station problems. Table 9 shows the effect of abstraction on the generated revenue and execution time based on five random instances of customer demand. While there is a very small drop in revenue (0.2%) due to abstraction, we get a very significant (at least an order of magnitude gain) in runtime.

Real-world Datasets: Majority of our results are provided on the CapitalBikeShare dataset. This data set has 305 active stations and we consider 50 abstract stations (obtained through k-means clustering). The planning horizon is 38 (30 minute intervals during the working hours from 5AM-12AM).

We now provide the performance comparison between our approaches and current practice (i.e., no redeployment during the day) with respect to lost demand and revenue generated for the bike-sharing company. We generate the overall

---

1 Data is from two leading US bike sharing companies: CapitalBikeShare [http://capitalbikeshare.com/system-data] and Hubway [http://hubwaydatachallenge.org/trip-history-data]

2 Typically, first 30 minutes for subscription rides is free. After that money is charged. In our model, to ensure consistency, we can represent revenue for first 30 minutes as the subscription fees divided by the average number or rides.

3 Mileage results are shown in Table 2 of (Fishman, Washington, and Haworth 2014) and http://www.globalpetrolprices.com/diesel_prices/#USA, shows that diesel price is 1.01 USD per liter, but we assumed it as 1.5 USD to consider worst case.

10 All the linear optimization models were solved using IBM ILOG CPLEX Optimization Studio V12.5 incorporated within python code on a 3.2 GHz Intel Core i5 machine with 4GB DDR3 RAM
<table>
<thead>
<tr>
<th>Instance</th>
<th>Revenue</th>
<th>Runtime (sec)</th>
<th>Revenue</th>
<th>Runtime (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23580</td>
<td>51</td>
<td>23640</td>
<td>3840</td>
</tr>
<tr>
<td>2</td>
<td>23627</td>
<td>106</td>
<td>23678</td>
<td>3540</td>
</tr>
<tr>
<td>3</td>
<td>23610</td>
<td>57</td>
<td>23727</td>
<td>3120</td>
</tr>
<tr>
<td>4</td>
<td>23613</td>
<td>49</td>
<td>23645</td>
<td>3150</td>
</tr>
<tr>
<td>5</td>
<td>23519</td>
<td>45</td>
<td>23590</td>
<td>3119</td>
</tr>
</tbody>
</table>

Table 9: Effect of Abstraction

<table>
<thead>
<tr>
<th>Whole day (5am-12am)</th>
<th>Peak period (5am-12pm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revenue gain</td>
<td>Lost demand reduction</td>
</tr>
<tr>
<td>Mean</td>
<td>3.47 %</td>
</tr>
<tr>
<td>Mon</td>
<td>2.33 %</td>
</tr>
<tr>
<td>Tue</td>
<td>3.07 %</td>
</tr>
<tr>
<td>Wed</td>
<td>3.30 %</td>
</tr>
<tr>
<td>Thu</td>
<td>2.86 %</td>
</tr>
<tr>
<td>Fri</td>
<td>2.51 %</td>
</tr>
<tr>
<td>Sat</td>
<td>3.87 %</td>
</tr>
<tr>
<td>Sun</td>
<td>3.01 %</td>
</tr>
</tbody>
</table>

Table 10: Revenue and lost demand comparison

Figure 3: Sensitivity analysis: (a) Revenue comparison (b) Lost demand comparison

<table>
<thead>
<tr>
<th>Mon Tue Wed Thu Fri Sat Sun</th>
<th>Revenue Gain (%)</th>
<th>Lost Demand Reduction (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.94 5.93 4.45 5.90 6.27 2.20 3.15</td>
<td>42.6 60.7 58.5 54.7 77.2 69.8 74.0</td>
<td></td>
</tr>
</tbody>
</table>

Table 11: Revenue and lost demand comparison (Hubway)

In this paper we addressed the dynamic redeploy problem in shared transportation systems. Our approach based on the Lagrangian Dual Decomposition and an abstraction based mechanism, addresses two key challenges (a) Provide an near-optimal policy for the dynamic redeployment of idle vehicles in conjunction with the routing solution for carriers (b) Provide a scalable solution for the real-world large scale problems. The empirical results on multiple real and synthetic data sets shown that our dynamic redeployment approach is not only able to achieve the original goal of reducing lost demand, but is also able to improve revenue for the bike sharing company, by using their existing resources. In future this work can be extended with a robust optimization technique which can account for all the realization of different demand scenarios.

8 Conclusion
Figure 4: Correlation of demand and supply: (a) Without redeployment (b) With dynamic redeployment

Acknowledgments
The research described in this paper was funded in part by the Singapore National Research Foundation (NRF) through the Singapore-MIT Alliance for Research and Technology (SMART) Center for Future Mobility (FM).

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