Singapore Management University

Institutional Knowledge at Singapore Management University

Dissertations and Theses Collection (Open Access)

Dissertations and Theses

12-2017

Proactive sequential resource (re)distribution for improving efficiency in urban environments

Supriyo GHOSH
Singapore Management University, supriyog.2013@phdis.smu.edu.sg

Follow this and additional works at: https://ink.library.smu.edu.sg/etd_coll

 Part of the Databases and Information Systems Commons, and the Data Storage Systems Commons

Citation

GHOSH, Supriyo. Proactive sequential resource (re)distribution for improving efficiency in urban environments. (2017).
Available at: https://ink.library.smu.edu.sg/etd_coll/140

This PhD Dissertation is brought to you for free and open access by the Dissertations and Theses at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Dissertations and Theses Collection (Open Access) by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email cherylds@smu.edu.sg.

Proactive Sequential Resource (Re)distribution for Improving Efficiency in Urban Environments

SUPRIYO GHOSH

SINGAPORE MANAGEMENT UNIVERSITY

2017

Proactive Sequential Resource (Re)distribution for Improving Efficiency in Urban Environments

by

Supriyo Ghosh

Submitted to School of Information Systems in partial fulfillment of the requirements for the Degree of Doctor of Philosophy in Information Systems

Dissertation Committee:

Pradeep VARAKANTHAM (Supervisor/Chair)
Associate Professor of Information Systems
Singapore Management University

Akshat KUMAR
Assistant Professor of Information Systems
Singapore Management University

Hoong Chuin LAU
Professor of Information Systems
Singapore Management University

Patrick JAILLET
Professor of Electrical Engineering and Computer Science
Massachusetts Institute of Technology

Singapore Management University
2017

Copyright (2017) Supriyo Ghosh

Proactive Sequential Resource (Re)distribution for Improving Efficiency in Urban Environments

Supriyo Ghosh

Abstract

Due to the increasing population and lack of coordination, there is a mismatch in supply and demand of common resources (e.g., shared bikes, ambulances, taxis) in urban environments, which has deteriorated a wide variety of quality of life metrics such as success rate in issuing shared bikes, response times for emergency needs, waiting times in queues etc. Thus, in my thesis, I propose efficient algorithms that optimise the quality of life metrics by proactively redistributing the resources using intelligent operational (day-to-day) and strategic (long-term) decisions in the context of urban transportation and health & safety.

For urban transportation, Bike Sharing System (BSS) is adopted as the motivating domain. Operational decisions are crucial for BSS, because the stations of BSS are often not balanced due to uncoordinated movements of resources (i.e., bikes) by customers. The imbalanced stations lead to significant loss in demand and increase the usage of private transportation and therefore, defeat the primary objective of BSS which is to reduce carbon footprint. In order to reduce the carbon footprint, I contribute three operational decision making approaches for sequential redistribution of bikes: (i) Optimising lost demand through dynamic redistribution; (ii) Optimising lost demand through robust redistribution; and (iii) Optimising lost demand through incentives. In the first approach, I consider the expected demand for multiple time steps to find a redistribution solution and provide novel decomposition and abstraction mechanisms to speed up the solution process. This approach is useful for BSS with consistent demand patterns. Therefore, the second approach proposes a robust redistribution solution using the notion of two-player adversarial game to address the scenarios where the demand has high variance. For the

third approach, within the central budget constraints of the operators, a mechanism is designed to incentivise the customers for executing the bike redistribution tasks by themselves. The experimental results on two real-world data sets of Capital Bikeshare (Washington, DC) and Hubway (Boston, MA) BSS demonstrate that our approaches significantly reduce the average and worse case lost demand over the current practices.

For health & safety, Emergency Medical System (EMS) is adopted as the motivating domain. EMS is an extremely sensitive and critical domain for public healthcare services, because reducing the response times for emergency incidents by a few seconds can save a human life. In order to reduce the response times, I propose strategic decision making approach for EMS so as to place base stations at "right" location and allocate "right" number of ambulances on those bases. An accelerated version of greedy algorithm on top of an existing data-driven optimisation formulation is proposed to jointly consider the placement of bases and allocation of ambulances. Subsequently, I provide insights to improve the operational decisions of EMS for dynamic redistribution of ambulances by incorporating the exact real-world dynamics of EMS into the existing data-driven optimisation formulation. Experimental results on real-world data sets demonstrate that both our strategic and operational decisions improve the efficacy of EMS over the existing approaches.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank all the individuals who have helped me in completing the journey of Ph.D.

First and foremost, I would like to thank my advisor, Associate Professor Pradeep Varakantham for his support and proper guidance in every stage of my Ph.D. journey. I am extremely fortunate to be advised by him as his passion for research and enthusiasm for work has always inspired me to overcome the critical challenges. Apart from the professional advise and support for research, I have received great amount of suggestions and help from him on a personal front. His advise has contributed in developing myself as an individual and academician.

I would also like to thank my co-advisor, Assistant Professor Akshat Kumar for his invaluable reviews, detailed suggestions and careful guidance. His guidance has helped me a lot in learning the art of scientific paper writing and presentation. I thank Professor Hoong Chuin Lau for his inspirational words, support and guidance on various research problems. I also thank Associate Professor Shih-Fen Cheng for his suggestions and support on multiple occasions.

I thank Professor Michael A. Trick from Carnegie Mellon University (CMU) for his valuable suggestions and guidance during my stay at Pittsburgh. His keenness in discussing the minute details about the research problems has helped me in overcoming some major research challenges. I also thank LARC for providing me the opportunity of spending a year at CMU as a research scholar. I would like to express my gratitude for Professor Patrick Jaillet from Massachusetts Institute of Technology (MIT) for his valuable suggestions, thoughtful comments and reviews on many

research projects and papers. I thank Yossiri Adulyasak and Sanjay Dominik Jena for being an extremely supportive collaborator and mentor. The thorough discussions with them on multiple occasions and their valuable comments and suggestions have helped me to beat some of the extremely hard research problems.

I have been lucky to have several collaborators and friends during my Ph.D. I would like to thank all my friends and colleagues: Asrar, Pritee, Meghna, Rajiv, Tanvi, Pallavi, Murali, Kapil, Ritesh, Larry, Jiali, Cen Chen, Fu Na, Sandhya, James, Pavneet and Sougata for their guidance, friendship, and unconditional support. A special thank to my friends: Atreya, Chanakya, Arka, Dibya, Nanki, Rhea, Subhra and Priti for helping me in overcoming all sort of difficulties and making my life easier in Singapore.

Finally, I sincerely thank my family for their support and encouragement. I am grateful to my parents for the unconditional love, sacrifices and consistent support for every decision of my life. A special thank to my elder sisters, brothers-in-law and my little niece Arohi for always supporting and encouraging me.

To

my parents, for their unconditional love and encouragement

&

my sisters, for always being understanding and supportive

Chapter 1

Introduction

1.1 Motivation and Background

Due to the fast urbanisation, the populations in today's cities are growing rapidly. According to the Department of Economic and Social Affairs of United Nations, around 54% of the world's population lives in urban areas and the growth is expected to reach 66% by 2050[1]. More importantly, the populations in urban areas are projected to experience substantial growth to approximately 86% and 80% by 2050 for high-income and upper-middle-income countries, respectively. While people migrates to cities for a better quality of life in terms of education, job opportunities and pleasant future, the lack of coordination leads to great challenge of mismatch between the demand for resources and the supply of resources, which has a detrimental effect on a wide range of quality of life metrics.

In particular, the increasing demand and the involuntary movements of people according to their needs lead to an inherent mismatch between the demand and supply of public resources (e.g., shared bikes, taxis) in urban transportation. This mismatch between the demand and supply of common resources has deteriorated a wide range of quality of life metrics such as success rate in issuing shared bikes,

[1]The data is collected from https://esa.un.org/unpd/wup/publications/files/wup2014-highlights.Pdf

waiting times in queues etc. These problems lead to an extensive usage of private vehicles in the mega cities as the city residents eventually purchase motor vehicles as soon as they can afford them, which is an expected phenomenon studied by Sperling and Salon (2002). For instance, the number of car owners per thousand population in China in 2002 was 16 (Dargay, Gately, & Sommer, 2007), which has increased by 300 percent from 2002 to 2008 (Fan, 2008).

Unfortunately, the extensive usage of private vehicles significantly contributes to the major growing concerns such as global warming, air pollution, usage of non-renewable resources, traffic congestion, climate change and emission of greenhouse gases (Wright & Fulton, 2005). Therefore, there is a practical need to develop intelligent decisions for proactive repositioning[2] of resources in public transportation (Cervero, 2013), which can reduce the usage of private transportation and contribute significantly in building sustainable and smart cities.

In addition, the rapid and unplanned urbanisation can have a major negative impact in health & safety of people[3]. The influx of migrants can lead to overcrowded cities and can be a catalyst for rapid transmission of infectious diseases (Neiderud, 2015). Furthermore, the overpopulation significantly increases the number of emergency incidents such as road accidents. For instance, the road traffic injuries constitute the ninth leading cause for death and illness and it is expected to rise to third source by 2030 (Peden, Scurfield, Sleet, Mohan, Hyder, Jarawan, Mathers, et al., 2004). For these reasons, the increasing demand has deteriorated the quality of life metrics (i.e., response times for emergency needs) for emergency medical systems (EMSs). Therefore, it is extremely important to manage and redistribute the common resources (i.e., locations of base stations, ambulances, fire bikes) of EMS in a fashion that can serve the emergency incidents within minimal response time.

To address the aforementioned issues in both the transportation and health & safety domain, the city planners need to determine intelligent decisions for the

[2]Note that, we use "redistribution", "repositioning", "rebalancing", "relocation" and "dynamic matching" synonymously throughout the document.

[3]http://www.who.int/world-health-day/2010/media/whd2010background.pdf

placement and management of public resources. Therefore, in my thesis, I propose intelligent decision making approaches that optimise the quality of life metrics (e.g., success rate in issuing shared bikes, response times for emergency needs) by proactively redistributing the public resources. There are two level of decision making that has the potential to improve efficiency of urban environments:

- Operational level: day-to-day decisions associated with repositioning of resources, i.e., how many resources to reposition, from where and how?
- Strategic level: capital and labor decisions, i.e., how many resources (e.g., bikes, ambulances, taxis) to buy, where to place them and how many permanent employees to hire?

In this thesis, I explore operational and strategic decision making problems in the context of urban transportation and health & safety. The objective is to proactively redistribute the resources to better meet the future demand, which in turn improves the efficiency of the system through better utilisation of public resources. Figure (1.1) provides a quick view of different types of decision making problems and briefly explains the contributions in each category.

1.1.1 Resource Redistribution for Urban Transportation

To alleviate the problem of traffic congestion and high carbon emission by the private vehicles, Bike Sharing Systems (BSSs) are widely adopted in major cities (e.g., *Capital Bikeshare* in Washington DC, *Hubway* in Boston, *Bixi* in Montreal, *Vélib'* in Paris, etc.) of the world (Meddin & DeMaio, 2016). BSSs provide an attractive alternative to the private transportation, specifically for the last-mile delivery and short distance travels. Therefore, I employ BSS as the motivating domain. In a BSS, base stations are strategically placed throughout a city and each station is stocked with a pre-determined number of bikes at the beginning of the day. Customers hire the bikes from one station and return them at another station.

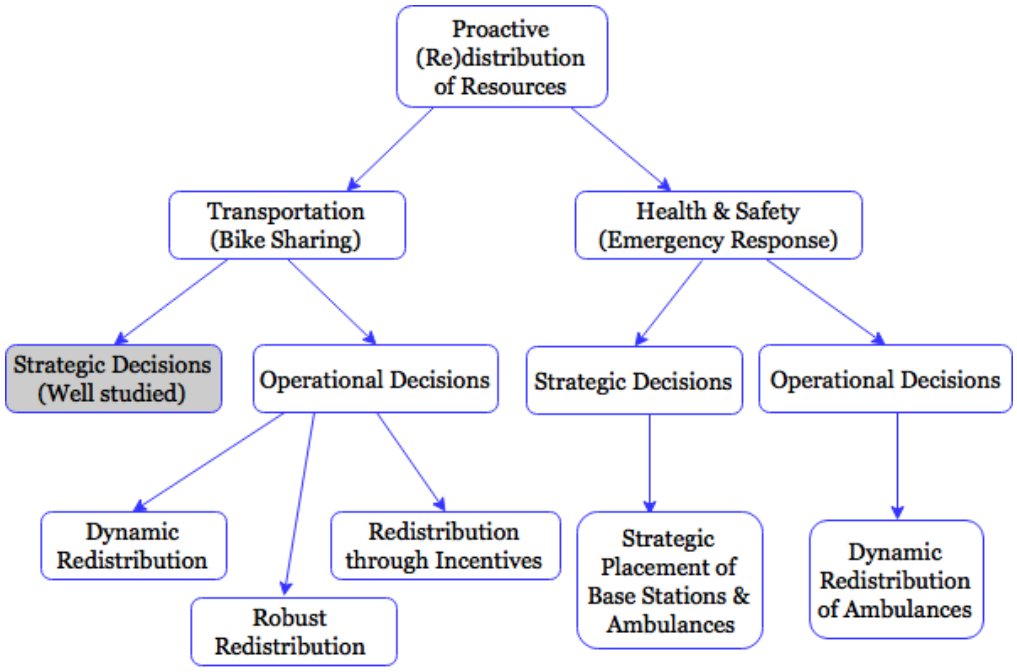


Figure 1.1: Characterisation of resource redistribution problems and contributions in each category.

Strategic decision making problems in BSS such as the placement of stations, discovering the optimal docking capacity and other issues pertinent to it, are heavily studied in recent past. There has been a wide range of papers (Shu, Chou, Liu, Teo, & Wang, 2013; Lin & Yang, 2011; Lin, Yang, & Chang, 2013; Martinez, Caetano, Eiró, & Cruz, 2012; Angelopoulos, Gavalas, Konstantopoulos, Kypriadis, & Pantziou, 2016) for efficiently solving the strategic decision making problems in BSS.

However, due to the involuntary movements of customers hiring bikes, there is either congestion (more than required) or starvation (fewer than required) of bikes at base stations. The congestion/starvation is a common phenomenon that leads to a large number of unsatisfied customers resulting in a significant loss in customer demand. While the primary objective of BSS is to reduce carbon footprint, due to the issue of lost demand, people resort back to their private transportation which contribute towards carbon emissions. Therefore, operational decision making for redistributing the idle bikes on a daily basis to reduce the lost demand during the day is a practically important and challenging problem for BSS.

To reduce the lost demand in BSS, the operators currently employ myopic heuristics to rebalance the system based on the assessment of the recently observed demand. Given the practical benefits of BSSs, there has been a wide variety of literature that studied the problems related to lost demand and other issues pertinent to it. The first thread of research focuses on static rebalancing (Benchimol, Benchimol, Chappert, De La Taille, Laroche, Meunier, & Robinet, 2011; Chemla, Meunier, & Wolfler Calvo, 2013; Schuijbroek, Hampshire, & Van Hoeve, 2017) at the end of the day or when the movements of bikes by customers are insignificant to achieve the desired configuration of bikes across the base stations. To solve the dynamic redistribution (i.e., the movements of bikes by the customers during the planning period is significant and not negligible) problem during the day, most of the papers (Pfrommer, Warrington, Schildbach, & Morari, 2014; Contardo, Morency, & Rousseau, 2012) provide myopic solution based on the assessment of near future demand. Recent research (Singla, Santoni, Bartók, Mukerji, Meenen, & Krause, 2015; Pfrommer et al., 2014) proposes the idea of incentivizing customers for assisting in dynamic redistribution.

However, these myopic solutions fail to properly capture the demand surges as they do not consider the long-term future demand and the underlying uncertainties associated with the demand. Therefore, in order to reduce the demand surges, I focus on three operational decision making problems for sequential redistribution of bikes: (i) Optimising lost demand through dynamic redistribution; (ii) Optimising lost demand through robust redistribution; and (iii) Optimising lost demand through incentives.

1.1.1.1 Optimising Lost Demand through Dynamic Redistribution

The existing real-world data sets demonstrate that the demand follows a consistent pattern (particularly in the weekdays) for several BSSs. So, it is important to consider multi-step expected demand (computed from past data) while computing the redistribution strategies for the idle bikes using carrier vehicles. While existing re-

search (Shu, Chou, Liu, Teo, & Wang, 2010; Shu et al., 2013) considered the future expected demand to compute the redistribution strategy, they assume that repositioning of bikes from one station to another is always possible without considering the routing of vehicles. In contrast, our goal is to find a dynamic repositioning strategy for the bikes using carrier vehicles, while considering the routes of the vehicles and the future demand for multiple time steps. This problem is referred to as Dynamic Repositioning and Routing Problem (DRRP).

1.1.1.2 Optimising Lost Demand through Robust Redistribution

Although the demand follows a consistent pattern for several BSSs, in many cities the future demand is unpredictable and changes dynamically. While the offline multi-step algorithms based on expected future demand are suitable for situations with stable demand patterns, they perform poorly when demand varies throughout the day. The unpredictable movements of bikes by the customers can make the planned offline redistribution solution irrelevant when customers return the required number of bikes to a station or counterproductive when vehicles pick up bikes from a station where customers need bikes. Therefore, it is important to consider the variance in the demand along with the expected demand during the planning period. The solution should also be robust to adapt the changes in customer demand and provide a reasonable outcome even in the worse case scenarios. This problem is referred to as Dynamic Repositioning and Routing Problem under demand Uncertainty (DRRPU).

1.1.1.3 Optimising Lost Demand through Incentives

While BSS is employed extensively primarily because of its nature friendly behaviour, the operators typically utilise a fleet of carrier vehicles to counter the loss in customer demand which contributes towards the carbon emissions and other issues pertinent to it. Moreover, this fuel burning mode of redistribution suffers with limited resource constraints (i.e., a fixed set of vehicles owned by the operators) and

incurs a significant amount of routing, labor costs. As an alternative, some operators (e.g., *Citibike* in New York City) have introduced the notion of bike trailers for repositioning the bikes. A bike trailer is an add-on to a bike that can help with carrying 3-5 bikes at once. While existing research by O'Mahony and Shmoys (2015) used trailers for the repositioning, they assume that all the tasks for the trailers can be achieved with dedicated staff which is not an economically viable option. In contrast, our goal is to introduce a potentially self-sustaining and environment friendly system of dynamic repositioning with the help of bike trailers and use incentivisation method for crowdsourcing the tasks of the trailers to users while ensuring a given budget constraint. This problem is referred to as Dynamic Repositioning and Routing Problem with Trailers (DRRPT).

1.1.2 Resource Redistribution for Health & Safety

For health & safety, Emergency Medical System (EMS) is employed as the motivating domain. In a typical EMS, the supply of resources corresponds to ambulances at base stations and demand corresponds to emergency requests. In EMS, a set of base stations are strategically placed in a city and each station is stocked with a few ambulances. Once an emergency request arrived, the operator dispatch the nearest available ambulance to serve the request. The assigned ambulance reaches the incident location, provides initial treatment, transfers the patient to a nearby hospital and then returns back to the same base from where it has been dispatched. Note that, EMS is an extremely sensitive and important domain for public health-care services, because arriving at the incident location a few seconds early can save a human life. Therefore, real-world EMS operators use sensitive performance metrics such as bounded time response (e.g., percentage of requests served within 15 minutes) and bounded risk response (e.g., least response time within which 80% of the requests are assisted).

On the contrary to BSS where the movements of resources (i.e., bikes) by the

customers lead to an imbalance system on a daily basis, the locations of the resources (e.g., base stations and ambulances) in EMS are strategically controlled by the operators. Therefore, I focus on two decision making problems for improving the efficiency of EMS: (a) Strategic decision making for placement of bases and ambulances; and (b) Operational decision making for dynamic redistribution of ambulances to bases.

1.1.2.1 Strategic Decision Making Problem in EMS

While strategic solutions for facility location problems in large-scale disaster response systems enjoy a rich history (Toregas, Swain, ReVelle, & Bergman, 1971; Jia, Ordóñez, & Dessouky, 2007; Huang, Kim, & Menezes, 2010), they are not applicable in EMS where incidents happen everyday and the patterns of how incidents occur change over time. Therefore, our goal is to improve the infrastructure for EMS and specifically the construction of base stations at the "right" locations and to allocate "right" number of ambulances in each of the bases so that the response times for emergency requests are minimised. I take the advantage of past incident data to learn the patterns of demand which generates insights for the strategic decisions. Apart from the complex dynamics of demand in real-world data, the strategic decision making in EMS is a computationally challenging task because of the: (i) exponentially large action space arising from having to consider combinations of potential base locations, which themselves can be significant; and (ii) direct impact on the performance of the ambulance allocation problem, which decides the allocation of ambulances to bases.

1.1.2.2 Operational Decision Making Problem in EMS

The recent research papers (Yue, Marla, & Krishnan, 2012; Saisubramanian, Varakantham, & Chuin, 2015) have introduced the notion of data-driven models for improving operational decisions of EMSs. Although these solution approaches have proven to be fruitful for optimising key performance metrics of EMS, the solutions are gen-

erated either greedily or using approximate dynamics of EMS. Specifically, the existing approaches do not consider the ambulance dispatch policy while achieving the optimal distribution of ambulances. Therefore, there is a significant gap between the objective of the optimisation model and the actual objective when simulating in the real-world environment. So, our goal is to improve solution for dynamic redistribution (on a daily basis) of ambulances to bases by considering the exact dynamics of real-world EMS.

1.2 Contributions

This section describes my contributions in order to tackle the aforementioned issues in the real-world decision making problems. My key contributions towards solving the operational decision making problems in BSS are summarised below:

- I propose a multi-step dynamic redistribution solution for solving the DRRP which primarily aims to jointly minimise the lost demand in BSS and the routing cost for vehicles. I introduce a mixed integer linear program (MILP) formulation to generate a multi-step repositioning solution for bikes using vehicles while also considering the routes for vehicles and future expected demand. The objective of the MILP is to maximise profit for the BSS that considers the trade-off between: (i) maximising served demand, and (ii) minimising cost incurred by vehicles.

- Unfortunately, the complexity of the MILP for solving the DRRP grows exponentially with the number of base stations and hence, cannot solve problems associated with the large-scale real-world BSSs. Therefore, I contribute two approaches that rely on decomposition (that exploits the weak dependency between the bike repositioning and vehicle routing) and abstraction (aggregation of base stations to reduce the size of the problem) mechanisms to reduce the computation time significantly. It is further shown that these approxima-

tion approaches guarantee to provide a near-optimal solution.

- To counter the uncertainty in future demand or alternatively for solving DRRPU, I propose a robust redistribution approach that minimises the loss in customer demand while considering the possible uncertainties in future demand. I develop a scenario generation approach based on an iterative two player adversarial game to compute a strategy of redistribution by assuming that the environment can generate a worse demand scenario (out of the feasible demand scenarios) against the current redistribution solution. These robust redistribution strategies are generated in an online fashion and executed on a simulation built on real-world data set of BSS using a rolling horizon framework.

- To develop a sustainable mode of redistribution for solving DRRPT, I provide an optimisation formulation that generates redistribution tasks for bike trailers by considering a set of training demand scenarios so as to minimise the expected lost demand over those training scenarios. In addition, within the budget constraints of the operator, a mechanism is designed to incentivise the potential customers who intend to execute the redistribution tasks.

- The experimental results on a simulation built on two real-world data sets of Capital Bikeshare (Washington, DC) and Hubway (Boston, MA) BSS demonstrate that our approaches reduce the average and worse case lost demand by 22% and 10% over the current practice. Furthermore, extensive results are provided to demonstrate that the approach for utilising bike trailers for the redistribution is highly competitive to the existing fuel burning mode of redistribution while being green.

My key contributions towards solving the decision making problems in EMS are summarised below:

- For solving the strategic decision making problem in EMS, I present an incre-

mental greedy approach to discover the placement of bases on top of a data-driven optimisation approach (adopted from recent literature) for allocation of ambulances to the bases so as to minimise the response times for the emergency requests. Furthermore, an accelerated version of the greedy approach, referred to as lazy greedy is employed, to speed up the solution process. We show that our approach can be utilised to optimise widely used performance metrics such as bounded time response (e.g., percentage of requests served within 15 minutes) and bounded risk response (e.g., least response time within which 80% of the requests are assisted). Using the properties of submodular optimisation, it is shown that our greedy algorithm provides quality guaranteed solutions for one of the objectives employed in real EMSs.

- For improving the operational decisions of EMS, I propose to dynamically redistribute the ambulances to bases using an optimisation model that exactly imitates the behaviour of real-world EMS. Specifically, I introduce an efficient way to incorporate the ambulance dispatch constraints within the existing optimisation model. However, as the extended optimisation model suffers with scalability issue for large-scale EMSs, I provide two approaches to approximately solve the optimisation problem and show that these heuristics perform well for real-world EMSs.

- The solution approach for strategic decision making is further validated by employing a real-life event driven simulator built on a real-world data set. In case of optimising the bounded time response objective, our approach is able to serve at least 3% extra requests within 15 minutes in comparison to all the existing approaches. On the other hand, in case of optimising the bounded risk response, by utilising less than 70% of existing bases, our approach is proven to be highly competitive to other benchmark approaches.

- Using two real-world EMS data sets, I empirically show that the proposed enhancements for operational decisions in EMS have the potential to serve up

to 2.5% extra requests within threshold time bound over existing approaches.

1.3 Organization of the Dissertation

The rest of this document is organised as follows: Chapter 2 describes the formal models, related literature and existing solution approaches for decision making in BSSs. Chapter 3 presents the approaches for optimising lost demand through dynamic redistribution in BSS. Chapter 4 explains how to counter the uncertainties in future demand through robust redistribution while optimising the lost demand. Chapter 5 elaborates the self-sustainable mode of redistribution through incentives. Chapter 6 describes the formal models, related literature and existing solution approaches for decision making in EMS. Chapter 7 elucidates the solution for the strategic decision making problem in EMS. Chapter 8 delineates the insights for improving the operational decisions of EMS. Finally, the conclusion is presented in Chapter 9.

Chapter 2

Models and Background for Decision Making in Bike Sharing Systems

This chapter explains the models, related research and existing solution approaches for decision making in Bike Sharing System (BSS).

2.1 Models

As indicated in Chapter 1, I categorised the operational decision making problems in BSS into three threads according to the characteristics of the demand patterns and the primary objective of reducing carbon footprint: (i) Dynamic Repositioning and Routing Problem (DRRP); (ii) Dynamic Repositioning and Routing Problem under demand Uncertainty (DRRPU); and (iii) Dynamic Repositioning and Routing Problem using bike Trailers (DRRPT). In this section, I briefly introduce the notations and models for these three categories of operational decision making problems. For the ease of understanding, the key notations and their definitions are compactly represented in Table (2.1).

Element	Definition
s	The indicator used to represent a station.
v	The indicator used to represent a carrier vehicle.
t	The indicator used to represent a time period or decision epoch.
S	The set of base stations.
V	The set of carrier vehicles.
$C_s^{\#}$	The capacity (i.e., the number of docks) of station s.
C_v^*	The capacity (i.e., the number of slots for bikes) of vehicle v.
$d_s^{\#,0}$	Initial distribution of bikes at base station s.
$d_v^{*,0}$	Initial distribution of bikes at vehicle v.
$\sigma_{v,s}^0$	Set to 1 if vehicle v is stationed at station s initially.
$F_{s,s'}^{t,k}$	The expected number of customers at time step t going out from station s and wanting to reach station s' at time step $t+k$.
$R_{s,s'}^{t,k}$	The revenue obtained if a bike is hired at time step t from station s and is returned at station s' after k time steps.
$P_{s,s'}$	The routing cost for any vehicle to travel from station s to s'.
$H_{s,s'}$	The distance between station s and s'.
B	The amount of budget allocated for the repositioning tasks for a given planning period.

Table 2.1: BSS notations and their definition.

2.1.1 Model: DRRP

I now formally describe the generic model for DRRP, that can be used to find a reposition strategy for the bikes using carrier vehicles, while considering the routes of the vehicles and the future demand for multiple time steps. We employ the following tuple:

$$\langle \mathcal{S}, \mathcal{V}, \mathbf{C}^{\#}, \mathbf{C}^{*}, \mathbf{d}^{\#,0}, \mathbf{d}^{*,0}, \{\sigma_v^0\}, \mathbf{F}, \mathbf{R}, \mathbf{P} \rangle$$

A brief description of these elements are shown in Table (2.1). Each station $s \in \mathcal{S}$ has a fixed capacity denoted by $C_s^{\#}$ and each vehicle $v \in \mathcal{V}$ has a fixed capacity denoted by C_v^*. Initial distributions of bikes at station s and vehicle v are given by $d_s^{\#,0}$ and $d_v^{*,0}$, respectively. σ_v^0 denotes the initial distribution of vehicle v at base stations. For ease of notation, we use the generic $\sigma_{v,s}^t$ and set it to 0, if $t > 0$. $\boldsymbol{F}$ represents the expected customer demand which is computed from the historical trip data. In a general case, the revenue depends only on the duration k, but we use the notation,

$R^{t,k}_{s,s'}$ to represent a generic model[1]. We further note that this revenue parameter can also be used to represent the monetary value of the social and environmental benefits associated with the usage of bike. Finally, $P_{s,s'}$ denotes the routing cost for any vehicle to travel from station s to s' which depends on the distance between the two stations and the fuel cost. This cost parameter is then multiplied by a predefined factor to incorporate other relevant operating costs.

The goal is to maximise the overall profit which provides a trade-off between minimising lost demand and reducing routing cost of vehicles. It should be noted that we do not directly minimise the lost demand because that can result in a significant cost of the routing of vehicles.

2.1.2 Model: DRRPU

The generic model for the DRRPU is represented as an extension of DRRP. DRRPU is formally defined using the following tuple:

$$\left\langle \mathcal{S}, \mathcal{V}, \mathbf{C}^{\#}, \mathbf{C}^{*}, \mathbf{d}^{\#,0}, \mathbf{d}^{*,0}, \{\sigma_v^0\}, \mathbf{H}, \mathbf{F} \right\rangle$$

Definitions of all the terms related to vehicles and stations (i.e, $\mathcal{S}, \mathcal{V}, \mathbf{C}^{\#}, \mathbf{C}^{*}, \mathbf{d}^{\#,0}, \mathbf{d}^{*,0}$, $\{\sigma_v^0\}$) for DRRPU remain the same as DRRP. For DRRPU, we employ the distance between station s and s' as $H_{s,s'}$, rather than indirectly considering it through routing cost, $P_{s,s'}$. In contrast to representing the expected future demand, we now consider the underlying uncertainties in the demand as this model is useful when there is a significant variance in the demand. **F** represents the set of demand bounds that is computed from the historical trip data. We compute three types of bounds on the arrival customer demand: (a) $\check{F}^t, \hat{F}^t$ denote the lower and upper bound on the system wide demand across all the stations at time step t; (b) $\check{F}^t_s, \hat{F}^t_s$ denote the bounds on the demand in station s at time step t; (c) $\check{F}^t_{s,s'}, \hat{F}^t_{s,s'}$ denote the bounds

[1]Such a generic model can for instance be used to capture higher price of hiring bikes in central business districts.

on the demand that arises in station s at time step t and reach station s' at time step $t+1$. These demand bounds are used in the solution approach to generate the strategy. On the other hand, the strategies are evaluated on a wide range of testing demand scenarios that are created using Poisson distribution and these scenarios are not forced to follow the bounds used in the decision making process.

2.1.3 Model: DRRPT

In this section, I formally describe the generic model of DRRPT as an extension of the DRRP model. DRRPT is compactly represented using the following tuple:

$$< \mathcal{S}, \mathcal{V}, \boldsymbol{C^{\#}}, \boldsymbol{C^{*}}, d^{\#,0}, \{\sigma_v^0\}, \boldsymbol{H}, \boldsymbol{F}, B >$$

All the terms related to stations (i.e., $\mathcal{S}, \boldsymbol{C^{\#}}, d^{\#,0}$) remain the same as DRRP. On the contrary to employing carrier vehicles, we now introduce a set of bike trailers for the repositioning task to minimise the carbon footprint. We have a set of bike trailers $\mathcal{V}$ where C_v^* denotes the number of bike slots in the trailer $v \in \mathcal{V}$. σ_v^0 symbolises the initial locations of the trailers, i.e., $\sigma_v^0(s)$ is fixed to 1 if trailer v is stationed at s initially and 0 otherwise. $\boldsymbol{H}$ denotes a two-dimensional matrix that stores the relative distance between each pair of stations. $\boldsymbol{F}$ represents a set of K discrete training demand scenarios. Specifically, $F_{s,s'}^k$ denotes the demand for the planning period for scenario k that arises at station s and reaches station s' in the next time step. Finally, B denotes the amount of budget allocated for the repositioning tasks for a given planning period.

For a better understanding of the differences among DRRP, DRRPU and DRRPT, we summarise the key features of these three BSS models in Table (2.2).

	DRRP	DRRPU	DRRPT
Characteristic	Dynamic repositioning of bikes to reduce lost demand for consistent demand patterns	Robust repositioning of bikes to reduce lost demand for highly uncertain demand scenarios	Repositioning of bikes using trailers to reduce lost demand and carbon emission.
Objective	Maximise profit by trading off between reducing lost demand and minimising routing cost of vehicles	Minimise worse case lost demand by considering the possible uncertainties in underlying demand	Generate repositioning tasks for trailers and crowdsource them to users within a central budget
Mode of repositioning	Carrier Vehicles	Carrier Vehicles	Bike Trailers
Input demand scenarios	Expected demand for multiple time steps	Set of demand bounds	Set of demand scenarios
Online/ Offline	Offline	Online	Online
Crowdsourcing	✗	✗	✓

Table 2.2: Features of different BSS models.

2.2 Related Work on Decision Making in BSS

Given the practical benefits of bike sharing systems, they have been studied extensively in the literature. In a recent survey, Laporte, Meunier, and Wolfler Calvo (2015) summarise the leading contributions in shared transportation systems in terms of strategic decision making (e.g., location of stations and sizes of fleet) and operational decision making (e.g., vehicle repositioning). To situate these contributions in the context of this thesis, we summarise along five threads. These five threads are described in the first five subsections. Furthermore, the broad problem we considered in Chapter 3 is one of sequential decision making in the presence of uncertainty. This problem is considered extensively in the Artificial Intelligence (AI) literature through Markov decision processes (MDPs) and its variants, stochastic network design in sustainability applications, and others. Thus, in Section 2.2.6, we summarise other research relevant to our methodological contributions, i.e., decomposition and abstraction in sequential decision making under uncertainty.

2.2.1 Designing a BSS

The first thread of research focuses on how to design a BSS. Kumar and Bierlaire (2012) provide a linear regression model to learn the correlation between station locations and customer's locations. They use customer demographics and personal information to identify the best locations for the placement of base stations. Martinez et al. (2012) provide an MILP formulation to optimally locate the stations, and solve it using a branch and bound algorithm. Lin and Yang (2011) and Lin et al. (2013) propose a decision support model to design a BSS. In addition, they provide a model that incorporates the service level requirements of BSS by employing constraints from the inventory management literature.

The focus in this thread of research is on strategic decision making and hence is complementary to the focus of our work, which is on operational decision making (day-to-day operations).

2.2.2 Static Redistribution

The second thread of research focuses on the rebalancing problem in static case where the movements of bikes during the repositioning period are negligible. Specifically, it focuses on the problem of finding routes for a fixed set of vehicles to reposition bikes at the end of the day or when the movements of bikes by customers are insignificant to achieve the desired configuration of bikes across the base stations. This problem is also known as the Static Bicycle Repositioning Problem (SBRP).

Benchimol et al. (2011) present an approximate solution which is inspired by the solution of *C-delivery TSP* (Chalasani & Motwani, 1999) to solve the static rebalancing and routing problem with a single vehicle. In a similar direction, Chemla et al. (2013) solve the static rebalancing problem for a single vehicle by employing a branch and cut algorithm that can solve a large-scale problem with more than a hundred stations. However, in the presence of multiple vehicles, these solution approaches are not very effective as the routing solution of the first vehicle controls the

routing solution of other vehicles and it affects the overall solution quality. Raviv and Kolka (2013), Raviv, Tzur, and Forma (2013), and Rainer-Harbach, Papazek, Hu, and Raidl (2013) address this concern and provide scalable exact and approximate solution approaches by introducing a set of MILP formulations to solve the SBRP with multiple vehicles. By employing an objective function that penalises unavailability of bikes or empty docks, they find solutions for the vehicles to reposition the bikes in a static manner. Raidl, Hu, Rainer-Harbach, and Papazek (2013), Di Gaspero, Rendl, and Urli (2013), and Di Gaspero, Rendl, and Urli (2016) propose approximate solutions for solving the SBRP using variable neighbourhood search based heuristics. Erdoğan, Laporte, and Wolfler Calvo (2014) present an integer programming formulation to solve the SBRP with demand intervals for a single vehicle. This is an empirically harder problem than the SBRP and they provide a Benders decomposition scheme for solving the problem efficiently. Unfortunately, these solutions are not suitable for solving the dynamic repositioning problems, as the movements of bikes during the planning period make the static solutions irrelevant.

Schuijbroek et al. (2017) propose a scalable approximate solution for the SBRP. They cluster base stations using a maximum spanning star (MAXSPS) approximation and allocate one vehicle to each of the clusters so that the service level requirements are satisfied. In addition, they represent this problem as a clustered vehicle routing problem (Battarra, Erdoğan, & Vigo, 2014). They assume that the movements of bikes by customers during the repositioning period are negligible. However, an online version of their approach can easily be employed to solve the dynamic repositioning problem using a rolling horizon framework that generates a routing and repositioning solution for each time step. In fact, in Chapter 3 and 4, we provide a comparison of our approaches against their approach, which can be considered as a representative of the ad-hoc and myopic heuristic. As demonstrated in our experimental results, such a myopic repositioning solution can significantly falter as it does not consider the future demand surges.

Our approach differs from approaches mentioned in this section, as we consider the repositioning of bikes in the presence of bike movements by customers during the rebalancing period. Such bike movements can make planned static repositioning irrelevant when customers return the required number of bikes to a station or counterproductive when vehicles pick up bikes from a station where customers need bikes. In addition, a static solution typically cannot adequately capture the surges in customer demand even if they are predictable.

2.2.3 Dynamic Redistribution

Since service level requirements in a BSS change over time due to involuntary movements of customers, static solutions can significantly falter when employed for performing operational planning in BSS. Thus, the third thread of research focuses on dynamical redistribution of bikes based on the assessment of demand.

Pfrommer et al. (2014) provide a myopic repositioning and routing solution for individual vehicles based on the assessment of demand for the next 30 minutes. In case of multiple vehicles, they employ a greedy approach, where the solutions for the vehicles are determined sequentially one at a time. Contardo et al. (2012) propose a dynamic repositioning model to deal with the loss in customer demand in rush hours. They provide a myopic repositioning solution by considering the current demand that was recently observed before the repositioning decisions are made. Furthermore, they employ Dantzig-Wolfe and Benders decomposition techniques to speed up the solving process. However, due to the complex structure of the problem, there was a significant gap between their solution and its lower bound. Recently, Lowalekar, Varakantham, Ghosh, Jena, and Jaillet (2017) propose a scalable online repositioning solution using multi-stage stochastic optimisation and online anticipatory algorithms. However, due to the fact that these myopic solutions do not capture the future expected demand for multiple time steps in the planning phase, they usually provide poor quality solutions when the demand is dynamic (as

observed in both our data sets).

Shu et al. (2010) and Shu et al. (2013) overcome this issue by considering the future expected demand for the entire planning horizon. They predict the stochastic demand from user trip data of Singapore metro system using a Poisson distribution. They provide an optimisation model that suggests a set of locations for the stations and propose a dynamic repositioning model to minimise the number of unsatisfied customers. However, they assume that repositioning of bikes from one station to another is always possible without considering the routing of vehicles.

Our approach (presented in Chapter 3) for solving the DRRPs differs from this thread of research as we consider the dynamic repositioning of bikes in conjunction with the routing of all the vehicles. In addition, we also consider multi-step expected demand that can account for demand surges at later time steps.

2.2.4 Incentivising Customers and Utilising Trailers for Bike Redistribution

The fourth thread of research focuses on incentivizing customers and utilising trailers for rebalancing the system. There has been existing research (Singla et al., 2015; Pfrommer et al., 2014) that present pricing mechanisms to provide incentives to users for assisting with repositioning. Pfrommer et al. (2014) propose the notion of delivering additional incentives to the customers if they return their bikes to the neighbouring starving station rather than submitting it to their original destination station. Singla et al. (2015) propose a pricing mechanism that dynamically calculates the incentive values for each neighbouring station and offers the corresponding incentive amounts to the users through a mobile application. There has been a wide spectrum of papers on car sharing (Chow & Yu, 2015; Mareček, Shorten, & Yu, 2016) that present pricing mechanisms to provide incentives to users for assisting with rebalancing the system. However, this line of work has primarily focused on individual bikes (without trailers) and has taken a myopic (individual station) view

on whether a bike is required at a station. Furthermore, unlike car sharing (Chow & Yu, 2015), the BSS operators cannot order users based on their utility and operate within tight budget constraints. In our approach presented in Chapter 5, we provide an end to end system that takes the global view (all stations) of the repositioning requirements and incentives their execution within the budget constraints.

On the other hand, O'Mahony and Shmoys (2015) predict the service level requirements for base stations in rush hours and introduce the notion of repositioning with bike trailers, by matching each trailer to its suitable producer and consumer stations, based on the assessment of inventory state of the base stations. However, they assume that all the tasks for the trailers can be achieved with dedicated staff which is not an economically viable option. In contrast, we propose an optimisation model in Chapter 5 to generate the repositioning tasks for trailers by considering the near future demand over a set of training demand scenarios. Furthermore, we design a mechanism to crowdsource the tasks to the users while ensuring the given budget constraints.

2.2.5 Demand Prediction and Analysis in BSS

The fifth thread of research which is complementary to the research presented in this thesis is on demand prediction and analysis. Borgnat, Abry, Flandrin, and Rouquier (2009) and Borgnat, Abry, Flandrin, Robardet, Rouquier, and Fleury (2011) propose the idea of predicting temporal user demand from the past data and providing forecasted information to users. Froehlich, Neumann, and Oliver (2008) and Lathia, Ahmed, and Capra (2012) predict the user demand in terms of available bikes or normalised available bikes in a station at a certain time period. Rudloff and Lackner (2013) provide demand forecasting model based on the weather and the state of the neighbouring stations. O'Mahony and Shmoys (2015) predict the service level requirements for each station in rush hours by analysing the data of *Citibike* in New York City. Furthermore, they provide an optimisation model to minimise

the maximum imbalance such that users are not too far from bikes or available docks. Specifically, all the above mentioned papers provide data-driven analysis of the system-wide demand in BSSs.

In contrast to the data-driven analysis of the demand, Nair and Miller-Hooks (2011) and Nair, Miller-Hooks, Hampshire, and Bušić (2013) provide a theoretical analysis of the service level of a BSS using dual-bounded joint-chance constraints. They predict the near future demand for a short period and ensure that the system-wide demand is served with a certain probability. While these insights are practical and useful in demand prediction, they are not applicable for large systems. Leurent (2012) represents the bike sharing system as a dual Markovian waiting system to predict the actual demand. In contrast, we assume that the users are impatient and leave the system if they encounter an empty station. Raviv et al. (2013) and Schuijbroek et al. (2017) represent a BSS as a queueing network with Markov assumptions. The pickup or drop-off of bikes by the users are represented as random variables with a known probability distribution. While these assumptions hold for one step or short term planning, it becomes intractable for multi-step or long-term planning due to the time-varying nature of the demand and the inter-dependency between the pickup and drop-off rates between stations at consecutive time steps.

Given its wide ranging applicability and accuracy, Poisson distributions have been extensively used in the literature to represent random arrival processes. It has also been used regularly to represent customer arrivals at base stations in BSSs (George & Xia, 2011; Shu et al., 2010, 2013; Kabra, Belavina, & Girotra, 2015). Due to its simplicity and accuracy in representing customer arrival processes, we also represent the demand arrival process at each of the stations at each time step as a Poisson distribution.

2.2.6 Abstraction and Decomposition in Sequential Decision Making Problems under Uncertainty

This thread of research is relevant to our two main methodological contributions in Chapter 3, namely, abstraction and dual decomposition. Both these general methods have been applied in various types of sequential decision making problems in the literature. Our work is focused on multi-step matching of demand and supply, which has similarities with well-studied models in sequential decision making. Since we consider optimising a reward, the relevant sequential decision making model would be Markov decision process (MDP). In fact, by considering every possible match of supply and demand as a potential state and modifications to each state as actions, we can represent a DRRP as an MDP. The corresponding MDP can be represented using following tuple $\langle S, A, P, R \rangle$:

- The state space, S of the MDP needs to incorporate all the possible combinations of inventory levels of the stations. Furthermore, to capture the physical limitations of vehicle routes and the number of bikes they can pick up or drop off from their origin station, the possible inventory levels and locations of the vehicles need to be incorporated through the state representation.

- Any changes in the inventory state of the stations by the vehicles can be represented through the action space, A of the MDP.

- The transition probabilities, P can be represented as the prior probabilities of moving bikes by the customers at a given state, according to the expected customer demand.

- The reward, R can be represented as a function of a given state at a time step and the expected customer demand at that time step.

However, since the number of possible matches grows exponentially with the number of stations, bikes and vehicles, the number of possible matches is extremely

large[2]. Thus, it would be difficult to even specify the problem. Even if we are able to specify the problem, abstraction of state space for the MDP of a corresponding DRRP will automatically abstract the action space. As outlined in the next two paragraphs, existing abstraction techniques only consider state abstraction where action space remains the same, so they cannot be employed to solve the large-scale MDP representations of DRRPs.

State abstraction has been widely adopted in Artificial Intelligence (Giunchiglia & Walsh, 1992) and Operations Research (Rogers, Plante, Wong, & Evans, 1991). Spatio-temporal abstraction (Sutton, Precup, & Singh, 1999; Mahadevan, 2002) is studied heavily in reinforcement learning (RL) literature, which is also a sequential decision making problem under uncertainty. Furthermore, abstracting time periods is widely used in the inventory routing problem (IRP) literature in order to simultaneously take into account the inventory planning and vehicle routing constraints (e.g., see Coelho, Cordeau, & Laporte, 2013; Papageorgiou, Nemhauser, Sokol, Cheon, & Keha, 2014).

Li, Walsh, and Littman (2006) propose five methods to perform state aggregation in MDPs while preserving the useful information that is critical for solving the complete problem. However, they assume that the action set does not change after abstraction, so the actions and policies of the original MDP and the abstract MDP are comparable. On the contrary, both the actions and states (when the DRRP is represented as an MDP) would change in our case, as the actions correspond to moving vehicles to a base station and the number of base stations changes with abstraction. Therefore, the insights of lossless abstraction are not applicable in our case. Given the extremely large-scale, we have to abstract at the level of state features and not at the level of individual states, so that is another differentiating factor from the work by Li et al. (2006). Abstracting a relevant set of features for state abstraction has been employed in both the offline (Sturtevant & White, 2006) and

[2]For *Capital Bikeshare*, we have 300 stations, each with a capacity of 20 and 5 vehicles, each with a capacity of 20, so there would be approximately $20^{300} \times 20^5 \times 300^5$ states and $(300 \times 20)^5$ actions.

online (Geramifard, Doshi, Redding, Roy, & How, 2011) settings for solving MDP and RL problems. These approaches focus on function approximation based on abstract state features and assume that the action space remains the same after abstraction. However, in our case, the abstraction of state features will also alter the action space. Due to this reason, these existing approaches cannot be directly applied for solving the DRRP. Furthermore, these existing approaches are suitable for problems with either low dimensional state space or with binary features. However, an MDP corresponding to a DRRP will have a large feature set and each feature can have a large number of values. As a consequence, the existing approaches will not be scalable for solving the DRRP.

Abstraction employed in Chapter 3 is similar to the work of Knoblock (1993). Specifically, their method (Knoblock, 1991) consists of three steps: (1) identify the abstract problem from the original problem, (2) solve the abstract problem, and (3) use the solution from the abstract problem to determine the solution for the original problem. The abstraction mechanism we employ in Chapter 3 to group the base stations is inspired by the recent work of Lu and Boutilier (2015). They use an abstraction technique to solve the multi-campaign, multi-channel marketing optimisation problem (MMMOP) by dynamically segmenting the customer population into a number of groups and use the solution of the abstract problem to guide the global solution. In a similar way, we group the nearby base stations into an abstract station because the customers using those stations have similar movement dynamics in aggregation and nearby stations have similar patterns of imbalance as shown in Section 3.4.

Dual decomposition has been employed in the literature to speed up the solution of sequential decision making problems. Dean and Lin (1995) use a decomposition technique to solve large-scale MDPs. They decompose the original problem into a number of subproblems which are solved independently and finally combine the solutions to determine the solution of the original problem. Furmston and Barber (2011) employ Lagrangian dual decomposition to decompose a stationary policy

finite-horizon MDP into a series of unconstrained MDPs that can be solved easily and use these solutions to guide the solution of the master problem in an iterative fashion. Recently, dual decomposition techniques have been employed successfully for solving sequential decision making problems such as factored MDPs (Guestrin & Gordon, 2002), spatial conservation planning (Kumar, Wu, & Zilberstein, 2012) and contact center planning (Kumar, Singh, Gupta, & Parija, 2014) problems. In Chapter 3, we adopt a dual decomposition framework to exploit the weak dependency between two critical components of the problem, namely, repositioning of bikes and routing of vehicles.

2.3 Details of Referred Solution Approaches

In this section, we present some of the existing approaches for solving the operational decision making problems in BSSs. We have compared our approaches with these existing benchmarks in the later parts of this document.

2.3.1 Static Redistribution in BSS

Static Repositioning implies the practice of no repositioning during the day. The stations are rebalanced at the end of the day to achieve a predefined inventory level. We use this as a baseline approach where no repositioning is done during the decision making period. For this static approach, we simulate the flows of bikes according to customer demand using the below mentioned simulation model to evaluate the performance of the approach.

2.3.1.1 Simulation Model

Similar to the work of Shu et al. (2013), we also evaluate the performance of relevant approaches by using a simulation that is based on the past data. Once the repositioning and routing solutions are determined for a time step, we execute them on the simulator for evaluating their performance on the realized demand scenario

for that particular time step. The customer demand computed in the data, $F^{t,k}_{s,s'}$, denotes the number of customers who want to hire bikes from station s at time step t and return at station s' after k time steps[3]. Let $d^{\#,t}_s$ represents the number of bikes present in station s at time step t after the repositioning tasks are completed. The flow of bikes between the stations is determined based on the following two cases: (a) If the arrival demand at a station is less than the number of bikes present in that station, then all the customers are able to hire bikes; (b) If the arrival demand at a station is higher than the number of bikes present in that station, then the actual flow of bikes between station s and s' (denoted as $x^{t,k}_{s,s'}$) is computed using the aggregated transition probability (i.e., $\frac{F^{t,k}_{s,s'}}{\sum_{k,s'} F^{t,k}_{s,s'}}$) that is observed in the data (courtesy: Shu et al., 2013).

$$x^{t,k}_{s,s'} = \left\{ \begin{array}{ll} F^{t,k}_{s,s'} & \textbf{if } \sum_{k,\tilde{s}} F^{t,k}_{s,\tilde{s}} \leq d^{\#,t}_s \\ \frac{F^{t,k}_{s,s'}}{\sum_{k,\tilde{s}} F^{t,k}_{s,\tilde{s}}} \cdot d^{\#,t}_s & \textbf{Otherwise} \end{array} \right\} \tag{2.1}$$

Let, $Y^{+,t}_s$ and $Y^{-,t}_s$ denote the number of bikes picked up and dropped off at station s by the vehicles or trailers at time step t. Once the actual flow of bikes by the customers at time step t is determined from equation (2.1), we calculate the distribution of bikes in station s at time step $t+1$ as the sum of un-hired bikes at time step t, net incoming bikes in station s at the beginning of time step $t+1$ and the net drop-off bikes by vehicles or trailers at time step $t+1$.

$$d^{\#,t+1}_s = d^{\#,t}_s + \Big[\underbrace{\sum_{k,\tilde{s}} x^{t-k,k}_{\tilde{s},s} - \sum_{k,s'} x^{t,k}_{s,s'}}_{\text{Net incoming bikes}} \Big] + \Big[\underbrace{Y^{-,t+1}_s - Y^{+,t+1}_s}_{\text{Net drop-off bikes}} \Big] \tag{2.2}$$

For the static reposition approach, we set the repositioning numbers (i.e., $\mathbf{Y}^+, \mathbf{Y}^-$) to 0. The number of bikes in station s at time step $t+1$, $d^{\#,t+1}_s$ (computed using

[3]We use the notation $F^{t,k}_{s,s'}$ for a generic representation of the customer demand. However, in the context of online decision making (e.g., DRRPU and DRRPT), a valid assumption is that the customers complete their trips within one time step and therefore, we can either remove the index k or fix the value of k to 0.

equation 2.2) does not take into account the station capacity constraints. To handle such boundary conditions and to ensure the capacity constraints are considered for the stations, we transfer extra bikes (i.e., $d_s^{\#,t+1} - C_s^{\#}$) to the nearest available station if $d_s^{\#,t+1}$ exceeds the station capacity, $C_s^{\#}$. These extra numbers are considered as the lost demand at the time of return. Once the distribution of bikes across the stations for time step $t+1$ is obtained, we utilise this information to compute the repositioning strategy for trailers for time step $t+1$. This iterative process continues until we reach the last decision epoch.

2.3.2 Myopic Redistribution in BSS

Myopic Repositioning entails that bikes are repositioned at each time step to reach a certain inventory level. Through the personal communication with bike sharing operators, we infer that 50% of the station capacity is the ideal inventory level and some operators rebalance the stations in a myopic fashion (without considering the demand patterns) to reach that specific inventory level. Table (2.3) provides the MILP formulation for the myopic approach that is used in each time step to generate a repositioning solution. As a vehicle can visit multiple stations within one time step, the time indicators are used in SOLVEMYOPIC() to represent the sequence of moves of all the vehicles.

The objective function (2.3) is to minimise total routing cost for all the vehicles. As a vehicle cannot visit all the stations within the rebalancing period, additional slack variables δ are added in the objective to ensure that maximum number of stations are balanced. w represents the unit penalty for deviating from the desired inventory level of the station and we set it to 1 in our experiments. s_i^0 is the initial distribution of bikes at station i in that time step. Our goal is to rebalance the system to reach a pre-configured inventory level $\alpha C_i^{\#}$ for station i. We set the value of α to 50% in the experiments. Let $z_{i,j,v}^t$ denote the binary decision variable that is set to 1 if a vehicle v travels from station i to j at time step t. $y_{i,v}^{+,t}, y_{i,v}^{-,t}$ denote the number of

$$\min \sum_{i,j,v,t} P_{i,j} z^t_{i,j,v} + w \sum_i \delta_i \tag{2.3}$$

$$\textbf{s.t.} \quad s^0_i + \sum_{t,v} (y^{-,t}_{i,v} - y^{+,t}_{i,v}) \geq \alpha C^{\#}_i - \delta_i, \qquad \forall i \tag{2.4}$$

$$s^0_i + \sum_{t,v} (y^{-,t}_{i,v} - y^{+,t}_{i,v}) \leq \alpha C^{\#}_i + \delta_i, \qquad \forall i \tag{2.5}$$

$$\sum_j z^t_{i,j,v} - \sum_j z^{t-1}_{j,i,v} = \sigma^t_{v,i}, \qquad \forall i,t,v \tag{2.6}$$

$$y^{+,t}_{i,v} + y^{-,t}_{i,v} \leq C^*_v \cdot \sum_j z^t_{i,j,v}, \qquad \forall i,t,v \tag{2.7}$$

$$d^{*,t}_v + \sum_i (y^{+,t}_{i,v} - y^{-,t}_{i,v}) = d^{*,t+1}_v, \qquad \forall t,v \tag{2.8}$$

$$s^0_i + \sum_{t,v} (y^{-,t}_{i,v} - y^{+,t}_{i,v}) \geq 0, \qquad \forall i \tag{2.9}$$

$$s^0_i + \sum_{t,v} (y^{-,t}_{i,v} - y^{+,t}_{i,v}) \leq C^{\#}_i, \qquad \forall i$$

(2.10)

$$0 \leq y^{+,t}_{i,v}, y^{-,t}_{i,v} \leq C^*_v, 0 \leq d^{*,t}_v \leq C^*_v, \delta_i \geq 0, z^t_{i,j,v} \in \{0,1\} \tag{2.11}$$

Table 2.3: SOLVEMYOPIC($\mathbf{s}^0, drrp$)

bikes picked up and drop off at station i by the vehicle v at time step t, respectively. $d^{*,t}_v$ denotes the number of bikes present at vehicle v at time step t.

As all the stations cannot be balanced within one time step, constraints (2.4)-(2.5) ensure that a station i is maximum δ_i number of bikes away from the desired configuration of $\alpha C^{\#}_i$ and in the objective we ensure that the value of δ is minimised. Constraints (2.6) enforce that the flows of vehicles in and out of stations are preserved. Constraints (2.7) ensure that the flows of bikes in and out of vehicles are preserved at the time of repositioning. Constraints (2.8) enforce that a vehicle can only pick up or drop off bikes at a station if it is present at that station. Lastly, Constraints (2.9)-(2.10) ensure that the station and vehicle capacity are not exceeded while repositioning the bikes.

In each time step, given the distribution of bikes, we find the repositioning solution by solving the MILP provided in Table (2.3). After the repositioning, we update the number of bikes in each station and simulate the flows of bikes accord-

ing to customer demand. Once we determine the flows of bikes, we can compute the distribution of bikes in each station for the next time step and this information is used to execute the MILP of Table (2.3) for the next time step. The process iterates until we reach the last time step.

2.3.3 Online Redistribution of Bikes Based on the Model of Schuijbroek et al. (2017)

To the best of our knowledge, we are not aware of any research paper that formally presents heuristics to solve the operational decision making problem in BSS online. The method that can be employed online to reposition bikes can be adapted from the algorithm provided by Schuijbroek et al. (2017). This approach can be executed online with the assumption of negligible movements of bikes by customers. Therefore, in this section, we provide the details of the model used by Schuijbroek et al. (2017) and also how we have adapted the model to solve the problem in our study. The primary goal of Schuijbroek et al. (2017) is to minimise the operational cost for the routing of vehicles such that the whole system can be balanced. The key aspects of their approach are as follows:

- Customer movements at the time of rebalancing operation are negligible.
- Vehicles can visit all the stations to rebalance the whole system within the rebalancing period.
- Each vehicle is assigned to a group of base stations. The entire system is divided into a set of cluster (the number of clusters is equal to the number of vehicles), each of which is allocated with a vehicle. Thus, a vehicle is only responsible for the repositioning of bikes within a particular cluster and it can visit all the base stations of that cluster within the rebalancing period.

With minor changes we are able to adapt their approach to the dynamic repositioning context. Intuitively, we make changes corresponding to the first two points

above and leave the rest of the approach as it is:

- Because of the dynamism and significant customer movements during the rebalancing period, the demand model assumed by Schuijbroek et al. (2017) is not valid in our context. Because of the different demand model assumptions, their approach to compute the inventory level is not applicable for solving the problems like DRRP. However, we figure out the best inventory levels (number of required bikes at stations) through experimentation. More specific details will be mentioned later.

- Since the assumption of visiting all stations to rebalance in one time step is not reasonable, we set a maximum number of stations that can be rebalanced in one time step. Specifically, we set it to 5 stations (any 5 stations can be visited in one time step) and this corresponds to an average distance travelled by a vehicle in one time step to be approximately 3.4 kilometers (which is slightly above the distance travelled with our approach which is around 2.6 kilometers).

$$\min \quad \hat{H} \tag{2.12}$$

$$\textbf{s.t.} \quad \sum_{v \in \mathcal{V}} z_{i,v} = 1, \qquad \forall i \in \{\mathcal{S} \setminus S^{+}\} \tag{2.13}$$

$$\sum_{v \in \mathcal{V}} z_{i,v} \leq 1, \qquad \forall i \in S^{+} \tag{2.14}$$

$$q_v^0 + \sum_{i \in S} s_i^0 z_{i,v} \geq \sum_{i \in S} s_i^{min} z_{i,v}, \qquad \forall v \in \mathcal{V} \tag{2.15}$$

$$-(C_v^* - q_v^0) + \sum_{i \in S} s_i^0 z_{i,v} \leq \sum_{i \in S} s_i^{max} z_{i,v}, \qquad \forall v \in \mathcal{V} \tag{2.16}$$

$$\hat{h}_v \geq \sum_{j \in \mathcal{S}} d_{i,j}(z_{i,v} + z_{j,v} - 1), \qquad \forall s \in \mathcal{S}, v \in \mathcal{V} \tag{2.17}$$

$$\hat{h}_v \geq \hat{H}, \qquad \forall v \in \mathcal{V} \tag{2.18}$$

$$z_{i,v} \in \{0,1\}, \hat{h}_v \geq 0, \hat{H} \geq 0 \tag{2.19}$$

Table 2.4: MAXSPS-BASED-CLUSTERING($\mathbf{s}^0, drrp$)

We now describe the details of the online heuristic approach. To generate the cluster of stations for each vehicle, a maximum spanning star (MAXSPS) based approximation technique is employed. The idea is to minimise the maximum expected routing cost $\hat{H}$ (delineated in Expression 2.12) such that the service level requirement for each station can be satisfied. An optimisation problem (described in Table 2.4) is solved to discover the cluster of stations for each vehicle. Let, $z_{i,v}$ be a binary decision variable that is set to 1 if station $i \in \mathcal{S}$ is assigned to the cluster of vehicle $v \in \mathcal{V}$. Let, S^+ denotes the set of self-sufficient stations and $\hat{h}_v$ denotes the routing cost for vehicle v. Constraints (2.13)-(2.14) ensure that insufficient stations ($\mathcal{S} \setminus S^+$) must be visited by vehicles while sufficient stations can be visited if required. Constraints (2.15)-(2.16) ensure that each cluster contains enough bikes such that service level requirement can be satisfied for each station. Constraints (2.17) estimate the lower bound on the routing cost for the resulting assignment and constraints (2.18) enforce that the objective value of makespan is equivalent to $\max_v \hat{h}_v$.

Table (2.5) provides the MILP formulation for the online heuristic approach that is used in each time step to generate a repositioning solution. As a vehicle can visit multiple stations within one time step, the time indicators are used in SOLVEONLINE() to represent the sequence of moves of all the vehicles. The objective function (2.20) is to minimise the maximum routing cost for all the vehicles. As a vehicle cannot visit all the stations within the rebalancing period, we add additional slack variables δ^+, δ^- in the objective to ensure that maximum number of stations are balanced. w represents the unit penalty for deviating from the minimum and maximum number of bikes required at each station and we set it to 1 in our experiments. s_i^0 is the initial distribution of bikes at station i in that time step. s_i^{min}, s_i^{max} represent the lower and upper bounds, respectively, on the number of bikes required at station i. Let, F_i^t is the expected demand at station i in the time step t, then s_i^{min}, s_i^{max} are determined as $(1-\epsilon)F_i^t$ and $(1+\epsilon)F_i^t$, respectively, where ϵ is the tolerance level[4].

[4]We take the value of ϵ as 10% in the experiment.

$$\min H + w\big(\sum_i \delta_i^+ + \sum_i \delta_i^-\big) \tag{2.20}$$

$$\textbf{s.t.}\quad s_i^0 + \sum_{t,v}(y_{i,v}^{-,t} - y_{i,v}^{+,t}) \geq s_i^{min} - \delta_i^+, \qquad \forall i \tag{2.21}$$

$$s_i^0 + \sum_{t,v}(y_{i,v}^{-,t} - y_{i,v}^{+,t}) \leq s_i^{max} + \delta_i^-, \qquad \forall i \tag{2.22}$$

$$\sum_j z_{i,j,v}^t - \sum_j z_{j,i,v}^{t-1} = \sigma_{v,i}^t, \qquad \forall i,t,v \tag{2.23}$$

$$\sum_{t,j\notin G_v} z_{i,j,v}^t = 0, \qquad \forall i,v \tag{2.24}$$

$$y_{i,v}^{+,t} + y_{i,v}^{-,t} \leq C_v^* \cdot \sum_j z_{i,j,v}^t, \qquad \forall i,t,v \tag{2.25}$$

$$d_v^{*,t} + \sum_i (y_{i,v}^{+,t} - y_{i,v}^{-,t}) = d_v^{*,t+1}, \qquad \forall t,v \tag{2.26}$$

$$H \geq \sum_{i,j,t} P_{i,j} \cdot z_{i,j,v}^t, \qquad \forall v \tag{2.27}$$

$$s_i^0 + \sum_{t,v}(y_{i,v}^{-,t} - y_{i,v}^{+,t}) \geq 0, \qquad \forall i \tag{2.28}$$

$$s_i^0 + \sum_{t,v}(y_{i,v}^{-,t} - y_{i,v}^{+,t}) \leq C_i^{\#}, \qquad \forall i \tag{2.29}$$

$$0 \leq y_{i,v}^{+,t}, y_{i,v}^{-,t} \leq C_v^*, 0 \leq d_v^{*,t} \leq C_v^*, H, \delta_i^+, \delta_i^- \geq 0, z_{i,j,v}^t \in \{0,1\} \tag{2.30}$$

Table 2.5: SOLVEONLINE($\mathbf{s}^0, drrp$)

A vehicle v is allocated to the cluster G_v.

Constraints (2.21) ensure that each station has at least the minimum required bikes after the repositioning. As all the stations cannot be balanced within one time step, the slack variable δ^+ is added in these constraints to avoid the infeasibility of the MILP. Constraints (2.22) ensure that the number of bikes at each station does not exceed the maximum required number of bikes after the repositioning and the slack variable δ^- is added to these constraints to avoid the above mentioned difficulties. Constraints (2.23) enforce that the flows of vehicles in and out of stations are preserved. Constraints (2.24) ensure that vehicle v can only visit stations within the cluster G_v. Constraints (2.25) ensure that the flows of bikes in and out of vehicles

are preserved at the time of repositioning. Constraints (2.26) enforce that a vehicle can only pick up or drop off bikes at a station if it is present at that station. Constraints (2.27) ensure that the variable H in the objective is greater than the routing cost for each vehicle or alternatively assure that we minimise the maximum routing cost of the vehicles. Lastly, Constraints (2.28)-(2.30) ensure that the station and vehicle capacity are not exceeded while repositioning the bikes.

Similar to the myopic approach, given the distribution of bikes, we find the repositioning solution in each time step by solving the MILP provided in Table (2.5). After the repositioning, we simulate the flows of bikes according to customer demand and compute the distribution of bikes in each station for the next time step.

Chapter 3

Optimising Lost Demand in BSS through Dynamic Redistribution

Bike Sharing Systems (BSSs) offer attractive alternatives to private transportation particularly in alleviating concerns associated with increased carbon emissions, traffic congestion and usage of non-renewable resources. BSSs have the ability to provide healthier living and greener environments while delivering fast movements for customers. A few examples of BSSs are *Capital Bikeshare* in Washington DC, *Hubway* in Boston, *Bixi* in Montreal, *Vélib'* in Paris, *Wuhan and Hangzhou* Public Bicycle in China, etc.

Bike sharing systems are currently adopted in 1,139 cities with a fleet of over 1,445,000 bicycles. In addition, there are 357 cities where BSSs are either in the planning stage or under construction (Meddin & DeMaio, 2016). Figure (3.1) provides a quick view of the bike sharing systems around the world. In a typical bike sharing system, a set of base stations is strategically placed throughout the city. At the beginning of the day, each station is stocked with a pre-determined number of bikes. Users can hire and return bikes from any designated station, each of which has a finite number of docks. Many bike sharing operators use vehicles (e.g., trucks) to reposition bikes at the end of the work day so as to return to a pre-determined configuration. In addition, several bike sharing operators (e.g., *Capital Bikeshare*

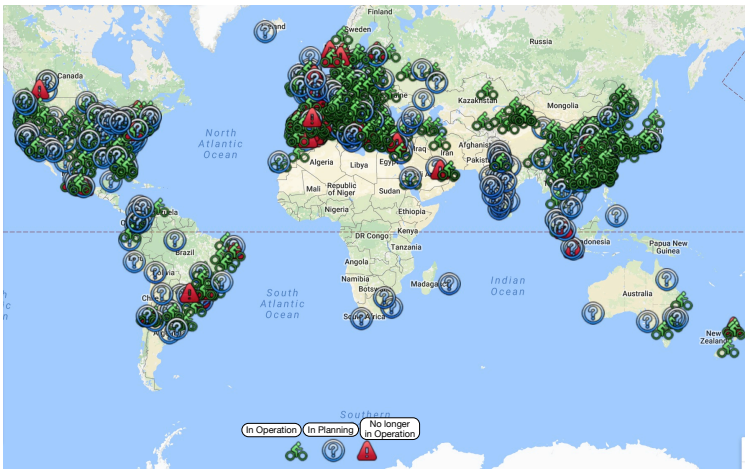


Figure 3.1: Visualisation of the BSSs worldwide (Meddin & DeMaio, 2016).

in Washington DC, *Hubway* in Boston) reposition the bikes during the day using myopic and adhoc methods.

Due to the individualistic movements of customers according to their personal needs, there is often congestion (more than required) or starvation (fewer than required) of bikes at base stations. Figure (3.2) provides the number of instances[1] when stations were empty or full throughout various months in 2013-2014 for *Capital Bikeshare*. 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 were approximately 100 cases of empty stations and 100 cases of full stations per day. At a maximum, there were approximately 750 cases of empty stations and 330 cases of full stations per day. Moreover, in around 40% of instances, the stations were empty or full for more than 30 minutes. In order to tackle this problem, we employ dynamic repositioning of bikes during the day to better match demand with supply. Dynamic repositioning refers to considering movements of bikes by customers (which are usually significant and not negligible) during the repositioning period (Shu et al., 2013). On the other hand, static repositioning refers to ignoring movements of bikes by customers during the rebalancing process (Chemla et al., 2013).

As demonstrated by Fricker and Gast (2016) and our experimental results, star-

[1]The data is taken from *Capital Bikeshare* [http://cabidashboard.ddot.dc.gov/cabidashboard/#Home].

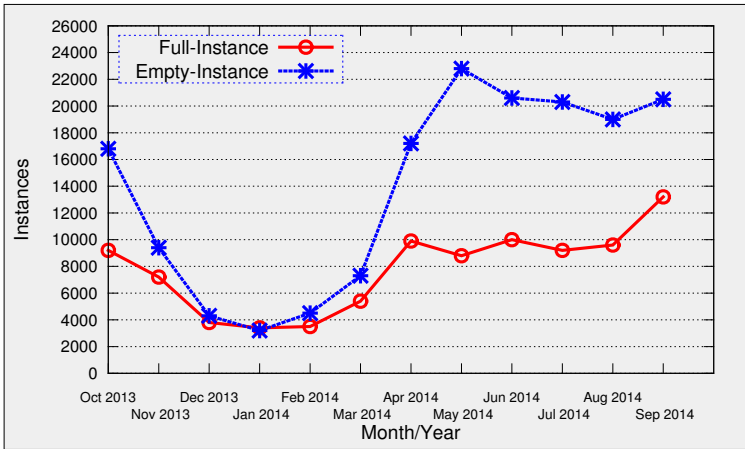


Figure 3.2: Number of empty and full instances of stations in *Capital Bikeshare*.

vation/congestion can result in a significant loss in customer demand. Such loss in demand can have two undesirable outcomes: (1) loss in revenue, and (2) increase in carbon emissions, as people can resort to fuel burning modes of transport. Moreover, some operators (e.g., *Vélib'* in Paris) are even penalised by the local government for loss in customer demand (Schuijbroek et al., 2017). So, there is a practical need to minimise the lost demand and our approach is to dynamically reposition bikes with the help of vehicles while considering future expected demand extracted from past data (Ghosh, Varakantham, Adulyasak, & Jaillet, 2015, 2017). Furthermore, to ensure that the minimisation in demand loss is commercially viable, we consider an objective that is a trade-off between minimising lost demand (alternatively maximising profit) and minimising cost incurred by vehicles.

We refer to the joint problem of bike repositioning and vehicle routing as the Dynamic Repositioning and Routing Problem (DRRP). The DRRP with minor modifications can be used to represent the problem of repositioning empty cars in car sharing systems (e.g., Car2go, Zipcar) (Kek, Cheu, Meng, & Fung, 2009; Barth, Todd, & Xue, 2004), empty vehicles in Personal Rapid Transit (Lees-Miller, Hammersley, & Wilson, 2010) and idle ambulances in emergency response (Yue et al., 2012; Saisubramanian et al., 2015). For example, in the case of car sharing, there is a need to continuously reposition cars to different parking spaces during the day to match with the pattern of demand.

Given the benefits of BSSs and the challenges of setting up such systems to operate efficiently, there have been a wide spectrum of research papers addressing the problem of lost demand and other issues pertinent to it. Some of the major differences between this work and existing research are as follows: (1) we generate routing and repositioning decisions for multiple time steps (e.g., multiple periods during an entire day) using expected demand for bikes at each base station and at each time step, and (2) we employ novel approaches based on decomposition and abstraction to provide scalable solutions to DRRPs for large-scale BSSs.

The DRRP can be considered as a generalisation of the static bicycle repositioning problem (SBRP) (Schuijbroek et al., 2017). The SBRP in turn can be reduced from the 1-PDTSP problem, which is a known NP-Hard problem (Hernández-Pérez & Salazar-González, 2004). Due to this relation, it can be shown that the DRRP is at least NP-Hard. Therefore, we focus on developing principled approximation methods. Our key contributions are as follows:

1. A mixed integer linear program (MILP) formulation to maximise profit for the BSS that considers the trade-off between:
   - maximising served demand, and
   - minimising cost incurred by vehicles.
2. A dual decomposition mechanism to decompose the MILP into two components – one which computes a repositioning solution for bikes and one which computes a routing solution for vehicles.
3. An abstraction mechanism that clusters the base stations in proximity to reduce the size of the problem and further speed up the solution process.
4. Extensive computational results using a simulation based on the real-world data sets of two bike sharing systems, namely, *Capital Bikeshare* (Washington, DC) and *Hubway* (Boston, MA), which demonstrate that our techniques

can significantly reduce lost demand and improve operational efficiency of BSSs.

3.1 Populating DRRP from Data of BSSs

Data Item	Definition
D1	The identification numbers, locations, capacities of base stations and the number of bikes present at each station at the beginning of the day.
D2	Total number of vehicles available for repositioning and capacities of those vehicles.
D3	Customer trip history records.
D4	Revenues associated with successful customer bookings.

Table 3.1: Definition of the data items provided in real-world data sets.

The details of the data items provided by the bike sharing data sets (of two real BSSs), which we use in this work are mentioned in Table (3.1). From these data items, we populate the DRRP tuple $\langle \mathcal{S}, \mathcal{V}, \mathbf{C}^{\#}, \mathbf{C}^{*}, \mathbf{d}^{\#,0}, \mathbf{d}^{*,0}, \{\sigma_v^0\}, \mathbf{F}, \mathbf{R}, \mathbf{P} \rangle$ from Section 2.1.1 as follows:

- Set of base stations, $\mathcal{S}$ and their capacities, $\mathbf{C}^{\#}$ are obtained from data item **D1**.
- Set of vehicles, $\mathcal{V}$ and their capacities, $\mathbf{C}^{*}$ are obtained from data item **D2**.
- $\mathbf{d}^{\#,0}$ is obtained from data item **D1** and $\mathbf{d}^{*,0}$ is set to $\mathbf{0}$ for all vehicles (i.e., vehicles start out empty at the beginning of the day).
- We set the starting positions of vehicles, $\{\sigma_v^0\}$ randomly. Note that we have experimented with different starting configurations, but they do not have a major impact on the results. This is primarily because the positions of vehicles were changed based on the decisions to accommodate the flows of demands after the first time step.

- The demand and transition model, **F** is constructed by aggregating the customer trips for each time step over the data in **D3**. We aggregate the demand for each day of the week, i.e., there is a separate demand model for Mondays, Tuesdays, etc.

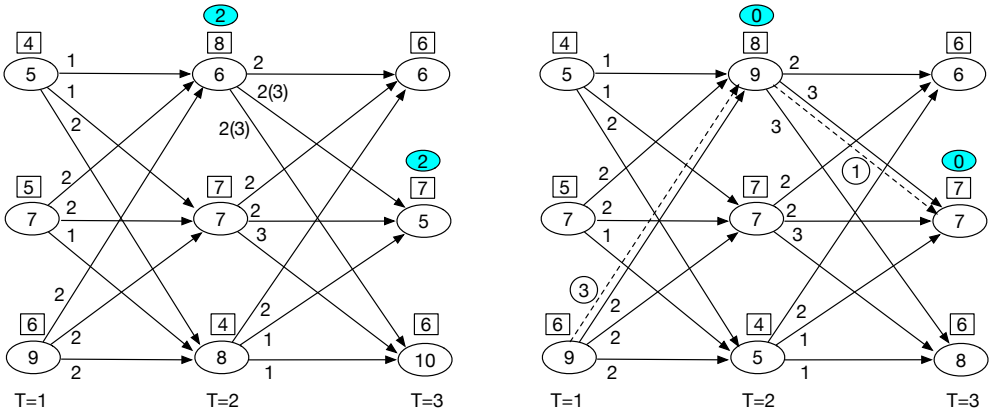
- Revenue model, **R** is constructed from the data item **D4**[2].

- Cost model, **P** is computed based on fuel costs in the location of the bike sharing system in conjunction with distances between base stations that are obtained from the data item **D1**.

In Figure (3.3), we provide an example to better explain the DRRP.

Example 3.1.1. *For ease of explanation, we take 3 base stations and the movements of bikes between stations are shown over 3 time steps. An oval represents a base station at a time step. The leftmost oval at the top is base station 1 at time step 1, the oval in the middle column at the top is base station 1 at time step 2 and so on. The number inside the oval represents the number of bikes present in the station at that time step. The number in the square box on top of each oval represents the actual demand in that station at that time step. The number on each arc shows the actual flow of bikes on that arc. The blue ellipse on the top of an oval represents the lost demand at that station due to unavailability of bikes. For this specific example, the total loss in demand is 4 as shown in Figure* (3.3a)*. However, if we reposition the idle bikes efficiently with the help of a vehicle (in Figure* (3.3b)*, the routes for the vehicle are shown using a dotted line and the reposition numbers are shown within the circle associated with each dotted line), then there is no lost demand.*

[2]Typically, the first 30 minutes for subscription rides is free and after that an additional charge is applied. In our model, to ensure consistency, we can represent revenue for first 30 minutes as the subscription fee divided by the average number of rides.



(a) (b)

Figure 3.3: An example to explain the need for dynamic repositioning: (a) without repositioning (lost demand = 4), (b) with repositioning (dotted line represents the repositioning solution, lost demand = 0).

3.2 Optimisation Model for Solving DRRP

In this section, we provide an optimisation model for a given DRRP. Specifically, we provide a mixed integer linear program (MILP) that computes a profit maximising repositioning and routing solution. For ease of understanding, the decision and intermediate variables employed in the formulation are described in Table (3.2).

Category	Variable	Definition
Decision	$y_{s,v}^{+,t}$	The number of bikes picked from station s by vehicle v at time step t.
	$y_{s,v}^{-,t}$	The number of bikes dropped at station s by vehicle v at time step t.
	$z_{s,s',v}^{t}$	Set to 1 if vehicle v moves from station s to s' at time step t.
Intermediate	$x_{s,s'}^{t,k}$	The number of bikes moving from station s at time step t to s' at $t+k$ by the customers.
	$d_{s}^{\#,t}$	The number of bikes present in station s at time step t.
	$d_{v}^{*,t}$	The number of bikes present in vehicle v at time step t.

Table 3.2: Decision and intermediate variables.

The MILP for solving the DRRP is presented compactly in Table (3.3). Intuitively, the constraints in the optimisation model ensure that: the flows of bikes in and out of the base stations and vehicles are preserved (constraints (3.2) and (3.4)), the flows of vehicles in and out of the base stations are preserved (constraints (3.5) and (3.6)), and capacities of base stations and vehicles are not vio-

$$\max_{\mathbf{y}^+,\mathbf{y}^-,\mathbf{z}} \sum_{t,k,s,s'} R^{t,k}_{s,s'} \cdot x^{t,k}_{s,s'} - \sum_{t,v,s,s'} P_{s,s'} \cdot z^t_{s,s',v} \tag{3.1}$$

$$\textbf{s.t.} \quad d^{\#,t}_s + \sum_{k,\hat{s}} x^{t-k,k}_{\hat{s},s} - \sum_{k,s'} x^{t,k}_{s,s'} + \sum_v \left(y^{-,t}_{s,v} - y^{+,t}_{s,v}\right) = d^{\#,t+1}_s, \quad \forall t, s \tag{3.2}$$

$$x^{t,k}_{s,s'} \leq d^{\#,t}_s \cdot \frac{F^{t,k}_{s,s'}}{\sum_{k,\hat{s}} F^{t,k}_{s,\hat{s}}}, \quad \forall t, k, s, s' \tag{3.3}$$

$$d^{*,t}_v + \sum_{s \in S} \left(y^{+,t}_{s,v} - y^{-,t}_{s,v}\right) = d^{*,t+1}_v, \quad \forall t, v \tag{3.4}$$

$$\sum_{s' \in \mathcal{S}} z^t_{s,s',v} - \sum_{s' \in \mathcal{S}} z^{t-1}_{s',s,v} = \sigma^t_{v,s}, \quad \forall t, s, v \tag{3.5}$$

$$\sum_{s' \in \mathcal{S}, v \in \mathcal{V}} z^t_{s,s',v} \leq 1, \quad \forall t, s \tag{3.6}$$

$$y^{+,t}_{s,v} + y^{-,t}_{s,v} \leq C^*_v \cdot \sum_{i \in \mathcal{S}} z^t_{s,i,v}, \quad \forall t, s, v \tag{3.7}$$

$$0 \leq x^{t,k}_{s,s'} \leq F^{t,k}_{s,s'}, 0 \leq d^{\#,t}_s \leq C^{\#}_s, 0 \leq y^{+,t}_{s,v}, y^{-,t}_{s,v} \leq C^*_v, 0 \leq d^{*,t}_v \leq C^*_v \tag{3.8}$$

$$z^t_{s,s',v} \in \{0, 1\} \tag{3.9}$$

Table 3.3: SOLVEDRRP()

lated (constraints (3.7) and (3.8)). More importantly, these constraints ensure that the flows of bikes between stations follow the flows of bikes observed in the demand and transition model, **F**. More specific details of the constraints employed in SOLVEDRRP() of Table (3.3) are as follows:

Objective: To represent the trade-off between lost demand (or alternatively the revenue from customer trips) and the cost of using vehicles, we employ the dollar value of both quantities and combine them into the overall profit. This objective is represented in Expression (3.1) of the MILP in SOLVEDRRP().

Flows of bikes in and out of stations are preserved: Constraints (3.2) enforce this flow preservation. Intuitively, in these constraints, we ensure that the number of

bikes at a base station at a time step ($d_s^{\#,t+1}$) is equivalent to the sum of the number of bikes at the same base station in the previous time step ($d_s^{\#,t}$) and the net number of bikes coming into the station during that time step $\Big(\sum_{k,\hat{s}} x_{\hat{s},s}^{t-k,k} - \sum_{k,s'} x_{s,s'}^{t,k} + \sum_v \left(y_{s,v}^{-,t} - y_{s,v}^{+,t}\right)\Big)$.

Flows of bikes between any two stations follow the transition dynamics observed in the data: As a subset of arrival demand can be served if the number of bikes present in a station is less than the arrival demand, constraints (3.3) ensure that the flows of bikes between station s and s' should be less than the product of the number of bikes present in the source station s (i.e., $d_s^{\#,t}$) and the transition probability that a bike will move from s to s' according to expected customer demand (i.e., $F_{s,s'}^{t,k} / \sum_{k,\hat{s}} F_{s,\hat{s}}^{t,k}$).

Flows of bikes in and out of vehicles are preserved: Constraints (3.4) enforce this flow preservation. Intuitively, in these constraints, we ensure that the number of bikes in a vehicle at a time step ($d_v^{*,t+1}$) is equivalent to the sum of the number of bikes in the vehicle at the previous time step ($d_v^{*,t}$) and the net number of bikes coming into the vehicle during that time step $\Big(\sum_{s \in S} \left(y_{s,v}^{+,t} - y_{s,v}^{-,t}\right)\Big)$.

Flows of vehicles in and out of stations are preserved: Constraints (3.5) enforce this flow preservation. Intuitively, in these constraints, we ensure that the number of vehicles going out of a station ($\sum_{s' \in \mathcal{S}} z_{s,s',v}^{t}$) is equivalent to the sum of the number of vehicles coming into the station ($\sum_{s' \in \mathcal{S}} z_{s',s,v}^{t-1}$) and the number of vehicles that were present at that station[3] ($\sigma_{v,s}^{t}$).

A maximum of one vehicle can be present in one station at any time step: Due to limited space availability near base stations and to avoid a synchronisation issue

[3]This second term is greater than zero for the first time step only and for rest of the time steps it is set to zero.

in pickup or drop-off events by multiple vehicles from the same station at the same time step, constraints (3.6) ensure that at any time step, the maximum number of vehicles at a station ($\sum_{s' \in \mathcal{S}} z^t_{s,s',v}$) is 1.

Vehicles can only pick up or drop off bikes at a station if they are present at that station: Constraints (3.7) enforce that the number of bikes picked up or dropped off from a station at each time step by each vehicle is bounded by whether the station is visited by the vehicle at that time step or not.

Station and vehicle capacities are not exceeded when repositioning bikes: Constraints (3.8) ensure that the number of bikes at a station s does not exceed the number of available docks at that station ($C^{\#}_s$). Similarly, these constraints also enforce that the number of bikes picked up or dropped off by a vehicle v in aggregate does not exceed the capacity of the vehicle (C^{*}_v).

The size of the above described model grows exponentially as the number of stations increases. To tackle this problem, we describe two mechanisms, namely, dual decomposition and abstraction to improve the scalability of the optimisation model delineated in Table (3.3).

3.3 Dual Decomposition Approach for Solving the DRRP

We now provide a decomposition approach to exploit the minimal dependency that exists in the MILP of SOLVEDRRP() between the routing problem (how to move vehicles between base stations to pick up or drop off bikes) and the repositioning problem (how many bikes to pick up and drop off from each station). The following observation highlights this minimal dependency:

Observation 1. *In the MILP of Table* (3.3)*:*

- $\boldsymbol{y}^+$ *and* $\boldsymbol{y}^-$ *variables capture the solution for the repositioning problem.*

- $\boldsymbol{z}$ *variables capture the solution for the routing problem.*

These sets of variables only interact due to constraints (3.7)*. In all other constraints of the original model, the routing and repositioning variables are completely independent.*

In order to exploit observation (1), we use the well-known Lagrangian dual decomposition (LDD) (Fisher, 1985; Gordon, Varakantham, Yeoh, Lau, Aravamudhan, & Cheng, 2012) technique. While this is a general purpose approach, its scalability, usability and utility depend significantly on the following characteristics of the model:

1. *Identifying the right constraints to be dualized:* This step is crucial to ensure that the resulting subproblems are easy to solve and the resulting bound derived from the dual solution is tight during the LDD process. If the right constraints are not dualized, then the underlying Lagrangian based optimisation may not be decomposable or it may take significantly more time than the original MILP to find the desired solution.

2. *Extraction of a primal solution from an infeasible dual solution:* The primal extraction process is important to derive a valid bound (heuristic solution) during the LDD process. In many cases, the solution obtained by solving the decomposed dual slaves can be infeasible with respect to the original formulation and hence, the overall approach can potentially lead to slower convergence and poor solutions.

The pseudo code for the LDD is provided in Algorithm (1). We first decompose the original problem into a master problem and two slaves (SOLVEREDEPLOY() and SOLVEROUTING()). As highlighted in observation (1), only constraints (3.7) contain a dependency between routing and repositioning variables. Thus, we dualize

Algorithm 1: : SolveLDD($drrp$)

Initialize: $\alpha^0, it \leftarrow 0$;
repeat
$o_1, \mathbf{x}, \mathbf{y}^-, \mathbf{y}^+ \leftarrow$ SOLVEREDEPLOY$(\alpha^{it}, drrp)$;
$\alpha^{it+1}_{s,t,v} \leftarrow \left[\alpha^{it}_{s,t,v} + \gamma \cdot (y^{+,t}_{s,v} + y^{-,t}_{s,v} - C^*_v \cdot \sum_i z^t_{s,i,v})\right]_+$;
$p, x_p, y^-_p, y^+_p \leftarrow$ EXTRACTPRIMAL $(\mathbf{z}, drrp)$;
$it \leftarrow it + 1$;
until $\left[p - (o_1 + o_2)\right] \leq \delta$;
return p, x_p, y^+_p, y^-_p, z;

constraints (3.7) using the dual variables, $\alpha_{s,t,v}$ and obtain the Lagrangian function (expressed as a minimisation problem as shown by Fisher, 1985) as follows:

$$\mathcal{L}(\boldsymbol{\alpha}) = \min_{\mathbf{z},\mathbf{y}^+,\mathbf{y}^-} \Big[- \sum_{t,k,s,s'} R^{t,k}_{s,s'} \cdot x^{t,k}_{s,s'} + \sum_{t,v,s,s'} P_{s,s'} \cdot z^t_{s,s',v} + \sum_{s,t,v} \alpha_{s,t,v} \cdot (y^{+,t}_{s,v} + y^{-,t}_{s,v} - C^*_v \cdot \sum_i z^t_{s,i,v}) \Big] \tag{3.10}$$

$$= \min_{\mathbf{y}^+,\mathbf{y}^-} \Big[- \sum_{t,k,s,s'} R^{t,k}_{s,s'} \cdot x^{t,k}_{s,s'} + \sum_{s,t,v} \alpha_{s,t,v} \cdot \big(y^{+,t}_{s,v} + y^{-,t}_{s,v}\big) \Big] + \min_{\mathbf{z}} \Big[\sum_{t,v,s,s'} z^t_{s,s',v} \cdot (P_{s,s'} - C^*_v \cdot \alpha_{s,t,v}) \Big] \tag{3.11}$$

In Equation (3.11), the first two terms correspond to the repositioning problem and the last term corresponds to the routing problem. The two subproblems corresponding to the repositioning and routing problems are given in Table (3.4) and Table (3.5), respectively.

$$\min_{\mathbf{y}^+,\mathbf{y}^-} - \sum_{t,k,s,s'} R^{t,k}_{s,s'} \cdot x^{t,k}_{s,s'} + \sum_{s,t,v} \alpha_{s,t,v} \cdot (y^{+,t}_{s,v} + y^{-,t}_{s,v})$$

s.t. *Constraints* (3.2), (3.3), (3.4) & (3.8) *hold*

Table 3.4: SOLVEREDEPLOY()

From Equation (3.11), given an α, the dual value corresponding to the original problem is obtained by adding up the objective function values from the two slaves,

$$\min_{\mathbf{z}} \sum_{t,v,s,s'} z^t_{s,s',v} \cdot (P_{s,s'} - C^*_v \cdot \alpha_{s,t,v})$$

s.t. *Constraints* (3.5), (3.6) & (3.9) *hold*

Table 3.5: SOLVEROUTING()

which yields a valid lower bound with respect to the original problem. It should be noted that the decomposition is only for $\mathcal{L}(\boldsymbol{\alpha})$. Next, we have to solve the following optimisation problem at the **master** in order to reduce violations of the dualized constraints:

$$\max_{\boldsymbol{\alpha} \geq 0} \mathcal{L}(\boldsymbol{\alpha}) \tag{3.12}$$

This **master** optimisation problem is solved iteratively using a sub-gradient descent method applied on the dual variables α:

$$\alpha^{k+1}_{s,t,v} = \left[\alpha^k_{s,t,v} + \gamma \cdot (y^{+,t}_{s,v} + y^{-,t}_{s,v} - C^*_v \cdot \sum_i z^t_{s,i,v})\right]_+ \tag{3.13}$$

where the $[]_+$ notation indicates that the value must be equal or greater than zero. This is because we have dualized a "less than or equal to" constraint and a value of less than zero indicates that there is no violation of the constraint. γ is a step-size parameter that is set using the standard strategy presented by Bertsekas (1999) (refer to section 6.3.1). The value within parenthesis $()$ in Equation (3.13) is the sub-gradient and is computed from the solutions of the two slaves.

The algorithm terminates 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 pre-determined threshold value δ. In order to compute the optimality gap[4], we need the best primal solution in conjunction with the dual solution. Therefore, it is important to obtain a primal solution after each iteration from the

[4]The gap between dual and primal solution which is known as duality gap, is the measurement of solution quality derived from the LDD. We reach an optimal solution if the duality gap becomes zero.

solutions of the slaves. In our case, however, the aggregate solution obtained from slaves may not always be feasible with respect to the original problem in Table (3.3).

Observation 2. *The infeasibility in the dual solution arises because the routes of the vehicles (obtained by solving the routing slave) may not be consistent with the repositioning plan of bikes (obtained by solving the repositioning slave). However, the solution for the routing slave is always feasible and can be fixed to obtain a feasible primal solution with respect to the original problem.*

Let, $Z^t_{s,v} = \sum_{s'} z^t_{s,s',v}$. We extract the primal solution by solving the optimisation formulation provided in Table (3.6). Essentially, we solve the repositioning slave with an additional set of constraints (3.14), which ensure that repositioning in a station is possible if a vehicle is present there. More specifically, constraints (3.14) are equivalent to constraints (3.7) where we use the solution values of the routing slave (**z**) as the input. Thus, ExtractPrimal() satisfies all the constraints of Table (3.3) and produces a feasible solution to the original problem. Finally, we subtract the routing cost from the objective value to get the correct primal value.

$$\max_{\mathbf{y}^+,\mathbf{y}^-} \sum_{t,k,s,s'} R^{t,k}_{s,s'} \cdot x^{t,k}_{s,s'}$$

s.t. *Constraints* (3.2), (3.3), (3.4), (3.8) *hold and*

$$y^{+,t}_{s,v} + y^{-,t}_{s,v} \leq C^*_v \cdot Z^t_{s,v}, \qquad \forall t, s, v \qquad (3.14)$$

Table 3.6: EXTRACTPRIMAL()

Proposition 1. *(Fisher, 1985) : The error in the solution quality obtained by the Lagrangian dual decomposition method in Algorithm* (1) *is bounded by the difference between the primal objective,* p *and the dual objective,* $(o_1 + o_2)$.

3.4 Abstraction Approach for Solving DRRP

Even after applying the LDD, we can only solve problems with at most 60 base stations, 38 time steps and 5 vehicles within a threshold time of 12 hours. However, in most of the cities, the number of base stations is higher. In this section, we provide an abstraction mechanism to further speed up the solution process. We first provide two observations from the real data that assist with deciding on the method to abstract base stations:

- Figure (3.4) provides heat maps of empty stations[5] during various times of the day for *Capital Bikeshare*. Figure (3.4a) and (3.4b) show the heat maps of empty stations in the morning peak hours. Similarly, Figure (3.4c) and (3.4d) show the heat maps of empty stations during the evening peak hours. All the heat maps indicate that the stations near to each other exhibit similar behaviour.

- Base stations are relatively close to each other. For instance, in case of the *Capital Bikeshare*, there are 5-6 base stations within 2 blocks (up to 0.4 miles or 0.64 kilometers) in the center of the city.

From the above observations, since nearby stations exhibit similar behaviour and are also close enough to be covered by a carrier vehicle with minimal travel, we exploit the geographical proximity based clustering method to obtain abstract stations. Specifically, we employ relative distances between stations while clustering stations into abstract stations. We follow the following three steps typically employed in abstraction and introduced by Knoblock (1991): (1) create an instance of the DRRP with abstract stations, each of which is a group (obtained from clustering) of the original base stations, (2) solve the abstract DRRP using the LDD and

[5]A station is considered empty if there are no bikes in the station for more than 2 minutes. For a specific time of a specific day (Monday, Tuesday, etc.), we use the trip history data and count the number of times each station became empty over a reasonably long duration (a year). Red corresponds to stations that became empty frequently, green corresponds to stations that became empty moderately and purple is for stations that rarely became empty.

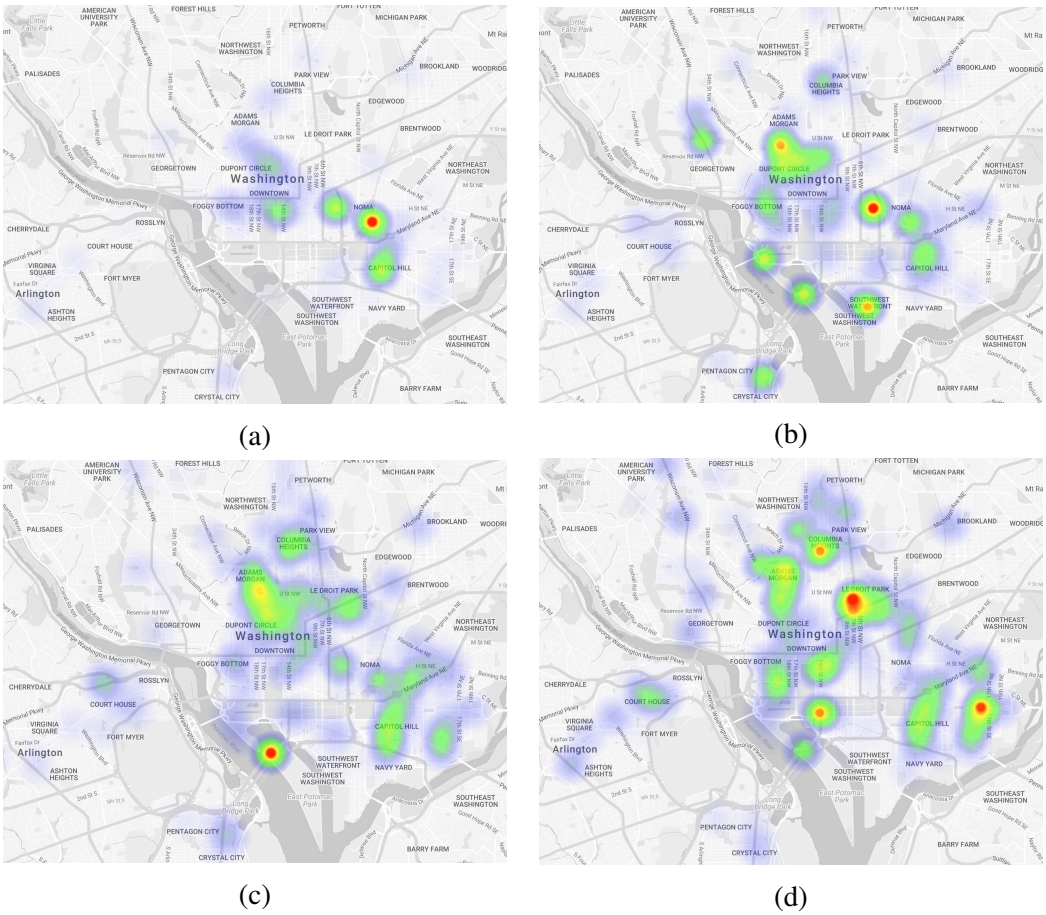


Figure 3.4: Heat maps for empty stations (data set: *Capital Bikeshare*): (a) morning peak (7AM - 9AM), (b) morning peak (9AM - 11AM), (c) evening peak (4PM - 6PM), (d) evening peak (6PM - 8PM).

obtain the routing and repositioning solution over abstract stations, and (3) derive the routing and repositioning solution for the original DRRP from the routing and repositioning solution of the abstract DRRP.

3.4.1 Create Abstract DRRP

The first step in this approach is to generate the abstract DRRP, $\langle \tilde{\mathcal{S}}, \mathcal{V}, \tilde{\mathbf{C}}^{\#}, \mathbf{C}^{*}, \tilde{\mathbf{d}}^{\#,0}, \mathbf{d}^{*,0}, \{\tilde{\sigma}_v^0\}, \tilde{\mathbf{F}}, \tilde{\mathbf{R}}, \tilde{\mathbf{P}} \rangle$ from the original DRRP. Everything related to vehicles in the abstract DRRP remains the same as in the original DRRP. In practice, revenue, $R_{s,s'}^{t,k}$ depends on the time step, t and the number of time steps, k for which the bike is hired and does not rely on the source or destination station. Hence, we can assume that the revenue model remains the same for the original and abstract DRRPs. We outline

below how the other elements of the abstract DRRP tuple are computed from the original DRRP:

- Stations in the abstract DRRP, $\tilde{\mathcal{S}}$: Grouping of stations $\mathcal{S}$ into abstract stations can either be done by an expert or computed by a clustering approach[6] (e.g., *k*-means clustering). Thus, each abstract station $\tilde{s} \in \tilde{\mathcal{S}}$ 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 an abstract station $\tilde{s}$ is the sum of capacities of all the stations $s \in \tilde{s}$.

- Initial distribution of bikes at the abstract station: $d^{\#,0}_{\tilde{s}} = \sum_{s \in \tilde{s}} d^{\#,0}_{s}$. The initial distribution of bikes of an abstract station $\tilde{s}$ is the sum of the initial distributions of all the stations $s \in \tilde{s}$.

- Initial distribution of vehicle: $\sigma^{0}_{v,\tilde{s}} = 1$, if $\exists s \in \tilde{s}$, $\sigma^{0}_{v,s} = 1$. The vehicle v is initially located in abstract station $\tilde{s}$ if its original location (i.e., station s) belongs to the abstract station $\tilde{s}$.

- Flows of bikes in the abstract DRRP: $F^{t,k}_{\tilde{s},\tilde{s}'} = \sum_{\{s \in \tilde{s}, s' \in \tilde{s}'\}} F^{t,k}_{s,s'}$. The flows of bikes from an abstract station $\tilde{s}$ to $\tilde{s}'$ are calculated as the sum of the flows of all the bikes taken by the customers from any station $s \in \tilde{s}$ to a station $s' \in \tilde{s}'$ in the original DRRP.

- Routing cost for the vehicles in the abstract DRRP: $P_{\tilde{s},\tilde{s}'} = \max_{\{s \in \tilde{s}, s' \in \tilde{s}'\}} P_{s,s'}$. We consider a conservative option of taking the worst case penalty. Specifically, we take the maximum routing cost for traveling between any pair of stations $s \in \tilde{s}$ and $s' \in \tilde{s}'$.

[6]The grouping of base stations can be done in various ways and the results may vary for different problem instances. Clustering base stations according to geographical proximity is one option and the experimental results show that it provides a reasonable improvement over the two benchmark approaches.

3.4.2 Solve the Abstract DRRP

In the second step, we use the LDD approach from Section 3.3 to solve the abstract DRRP. There are two possible assumptions we can make about the movements of vehicles in an abstract station: (1) a vehicle can visit all stations of an abstract station in a single time step, and (2) a vehicle can visit one station within an abstract station in a single time step.

3.4.2.1 Vehicle Can Visit All Stations of an Abstract Station in a Single Time Step

As a vehicle can visit all the stations of an abstract station within one time step, we need to ensure that at most one vehicle is present in an abstract station in each time step to avoid the inconsistency in pickup or drop-off events by different vehicles. Therefore, the optimisation model for solving the abstract DRRP is equivalent to the one shown in Table (3.3) and we directly use the LDD approach from Section 3.3 to efficiently solve the abstract DRRP.

$$\min_{\tilde{\mathbf{z}}} \sum_{t,v,\tilde{s},\tilde{s}'} P_{\tilde{s},\tilde{s}'} \cdot \tilde{z}^t_{\tilde{s},\tilde{s}',v} - \sum_{\tilde{s},t,v} \alpha_{\tilde{s},t,v} \cdot C^*_v \cdot \sum_i \tilde{z}^t_{\tilde{s},i,v}$$

s.t. *Constraints* (3.5) *&* (3.9) *hold*

$$\sum_{j \in \tilde{S}, v \in \mathcal{V}} \tilde{z}^t_{\tilde{s},j,v} \leq |\tilde{s}|, \qquad \forall t, \tilde{s} \quad (3.15)$$

Table 3.7: SOLVEABSTRACTROUTING()

3.4.2.2 Vehicle Can Visit One Station within an Abstract Station in a Single Time Step

As the abstract stations contain multiple base stations, we need to modify constraints (3.6) to allow multiple vehicles in an abstract station. Table (3.7) provides the modified version of the routing slave to solve the abstract DRRP, where constraints (3.5) & (3.9) are defined over $\tilde{z}$. The modified set of constraints (3.15)

ensure that at any time step maximum $|\tilde{s}|$ vehicles can visit an abstract station $\tilde{s}$. However, the repositioning slave and master function remain unchanged. There are two key outputs from the LDD algorithm: (1) repositioning solution, $\tilde{y}$ for moving bikes between abstract stations, and (2) routing solution, $\tilde{z}$ for moving vehicles between abstract stations at different time steps.

3.4.3 Deriving Solutions for the Original DRRP

In the third step, we retrieve the solution for the original DRRP from the abstract DRRP solution.

3.4.3.1 Vehicle Can Visit All Stations of an Abstract Station in a Single Time Step

As we are abstracting the base stations based on their relative distance, all the base stations within an abstract station are located nearby (less than a few kilometers in our data sets). So, in reality it is possible for a vehicle to visit all the base stations of an abstract station within one time step. The mechanisms to retrieve the repositioning and routing solutions for the original DRRP from the abstract DRRP solution are outlined below.

Repositioning solution for the original DRRP: Based on a fixed routing solution, $\tilde{\mathbf{z}}$ for the abstract DRRP, we retrieve the repositioning solution for the original DRRP. Specifically, we fix the locations where vehicles will be present at different time steps and remove all constraints which are only related to vehicle routing (since the routing solution is fixed) in the optimisation model of Table (3.3). Solving this optimisation model yields a repositioning solution for the original DRRP. Formally, from the abstract DRRP solution $\tilde{\mathbf{z}}$, we obtain constants $\mathbf{Z}$ as follows:

$$Z_s^t = \begin{cases} 1, \textbf{if } s \in \tilde{s} \ \wedge \ \sum_{v,\tilde{s}'} \tilde{z}^t_{\tilde{s},\tilde{s}',v} = 1 \\ 0, \textbf{otherwise} \end{cases}$$

The final optimisation model to obtain the repositioning solution for the original DRRP is shown in Table (3.8). The key differentiating constraints that have not been used earlier are constraints (3.20). These constraints ensure that the 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 from the abstract station according to the repositioning solution of the abstract DRRP.

$$\max_{\mathbf{y}^+,\mathbf{y}^-} \sum_{t,s,s,s'} R^{t,k}_{s,s'} \cdot x^{t,k}_{s,s'} \tag{3.16}$$

$$\textbf{s.t.} \quad d^{\#,t}_s + \sum_{k,\hat{s}} x^{t-k,k}_{\hat{s},s} - \sum_{k,s'} x^{t,k}_{s,s'} + y^{-,t}_s - y^{+,t}_s = d^{\#,t+1}_s, \qquad \forall t, s \tag{3.17}$$

$$x^{t,k}_{s,s'} \leq d^{\#,t}_s \cdot \frac{F^{t,k}_{s,s'}}{\sum_{k,\hat{s}} F^{t,k}_{s,\hat{s}}}, \qquad \forall t, k, s, s' \tag{3.18}$$

$$y^{+,t}_s + y^{-,t}_s \leq C^*_v \cdot Z^t_s, \qquad \forall t, s \tag{3.19}$$

$$\sum_{s \in \tilde{s} | \sum_{\tilde{s}'} \tilde{z}^t_{\tilde{s},\tilde{s}',v}=1} [y^{+,t}_s - y^{-,t}_s] = d^{*,t+1}_v - d^{*,t}_v, \qquad \forall t, \tilde{s} \tag{3.20}$$

$$0 \leq x^{t,k}_{s,s'} \leq F^{t,k}_{s,s'}, y^{+,t}_s, y^{-,t}_s \leq C^*_v, d^{\#,t}_s \leq C^{\#}_s \tag{3.21}$$

Table 3.8: GETSTATIONREDEPLOY($\mathbf{Z}, \mathbf{d}^*$)

Routing solution for the original DRRP: Given the routing solution for the abstract DRRP (also referred to as the abstract routing solution), the vehicle assigned to each abstract station at a time step is fixed. From this abstract routing solution, our goal is to find the routing solution for all the stations within each abstract station at each time step. This routing solution must be consistent with the repositioning solution computed for the original DRRP. We use $\mathbf{Y}$ (instead of $\mathbf{y}$) to represent the final repositioning solution.

For each vehicle, we compute the routing solution for the original DRRP incrementally by starting at the first time step and from the starting abstract station. We identify the route to be taken between all the base stations within this starting abstract station. Then, we move to the abstract station for the next time step

recommended by the abstract routing solution and so on.

For each vehicle, the first step in computing the routing solution for stations within an abstract station is to identify the starting station[7]. We consider the starting station for non-starting abstract stations as the one that is nearest to the station from where the vehicle has exited in the previous time step. An advantage of this incremental method is that it minimises the routing cost for transition between abstract stations.

Once the starting station is obtained and the repositioning solution $\mathbf{Y}$ is known, we employ the optimisation model in Table (3.9) to find an intra-abstract station routing solution. We compute the best route within the stations of an abstract station $\tilde{s}$, while visiting each base station once and satisfying the repositioning numbers from each station, $\mathbf{Y}$.

$$\min_{\mathbf{z}} \sum_{t,s,s'} P_{s,s'} \cdot z^t_{s,s',v} \tag{3.22}$$

$$\textbf{s.t.} \quad \hat{d}^{*,t} + \sum_{s} (Y^+_s - Y^-_s) \cdot \sum_{s'} z^t_{s,s',v} = \hat{d}^{*,t+1}, \qquad \forall t \in \hat{T} \tag{3.23}$$

$$\sum_{t,s'} z^t_{s,s',v} = 1, \qquad \forall s \in \tilde{s} | (Y^+_s + Y^-_s) > 0 \tag{3.24}$$

$$\sum_{s'} z^t_{s,s',v} - \sum_{\hat{s}} z^{t-1}_{\hat{s},s,v} = \sigma^t_{v,s}, \qquad \forall t \in \hat{T}, s \in \tilde{s} \tag{3.25}$$

$$0 \leq \hat{d}^{*,t} \leq C^*_v, z^t_{s,s',v} \in \{0,1\} \tag{3.26}$$

Table 3.9: GETINTRAROUTING($\tilde{s}, v, \mathbf{Y}$)

The objective delineated in Expression (3.22) is to minimise the routing cost of the vehicle and the constraints are defined as follows:

- *Flows of bikes in and out of a vehicle are preserved:* Constraints (3.23) enforce this by ensuring that the number of bikes in the vehicle at time step $t+1$ is equal to the sum of the number of bikes present in the vehicle at time step t plus the net number of bikes picked up from a station s at that time step. Note

[7]Since the position of every vehicle is known at first time step in the original DRRP tuple, we have the starting station for the starting abstract station.

that, the time index t here is used to represent the sequence of moves for the vehicle between the base stations within an abstract station.

- *Each station is visited only once:* Constraints (3.24) restrict that each base station where a repositioning is required (i.e., $Y_s^+/Y_s^- > 0$) is visited only once.

- *Flow conservation of each vehicle at a station:* Constraints (3.25) ensure that the flow in to a station s (i.e., $\sum_{s'} z^{t-1}_{s',s,v}$) is equal to the flow out from that station at time step t (i.e., $\sum_{s'} z^{t}_{s,s',v}$). σ^0 represents the initial location of the vehicle and it is used to ensure that the vehicle moves appropriately out of the initial location.

- *Capacity of the vehicle is not exceeded during repositioning:* Constraints (3.26) ensure that the number of bikes picked up or dropped off by a vehicle in aggregate does not exceed the capacity of the vehicle (C_v^*).

Example 3.4.1. *Figure* (3.5) *provides a handcrafted toy example to illustrate the abstraction method where a vehicle can cover all stations within an abstract station in one time step. Solid dots represent the base stations and big circles represent the abstract stations. We considered a problem with 13 base stations and grouped them into three abstract stations (with 5 stations in abstract station 1 and 4 stations each in abstract stations 2 and 3). Initial location of a vehicle is indicated with a circle over the solid dot. Figure* (3.5a) *depicts the optimal abstract station level routing solution (by solving the LDD based global MILP on the abstract DRRP) for the vehicle. Figure* (3.5b) *depicts the base station level routing solution within the abstract station 1 at the initial time step. It also shows the route from the exit station of abstract station 1 to its nearest station in abstract station 2. By this incremental process, we find the base station level routing solution for the vehicle. Figure* (3.5c), (3.5d) *depict the base station level routing solution within abstract station 2 and 3 respectively.*

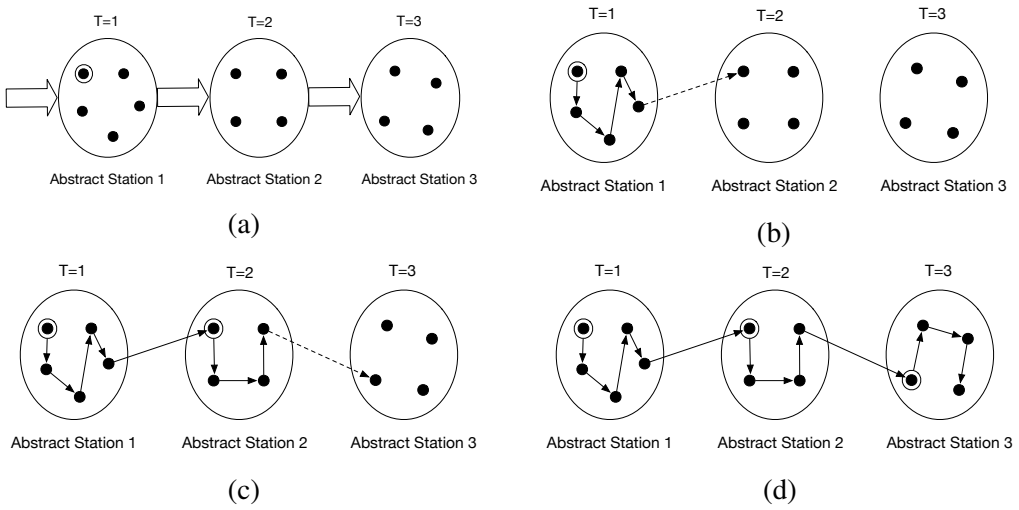


Figure 3.5: Routing solution in the abstract DRRP: (a) abstract station level routing solution, (b) routing solution within abstract station 1, (c) routing solution within abstract station 2, and (d) routing solution within abstract station 3.

3.4.3.2 Vehicle Can Visit One Station within an Abstract Station in a Single Time Step

Table (3.10) provides the optimisation model to compute a feasible solution for the original DRRP with the assumption that a vehicle can only travel to one station in each time step. We solve the global MILP SolveDRRP() for the original DRRP provided in Table (3.3) with an additional set of constraints (3.27) to ensure that a vehicle can only be present in a base station at any time step if the station belongs to the abstract station where the vehicle is located in the abstract DRRP solution. Specifically, the decision variable $z^t_{s,s',v}$ can only be 1 if $s \in \tilde{s}$, $s' \in \tilde{s}'$ and $\tilde{z}^t_{\tilde{s},\tilde{s}',v} = 1$. In the MILP of RetrieveDRRP(), we set the decision variables $z^t_{s,s',v}$ to 0 if $s \in \tilde{s}$, $s' \in \tilde{s}'$ and $\tilde{z}^t_{\tilde{s},\tilde{s}',v} = 0$. Thus, RetrieveDRRP() becomes easier to solve than SolveDRRP().

3.4.4 Reasons for Improvement in Scalability

The scale of the optimisation models used for solving the abstract DRRP and deriving an original DRRP solution from the abstract DRRP solution are reduced in comparison to the original optimisation model. Hence, this abstraction method is

$$\max_{\mathbf{y}^+,\mathbf{y}^-,\mathbf{z}} \sum_{t,k,s,s'} R^{t,k}_{s,s'} \cdot x^{t,k}_{s,s'} - \sum_{t,v,s,s'} P_{s,s'} \cdot z^t_{s,s',v}$$

s.t. *Constraints* (3.2)- (3.9) *hold and*

$$\sum_{s\in\tilde{s},s'\in\tilde{s}'} z^t_{s,s',v} = \tilde{z}^t_{\tilde{s},\tilde{s}',v}, \qquad \forall \tilde{s},\tilde{s}',t,v \qquad (3.27)$$

Table 3.10: RETRIEVEDRRP()

able to substantially speed up the solution process. Specifically, here are the reasons for reduction in runtime of the optimisation models:

- **Reduction in the number of variables and constraints**: The number of variables and constraints in the optimisation model are significantly reduced. For instance, for a 300 station problem in the original optimisation model of Table (3.3), there would be 90000 binary decision variables, **z** for each time step. On the other hand, for an abstract DRRP with 50 abstract stations, there would only be 2500 **z** variables for each time step.

- **Relaxation**: Another important reason for significant improvement in scalability is that the optimisation models for computing routing and repositioning solution for the abstract DRRP are the relaxations of the optimisation model for the original DRRP. This is because constraints in the optimisation models for the abstract DRRP are obtained by aggregating the constraints that are present in the optimisation model for the original DRRP.

3.5 Experimental Setup

In this section, we describe the real and synthetic data sets that are used in the computational experiments, the benchmark approaches that are implemented for the computational comparisons, and the simulation model used to compute the comparison metrics.

Since our goal is to avoid people from going back to using private vehicles

due to unavailability of bikes, the key comparison metric is the total amount of lost demand. To ensure that the amount of lost demand is reduced at no extra fuel cost to the operators, we also consider the total profit as a metric. The runtime is primarily employed to measure scalability and whether we are able to get a high quality solution within a reasonable amount of time.

3.5.1 Real and Synthetic Data Sets

We employ data sets of two leading bike sharing systems in US[8], namely, *Capital Bikeshare* (Washington, DC) and *Hubway* (Boston, MA), and the synthetic data sets are derived from these real-world data sets. The data items contained in the bike sharing data sets are previously mentioned in Table (3.1). In addition to the data items provided in the data sets, we collect data about the cost of fuel for vehicles[9] from authentic sources. It should be noted that we consider a significant overestimation of the costs to ensure our results are not too sensitive to these values. These elements of real-world data sets and the data collected from the authentic sources are used to populate the DRRP model.

As for the synthetic data sets, they are generated from the two real-world data sets as follows: (1) we take a subset of the stations from the real-world data sets, (2) customer demand, station capacity, geographical location of stations and initial distribution are drawn from the real-world data for the selected stations, and (3) we take the same revenue and cost model discussed earlier from the real-world data sets.

[8]The data is taken from *Capital Bikeshare* [http://capitalbikeshare.com/system-data] and *Hubway* BSS [http://hubwaydatachallenge.org/trip-history-data].

[9]The mileage results in Table 2 of Fishman, Washington, and Haworth (2014) show that carrier vehicles in BSS consume 1 litre of diesel for traveling approximately 12 kilometers. http://www.globalpetrolprices.com/diesel_prices/#USA shows that the price of diesel in January, 2017 is 0.67 USD per litre, but we overestimate it as 1.5 USD to include other operational costs.

3.5.2 Approaches

We employ the commercial linear optimisation solver CPLEX to solve linear programs and mixed integer linear programs. We refer to the optimisation model of Table (3.3) as **MILP**. The dual decomposition method for solving the MILP that is described in Section 3.3 is referred as **LDD** (Lagrangian dual decomposition). Finally, we refer to the abstraction approach described in Section 3.4 as **Abstraction**. The overall approach (LDD+Abstraction) is referred to as **dynamic**.

The first approach that is used as a benchmark for performance comparison is the static repositioning approach (referred to as **static** in the graphs) from Section 2.3.1. We also compared our approach with the online heuristic approach which is adapted from Schuijbroek et al. (2017) as mentioned in Section 2.3.2. This approach is referred to as **online**. Finally, we evaluate the performance of all the above mentioned approaches by using a simulation model from Section 2.3.1.1 that is based on the past data.

3.5.3 Estimating Actual Demand

We only know the satisfied demand from existing data sets. Specifically, when a base station becomes empty, the unobserved lost demand is not captured in the data sets.

Previous works in inventory management have represented and verified the random arrival of customer demand following a Poisson process. More importantly, earlier works in bike sharing (Kabra et al., 2015; Shu et al., 2013; George & Xia, 2011) have also represented the random arrival of customers at each station and at each time step using a Poisson distribution and assumed that customers choose their destination station with a certain probability. In a similar vein, we also represent the arrival of customers at a base station in a time step using a Poisson distribution. Since we can only know about the satisfied demand from the data sets, the mean of the Poisson distribution is the average served demand (outgoing flow) in that time

step. If $f_{s,s'}^{t,k}$ denotes the average number of bikes booked from station s at time step t and reached station s' at time step $t+k$, then the total outgoing flow from station s at time step t is given by $O_s^t = \sum_{s',k} f_{s,s'}^{t,k}$.

Formally, a demand scenario at station s at time step t (denoted by D_s^t) is generated from a Poisson distribution with a mean of O_s^t (i.e., $D_s^t = \text{Poisson}(O_s^t)$). Finally, the flow from station s to s' at time step t is calculated as the product between outgoing flow from station s and the probability of moving from s to s' at time step t (i.e., $F_{s,s}^{t,k} = D_s^t \cdot \frac{f_{s,s'}^{t,k}}{\sum_{k,s'} f_{s,s'}^{t,k}}$).

3.5.4 Evaluation Methodology

We employ the following two general steps to evaluate our approach:

1. We compute repositioning and routing solution based on the DRRP tuple that is populated from the training data set.

2. The computed solutions are then evaluated on a simulation using the test data set. That is to say, transitions in the simulation follow the aggregate transition dynamics observed in testing data set where the demand scenarios are generated from Poisson distribution. The evaluation is then aggregated over these generated demand scenarios.

In cases where we do not have sufficient data, we calculate a solution based on the entire data set and we evaluate our solution on various samples from the Poisson distribution with the mean computed from that data set.

For the online benchmark approach of Schuijbroek et al. (2017), the next time step solution for moving vehicles and bikes (recommended by the repositioning strategy) is executed using the current positions of bikes and vehicles in the simulation based on the test data set.

For the static method, we employ the simulation to compute the flows of bikes in each time step when no repositioning is done and use that flow information to

calculate the expected profit and lost demand. Given the aggregated flow **F** and the actual flow **x**, the revenue is computed as $\sum_{t,k,s,s'} \left[R^{t,k}_{s,s'} \cdot x^{t,k}_{s,s'}\right]$, while the lost demand is computed as $\sum_{t,k,s,s'} \left[F^{t,k}_{s,s'} - x^{t,k}_{s,s'}\right]$.

3.6 Experimental Results

In this section, we verify the following claims[10]:

1. In terms of scalability, the LDD improves over the MILP and the use of Abstraction on top of the LDD further improves the performance. In terms of solution quality, both the LDD and Abstraction obtain near optimal solutions.

2. Our dynamic approach (LDD + Abstraction of MILP) improves upon the two benchmark approaches (static and online) in terms of lost demand and profit.

3. Our approach remains robust with respect to changes in other input parameters such as the number of vehicles and the unit cost for routing.

3.6.1 Utility of LDD and Abstraction

To validate the claim that the LDD and Abstraction both improve the original MILP, we provide three sets of results. As the MILP with and without the LDD can only solve small problem instances, we provide these results on the synthetic data sets generated from the *Capital Bikeshare* data set.

Runtime performance: First, we compare the runtime performance of the LDD with the global MILP (SOLVEDRRP()) in Figure (3.6a). The X-axis denotes the scale of the problem where we varied the number of stations from 5 to 50. The Y-axis denotes the total time taken to solve the problem in seconds on a logarithmic scale. Except on small instances (e.g., 5-10 stations), the LDD outperforms

[10]All the linear optimisation models were solved using IBM ILOG CPLEX Optimisation Studio V12.5 incorporated within python code on a 3.2 GHz Intel Core i5 machine.

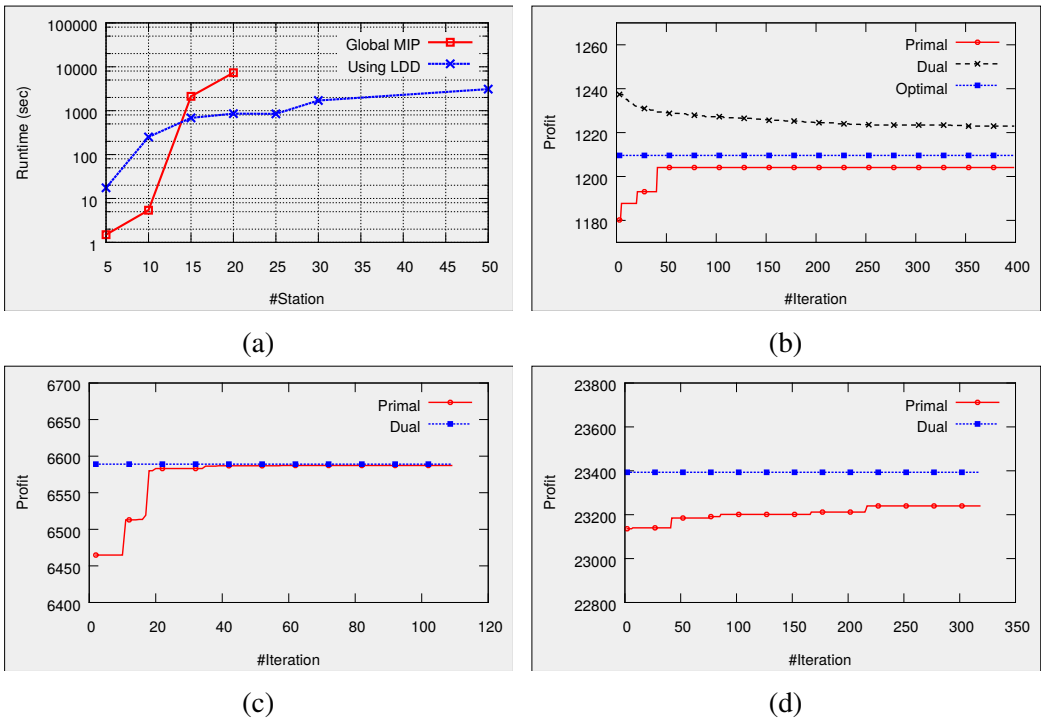


Figure 3.6: (a) Runtime comparison between the global MILP and LDD, (b) duality gap in the synthetic data set with 20 stations, (c) duality gap in the *Hubway* data set, (d) duality gap in the *Capital Bikeshare* data set.

the global MILP with respect to runtime. More specifically, the global MILP was unable to finish within a cut-off time of 6 hours for any problem with more than 20 stations, while the LDD was able to obtain near optimal solutions on problems with 50 stations in less than an hour.

Duality gap: In the second set of results, we demonstrate the convergence of the LDD to near optimal solutions. The LDD achieves an optimal solution if the duality gap, i.e., the gap between primal and dual solutions, becomes zero. Figure (3.6b) shows that the duality gap for the instance with 20 stations is only 1%. Figure (3.6c) and (3.6d) depict the duality gap for the real-world data set of *Hubway* (with 95 base stations and grouped into 25 abstract stations) and of *Capital Bikeshare* (with 305 base stations and grouped into 50 abstract stations), respectively. For these larger problems we are able to obtain a solution with the duality gap of less than 0.5%.

Effect of abstraction: Finally, we demonstrate the performance of the abstraction method in comparison with the optimal solution of an instance with 30 base stations. We grouped these 30 base stations into 8 abstract stations. Then we run the LDD based optimisation on both the base station and abstraction station problems. Table (3.11) shows the effect of the abstraction approach on the generated profit and runtime based on five random scenarios of customer demand. With abstraction, while there is only a reduction of less than 0.3% in profit on average from the optimal solution, it gives a significant computational gain.

	With abstraction		Without abstraction		
Instance	**Profit**	**Runtime (sec)**	**Profit**	**Runtime (sec)**	**Profit loss for abstraction**
1	23580	51	23640	3840	0.25%
2	23627	106	23678	3540	0.21%
3	23610	57	23727	3120	0.5%
4	23613	49	23645	3150	0.13%
5	23519	45	23590	3119	0.30%
Average	23590	62	23656	3354	0.27%

Table 3.11: Effect of abstraction.

The key reason behind this negligible loss of demand when using the abstraction technique is the specific demand patterns observed in the real-world data sets. As shown in the heat maps of Figure (3.4), the stations that become empty in a particular time period are typically close to each other and hence can be rebalanced within a time step. Since our abstraction is based on geographical proximity, it is ideally suited to handle such situations.

However, the solution quality of our geographical proximity based abstraction mechanism deteriorates if most of the abstract stations become empty at the same time. To demonstrate this situation, we generate artificial demand scenarios where we readjust demand and have high demand for one random station in each abstract station. Table (3.12) demonstrates the performance of our abstraction approach in comparison with the optimal solution. Even in this small example scenario (with 30 stations) for these artificially crafted demand instances, we observe a higher reduc-

tion (more than 3.2% on average) in the profit due to the abstraction in comparison with the case where we consider the real-world demand (Table 3.11).

	With abstraction		Without abstraction		
Instance	**Profit**	**Runtime (sec)**	**Profit**	**Runtime (sec)**	**Profit loss for abstraction**
1	11730	21	12092	2918	2.99%
2	11958	46	12314	3201	2.89%
3	11759	24	12114	3050	2.93%
4	11530	36	12060	1641	4.39%
5	11658	31	11997	1467	2.83%
Average	11727	32	12115	2455	3.21%

Table 3.12: Effect of abstraction on artificially crafted demand.

In this work, we show that our geographical proximity based abstraction mechanism significantly outperforms the existing benchmark approaches due to the specific demand patterns observed in both the real-world data sets of our study. However, our solution approaches are complementary to any abstraction mechanism that can be used to group base stations to reduce the size of the DRRP.

3.6.2 Comparison against Benchmarks

In this section, we provide the following key comparison results of our approach (dynamic) with the two benchmarks (static and online):

1. Results with respect to profit and lost demand
2. Sensitivity results over different demand scenarios generated from a Poisson distribution
3. Sensitivity results with respect to additional unknown demand
4. Sensitivity results with respect to additional known demand

3.6.2.1 Results with respect to Profit and Lost Demand

We first provide the average results on the *Hubway* and *Capital Bikeshare* data sets for the static, online and our dynamic approach. As indicated earlier, our key performance evaluation metric is the lost demand. However, we also provide the performance comparison with respect to the overall profit to show that we can reduce the lost demand without incurring extra cost to the operators. *Hubway* system consists of 95 active stations and *Capital Bikeshare* system consists of 305 active stations. In our approach, we employ k-means clustering to generate 25 and 50 abstract stations, respectively. Stations within an abstract station are typically within a kilometer of each other for both data sets. For fairness in comparison, we allow a vehicle to visit multiple stations in one time step for the online approach. This is because vehicles are allowed to visit all the stations within an abstract station in one time step in our approach. In fact, we provide a reasonable advantage for the online approach by allowing it to visit 5 stations (anywhere in the city) within one time step[11]. Based on the information obtained from Schuijbroek et al. (2017), we employ 5 vehicles for the experiments on *Capital Bikeshare* data set and 3 vehicles for the experiments on *Hubway* data set. We show the results during the peak period and also for the entire day[12].

Table (3.13) shows the average percentage gain in profit and reduction in lost demand with our approach in comparison to the benchmark approaches on the two real-world data sets. The performance gain of our dynamic approach in comparison with static repositioning is computed as follows:

$$\text{Profit gain} = \frac{\text{Profit with dynamic repositioning} - \text{Profit with static repositioning}}{\text{Profit with static repositioning}}$$

[11]This allows for an average distance travelled in one time step with the online heuristic as 17.8 kilometers as opposed to 13.8 kilometers with our approach. Even with this advantage, we demonstrate that our approach performs better.

[12]The planning horizon for our approach is 38 time steps (30 minute intervals during the working hours from 5AM-12AM) for the entire day and 14 time steps for the peak period (30 minute intervals during the morning working hours from 5AM-12PM).

	Whole day (5AM-12AM)				Peak period (5AM-12PM)			
	Gain over static repositioning		Gain over online heuristic		Gain over static repositioning		Gain over online heuristic	
	Profit gain	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**
Hubway	3.47%	45.80%	3.02%	41.17%	9.16%	46.21%	7.15%	44.75%
Capital Bikeshare	2.14%	22.33%	1.4%	9.9%	4.52%	26.38%	0.96%	5.11%

Table 3.13: Profit and lost demand comparison (*Hubway* and *Capital Bikeshare* data sets).

$$\text{LD gain} = \frac{\text{LD with static repositioning} - \text{LD with dynamic repositioning}}{\text{LD with static repositioning}}$$

Based on the aggregate results, our approach (LDD + Abstraction) is always able to outperform both the static and online repositioning solutions with respect to both the profit gain and lost demand. Over the entire day, our approach reduces the lost demand in the *Hubway* data set by at least 45.80% and 41.17% in comparison to the static and online approaches, respectively. For the *Capital Bikeshare* data set, we improve by 22.33% and 9.9% in comparison to the static and online approaches, respectively. Similar results (slightly inferior) were obtained when we considered only the peak hour as the planning period.

Comparison results on *Hubway* Data set: We begin with the results on the real-world data set of *Hubway*. *Hubway* BSS comprises with 95 base stations and we group them into 25 abstract stations. We employ 3 vehicles (courtesy: Schuijbroek et al., 2017) for this experiment. We only have the proper trip history data for third quarter of 2012, from which we compute the average demand for individual weekdays.

Table (3.14) provides the comparison results (on profit and lost demand) between our approach and the two benchmark approaches. Our approach is able to gain 3.5% in profit on average while the lost demand is reduced by an average of 45% over the practice of no repositioning during the day. In comparison with on-

	Whole day (5AM-12AM)				Peak period (5AM-12PM)			
	Gain over static repositioning		Gain over online heuristic		Gain over static repositioning		Gain over online heuristic	
	Profit gain	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**
Mon	2.47%	24.53%	2.41%	22.86%	8.08%	43.56%	7.15%	37.41%
Tue	3.62%	35.15%	4.32%	36.87%	13.01%	55.79%	11.17%	52.27%
Wed	3.17%	30.13%	3.20%	29.22%	12.30%	53.76%	9.43%	48.05%
Thu	4.03%	36.93%	3.92%	35.89%	13.32%	52.56%	9.26%	45.16%
Fri	5.63%	50.00%	5.08%	47.06%	16.15%	67.78%	12.22%	61.33%
Sat	2.20%	69.89%	1.18%	58.21%	0.70%	25.00%	0.63%	35.71%
Sun	3.15%	74.00%	1.00%	58.06%	0.53%	25.00%	0.21%	33.33%
Mean	3.47%	45.80%	3.02%	41.17%	9.16%	46.21%	7.15%	44.75%

Table 3.14: Profit and lost demand comparison (data set: *Hubway*, 3rd quarter of 2012).

line heuristic, our approach is able to reduce the lost demand by an average of 41%, while the profit is increased by 3% on average. In the peak hours, our approach reduces the lost demand by an average of 46% and 44% over the static repositioning and online heuristic approach respectively.

Comparison results on *Capital Bikeshare* Data set: We consider the trip history data of four quarters of 2013 for *Capital Bikeshare* and for each quarter we have done the same set of experiments. Table (3.15) shows that for the first quarter of data, our dynamic approach is able to outperform static repositioning during the peak time as well as during the day, with respect to both the profit gain and the reduction in lost demand. We reduce the lost demand by an average of 20%, a significant improvement over the static repositioning. As expected, for all of these instances, the percentage gain in profit in the peak hours is much higher because most of the lost demand occur in the peak hours. Although the online heuristic performs well in the peak hours, it fails to provide a good quality solution when we consider a long planning horizon (38 time step). In case of long planning horizon, our approach outperforms the online heuristic for all the weekdays both in terms of profit gain and lost demand reduction.

	Whole day (5AM-12AM)				Peak period (5AM-12PM)			
	Gain over static repositioning		Gain over online heuristic		Gain over static repositioning		Gain over online heuristic	
	Profit gain	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**
Mon	1.92%	20.61%	0.51%	5.43%	5.01%	29.07%	-0.74%	-2.81%
Tue	2.14%	19.82%	-0.18%	-0.46%	4.58%	23.08%	-4.31%	-20.19%
Wed	3.11%	24.59%	-0.11%	2.97%	8.89%	34.75%	0.39%	5.38%
Thu	3.55%	28.62%	5.28%	22.92%	7.17%	31.03%	2.18%	12.16%
Fri	3.34%	28.31%	3.39%	15.26%	7.56%	31.69%	2.51%	9.42%
Sat	0.04%	12.06%	-1.74%	-19.86%	-0.34%	12.15%	-2.5%	-10.59%
Sun	0.18%	9.44%	0.07%	-0.39%	-1.46%	3.33%	0.61%	13.86%
Mean	2.04%	20.49%	1.03%	3.70%	4.49%	23.59%	-0.27%	1.03%

Table 3.15: Profit and lost demand comparison (data set: *Capital Bikeshare*, 1st quarter of 2013).

	Whole day (5AM-12AM)				Peak period (5AM-12PM)			
	Gain over static repositioning		Gain over online heuristic		Gain over static repositioning		Gain over online heuristic	
	Profit gain	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**
Mon	2.51%	27.1%	1.83%	18.76%	5.13%	32.84%	1.3%	9.57%
Tue	3.02%	26.44%	1.48%	15.33%	6.79%	28.72%	1.89%	8.84%
Wed	2.67%	22.09%	1.88%	14.31%	5.19%	21.83%	0.93%	3.46%
Thu	3.98%	32.45%	2.98%	24.02%	9.4%	38.19%	5.72%	24.51%
Fri	2.62%	30.76%	2.24%	22.75%	4.25%	27.41%	0.02%	0%
Sat	1.09%	16.52%	-0.23%	-3.72%	1.95%	28.08%	-0.93%	-8.96%
Sun	1.88%	25.65%	1.11%	10.96%	3.72%	40.2%	3.1%	20.78%
Mean	2.54%	25.86%	1.61%	14.63%	5.20%	31.04%	1.72%	8.31%

Table 3.16: Profit and lost demand comparison (data set: *Capital Bikeshare*, 2nd quarter of 2013).

Table (3.16) shows the percentage gain in profit and the percentage reduction in lost demand in comparison with the two benchmarks for the second quarter. Our approach is able to reduce the lost demand in all the cases by at least 16%, while the profit is improved by an average of 2.5% over the static repositioning. Our approach always almost outperforms the online heuristic also. We are able to reduce the lost

demand by an average of 10%, while the profit is improved by an average of 1.5% in comparison with the online heuristic.

Table (3.17) shows the comparison with the two benchmarks for the third quarter. It is the busiest quarter in the year. For this quarter, our approach is able to reduce the lost demand by an average of 20%, while the profit is improved by an average of 3% over the static repositioning. Our approach also always performs better than the online heuristic for this quarter. Our approach is able to reduce the lost demand by an average of 13%, while the profit is improved by an average of 2% over the online heuristic. Moreover, these results show the strength of our approach in the presence of high customer demand.

	Whole day (5AM-12AM)				Peak period (5AM-12PM)			
	Gain over static repositioning		Gain over online heuristic		Gain over static repositioning		Gain over online heuristic	
	Profit gain	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**
Mon	2.22%	20.81%	1.17%	9.62%	5.72%	27.11%	2.66%	10.2%
Tue	3.29%	25.79%	2.49%	18.01%	7.13%	29.45%	0.91%	5.79%
Wed	3.62%	28.06%	2.86%	21.29%	8.37%	34.59%	4.6%	21.43%
Thu	3.09%	30.58%	2.46%	23.05%	7.31%	35.89%	4%	21.1%
Fri	1.98%	26.69%	1.18%	13.45%	3.85%	26.43%	1.02%	2.97%
Sat	2.52%	31.18%	1.87%	17.25%	4.5%	49%	2.27%	21.12%
Sun	1.58%	26.95%	0.7%	8.54%	4.07%	40.38%	1.82%	14.59%
Mean	2.61%	27.15%	1.82%	15.89%	5.85%	34.69%	2.47%	13.89%

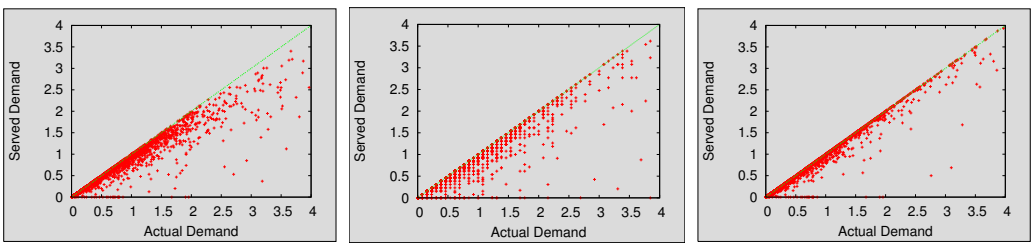
Table 3.17: Profit and lost demand comparison (data set: *Capital Bikeshare*, 3rd quarter of 2013).

Table (3.18) shows the percentage gain in profit and the percentage reduction in lost demand in comparison with the two benchmarks for the last quarter. For this data set, our approach reduces the lost demand by at least 12% over the static repositioning, while in comparison with the online heuristic our approach reduces the lost demand by an average of 5%. For all of these quarters, the percentage gain in profit in the peak hours is almost double because most of the lost demand occur during this period.

	Whole day (5AM-12AM)				Peak period (5AM-12PM)			
	Gain over static repositioning		Gain over online heuristic		Gain over static repositioning		Gain over online heuristic	
	Profit gain	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**	**Profit gain**	**Lost demand reduced**
Mon	1.58%	15.82%	1.46%	12.86%	3.11%	19.32%	1.52%	9.09%
Tue	0.75%	12.96%	1.09%	3.75%	2.73%	19.4%	0.3%	2.14%
Wed	3.3%	25.53%	3.53%	23.38%	6.82%	29.37%	3.96%	17.12%
Thu	1.67%	16.48%	0.58%	3.99%	4.36%	22.49%	-1.27%	-5.97%
Fri	0.88%	15.98%	0.73%	1.85%	2.18%	19.28%	-1.92%	-6.08%
Sat	1.19%	12.48%	0.31%	-8.07%	0.26%	6.71%	-0.93%	-18.8%
Sun	0.29%	11.51%	-0.4%	-0.62%	-1.62%	-3.15%	-2.12%	-16.96%
Mean	1.38%	15.82%	1.04%	5.31%	2.55%	16.20%	-0.07%	-2.78%

Table 3.18: Profit and lost demand comparison (data set: *Capital Bikeshare*, 4th quarter of 2013).

Correlation between supply and demand: Lastly, to visualise the effect of repositioning, we draw the correlation between the actual demand and the served demand over the entire planning horizon. Figure (3.7) shows the correlation between the actual demand and the demand served by following the three approaches. Each point in the graphs corresponds to the values of an actual demand and its corresponding served demand for all time steps and in all stations in the *Hubway* data set. Therefore, it is better if more points are closer to the identity line ($x = y$). As can be noted, our approach has significantly more points closer to the identity line than the two benchmarks and therefore, is able to better match the supply of bikes with the demand for bikes.



(a) (b) (c)

Figure 3.7: Correlation of demand and supply: (a) static repositioning, (b) repositioning using online heuristic, and (c) dynamic repositioning.

3.6.2.2 Sensitivity Results over Different Demand Scenarios Generated from Poisson Distribution

We now demonstrate the sensitivity of our approach with respect to different demand scenarios. We created ten demand scenarios for each of the weekdays from the underlying Poisson distribution with satisfied demand as the mean (refer to Section 3.5 for details). For each demand scenario, we calculate the profit and lost demand for the benchmark approaches and our approach. Figures (3.8)-(3.9) show the mean along with error bars for profit and lost demand for the four quarters of 2013 of the *Capital Bikeshare* data set. The key observations are as follows:

- Our approach (dynamic) is able to provide significantly better results with respect to reduction in lost demand than the two benchmarks on almost all the cases.

- The only cases where the online approach performs better than our approach with respect to lost demand is in quarters 1 and 4 (where the demand was significantly lower than in quarters 2 and 3) and specifically on weekends. On weekends, there is higher variance and inconsistency in demand, so our solution computed using the average demand is unable to adapt as well as the online approach.

- In terms of profit, while the difference is small, our approach is always better than the two benchmarks.

Therefore, even considering the variance, our approach provides a significant reduction in lost demand compared to the two benchmarks.

3.6.2.3 Sensitivity Results with respect to Additional Unknown Demand

For each day of the week, we evaluate our solution when demand scenarios are modified to include artificial demand. We generate our solution by considering the mean of the historical trip data that does not consider the additional artificial

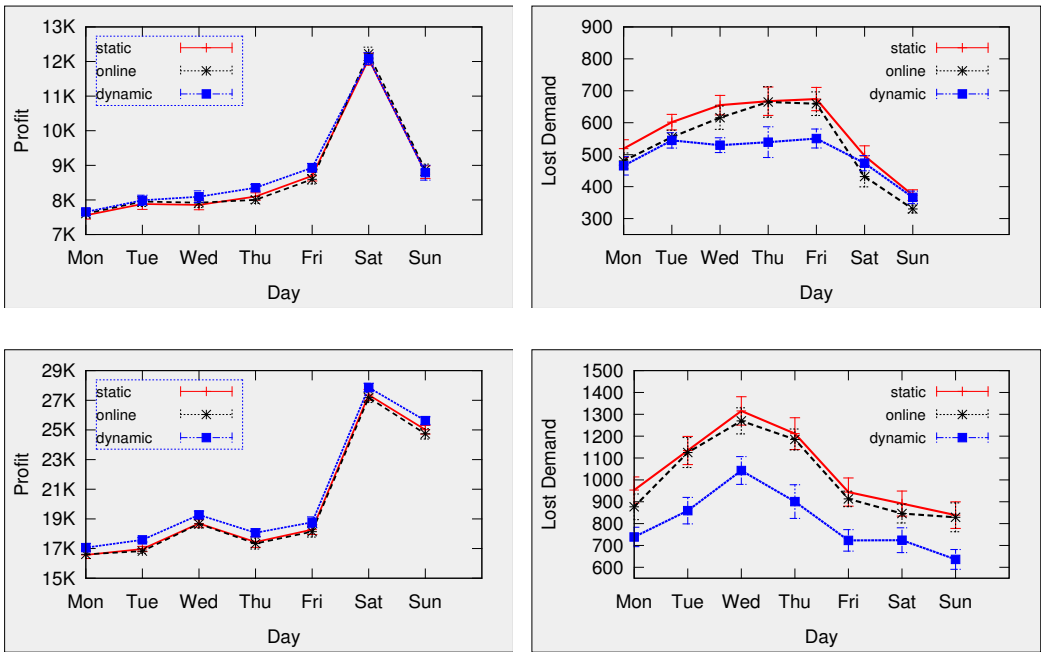


Figure 3.8: Sensitivity analysis (data set: *Capital Bikeshare*, 1st and 2nd quarters of 2013): (a) profit comparison, and (b) lost demand comparison.

demand. This artificial demand is added to a station in a time step, if that station was observed to be empty at that time interval in the data. Specifically, if a station s is observed to be empty at time step t on one day, then $\alpha\%$ of the mean served demand, F_s^t is added to that station. The destination station and booking period for the newly generated demand are chosen based on the distribution observed in the historical data.

In Table (3.19), we provide the comparison results between our approach, static repositioning and online heuristic for one of the weekdays where we vary α from 10% to 100%. The most important result is that even at 100% increase in demand at the empty stations, our dynamic approach performs better by at least 5% in terms of reducing lost demand. Furthermore, the drop in performance as unknown demand increases is gradual.

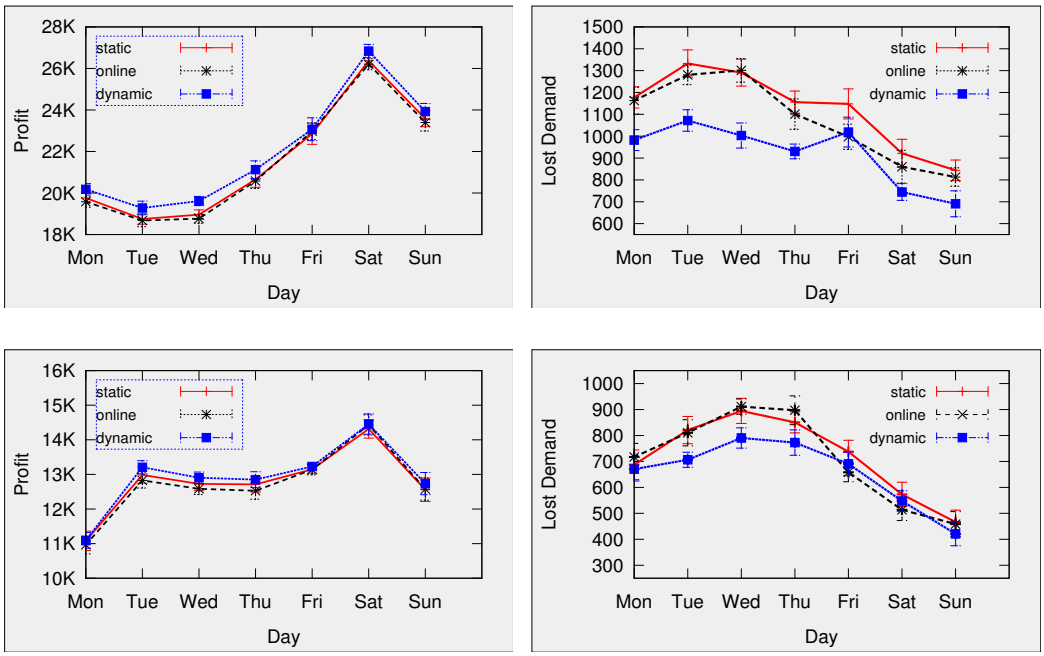


Figure 3.9: Sensitivity analysis (data set: *Capital Bikeshare*, 3rd and 4th quarters of 2013): (a) profit comparison, and (b) lost demand comparison.

	Gain over static repositioning		Gain over online heuristic	
$\alpha\%$	**Profit gain**	**Lost demand reduction**	**Profit gain**	**Lost demand reduction**
10	1.54%	23.86%	0.95%	15.73%
20	1.55%	23.78%	0.95%	15.06%
30	1.54%	23.45%	0.82%	13.97%
40	1.54%	22.78%	0.75%	12.77%
50	1.53%	22.34%	0.73%	11.89%
60	1.42%	20.81%	0.55%	9.83%
70	1.32%	19.97%	0.46%	8.73%
80	1.27%	18.76%	0.42%	7.78%
90	1.3%	18.37%	0.37%	6.75%
100	1.26%	16.54%	0.28%	4.8%

Table 3.19: Sensitivity analysis with respect to unknown increase in mean demand (data set: *Capital Bikeshare*).

3.6.2.4 Sensitivity Results with respect to Additional Known Demand

Predicting unobserved lost demand is a challenging issue in many real-world planning problems including retail planning. Many heuristic methods are mentioned in the literature (Kök & Fisher, 2007; Musalem, Olivares, Bradlow, Terwiesch, &

Corsten, 2010; Vulcano, Van Ryzin, & Ratliff, 2012) to predict the unobserved lost demand. Our repositioning approach is not dependent on the method employed to predict mean demand. So, we can always complement our approach with the best approach from the literature for predicting the mean demand.

We could also apply simple heuristics to learn demand values over time. We can consider small increments in mean demand for those stations and time steps when they become empty. For example, if station X typically becomes empty (say observed over a month) at a time step, we then consider the mean demand for station X as 102% of the realised demand at that time step. Over time, if we still observe that the station becomes empty at that time step, then we consider a mean demand that is further 2% over the realised demand. Such an approach over time will converge to the actual demand.

Furthermore, some bike sharing systems (e.g., *Bixi* in Montreal) are considering an operational enhancement that will further alleviate the problem of identifying the actual demand. In this enhancement, if customers encounter an empty or congested station, there is a provision for them to enter this information in the system that is installed at each of the base stations (assuming there is an incentive for riders to provide their information). With this minor operational enhancement, the accuracy of actual demand will increase significantly and our approach will benefit from higher accuracy on predicting the exact demand values.

In order to demonstrate generality, we now provide a detailed comparison between our solution, static repositioning and online heuristic by assuming that the extra demand is known a priori using one of the methods provided in the previous paragraphs. We employ the same mechanism to introduce extra artificial demand using α parameter as described in Section 3.6.2.3. However, since the demand is known beforehand, it is taken into consideration in our approach as well as in the online heuristic to compute repositioning and routing strategies. In Table (3.20), we provide the comparison results with respect to profit and lost demand while α is varied from 10% to 100%. As clearly shown in Table (3.20), our approach provides

	Gain over static repositioning		Gain over online heuristic	
$\alpha\%$	**Profit gain**	**Lost demand reduction**	**Profit gain**	**Lost demand reduction**
10	1.66%	29.69%	1.74%	27.44%
20	1.63%	30.09%	1.78%	28.94%
30	1.67%	28.24%	1.48%	22.75%
40	2.20%	34.26%	1.40%	23.33%
50	1.56%	24.57%	0.96%	16.38%
60	2.03%	32.32%	1.15%	22.03%
70	2.28%	32.61%	1.64%	25.41%
80	1.63%	25.61%	0.82%	16.64%
90	1.92%	27.64%	0.26%	11.52%
100	2.02%	22.78%	0.28%	7.30%

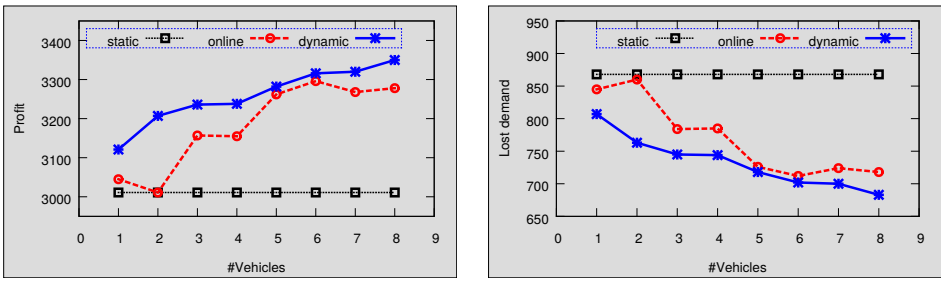
Table 3.20: Sensitivity analysis with respect to known increase in mean demand (data set: *Capital Bikeshare*).

better results for all values of α in comparison to the static and online approaches.

3.6.3 Performance Comparisons with Changes in Parameters

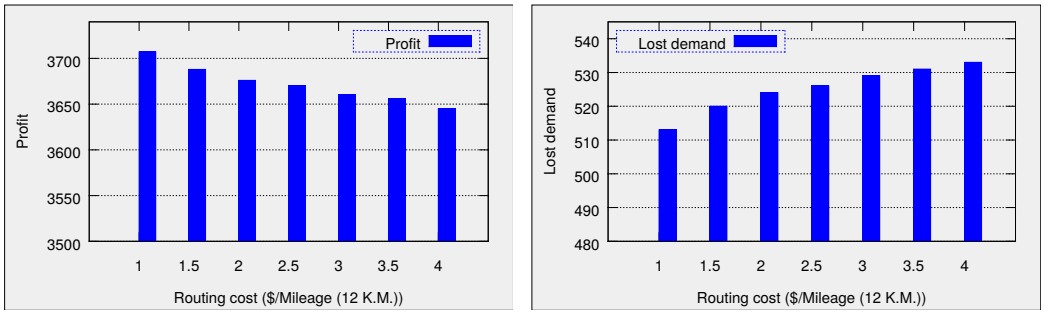
The performance of the repositioning solution is reliant upon input parameters such as the number of vehicles, unit cost for routing and duration of each time step. In this section, we describe the effect of those input parameters on key performance metrics such as profit earned by the operator and the lost demand.

Effect of the number of vehicles: To understand the effect of the number of vehicles we compare the performance of the three approaches (static, online and dynamic) with different numbers of carrier vehicles. Figure (3.10) shows the analysis of profit and lost demand on a synthetic data set with 20 stations. Figure (3.10a) shows that the profit obtained by using our approach increases monotonically as we increase the number of vehicles. Although the profit gain of our approach in comparison to the online approach fluctuates due to the myopic nature of the online heuristic, the gain is always positive. Figure (3.10b) shows a similar pattern in the performance with respect to lost demand. Lost demand reduces monotonically for our approach as the number of vehicles is increased and the gain in reducing lost



(a) (b)

Figure 3.10: Effect of the number of vehicles on (a) profit, and (b) lost demand.



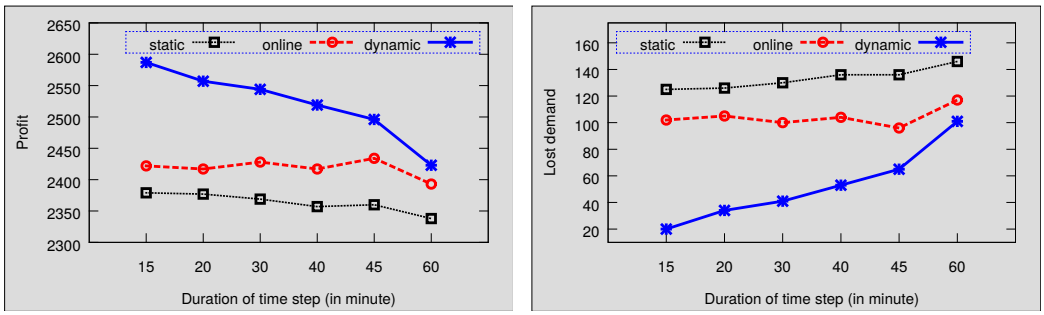
(a) (b)

Figure 3.11: Effect of routing cost on (a) profit, and (b) lost demand.

demand for our approach over both the static and the online approach is always positive.

Effect of routing cost: Routing cost, P is an important parameter in our optimisation model. While some bike sharing operators outsource the repositioning tasks to other agencies that charge a certain amount for moving individual bikes, most BSS operators use their own vehicles and the cost of repositioning is equivalent to the routing cost for the vehicles. In this section, we provide a sensitivity analysis with respect to the fuel price (i.e., dollar cost per 12 kilometers of routing which is the average mileage of vehicles as shown by Fishman et al., 2014) on a synthetic data set with 20 stations. Figure (3.11) plots the profit and lost demand when we vary the unit fuel cost from 1 dollar to 4 dollars on the X-axis. As expected, Figure (3.11a) shows that the profit earned by the operator decreases as we increase the unit cost of routing. Furthermore, Figure (3.11b) depicts that the lost demand increases by a small amount if the unit cost for routing is increased.

Effect of duration of time step: We now provide an analysis on the profit and



(a) (b)

Figure 3.12: Effect of the duration of time step on (a) profit, and (b) lost demand.

lost demand, when the duration of time step is varied. Figure (3.12) plots the performance metrics when we vary the duration of time step from 15 minutes to one hour on the X-axis. Figure (3.12a) shows that the profit for the operator reduces monotonically as we increase the duration of time step. Increasing the duration of time step entails vehicles can visit and rebalance a fewer number of base stations and therefore, produces lower profit for the operator. Figure (3.12b) shows that the lost demand increases significantly if we increase the duration of time step. Most importantly, performance gain of our approach over the static repositioning and online heuristic increases monotonically as we reduce the duration of time step. This can be attributed to our dynamic approach making better use of the extra repositioning opportunities (due to shorter duration) and promptly react to future demand changes.

On the other hand, reducing the duration of time step notably increases the runtime. For example, the runtime of the problems with 15 minutes of time step is approximately 2 hours while the problems with 30 minutes of time step are solved within 30 minutes. So, there is a clear trade-off between utility and runtime in deciding the right duration of time step. Although the performance in terms of profit and lost demand decreases by a small amount for 30 minutes of time step (over 15 minutes of time step), it provides a significant computational gain and is particularly helpful when solving large problems. Therefore, we choose 30 minute as the default setting for the duration of time step.

3.7 Model Extensions and Supplementary Analysis

In this section, we discuss ways of relaxing some of the assumptions made in the generic formulation of Table (3.3). We further provide a discussion on potential extensions.

3.7.1 Accounting for Physical Limitations in Vehicle Movement

In the MILP of Table (3.3) we assume that a vehicle can travel between any pair of stations within one time step without considering their relative distance. For the two data sets we considered, the average distance between any two stations is approximately 2 miles for *Hubway* and 5 miles for *Capital Bikeshare*[13], so our assumption of being able to travel within 30 minutes is reasonable and conservative (that accounts for the time to load and unload bikes). However, in other settings, it may not be the case and there might be multiple time steps needed to cover certain stations.

In this segment, we provide a minor update to the previous formulation which is able to account for physical limitations in vehicle movement. We introduce a new set, $\mathcal{B}_s^{\hat{t}}$ to capture the physical reachability of stations in a certain number of time steps. Specifically, $\mathcal{B}_s^{\hat{t}}$ denotes the set of stations which can be reached within $\hat{t}$ time steps from station s. The modified optimisation model is provided in Table (3.21). Amongst the constraints that consider vehicle movements in the original formulation of Table (3.3), the ones that must be modified due to physical limitations of vehicle movements are the vehicle flow conservation constraints (3.5). Essentially, the change in updated constraints (3.28) reflects the fact that we only need to consider vehicle transitions from stations that are reachable in a given number of time steps (and not others).

[13]The maximum distance between any two stations was 18 miles and 90th percentile of the distances between any two stations are within 10 miles for both the data sets. So, even in the worst case any two stations could be covered within 30 minutes. Furthermore, our solutions (rather any good solution) would not recommend a carrier vehicle to travel the maximum distance to shift the bikes.

$$\max_{\mathbf{y}^+,\mathbf{y}^-,\mathbf{z}} \sum_{t,k,s,s'} R^{t,k}_{s,s'} \cdot x^{t,k}_{s,s'} - \sum_{t,v,s,s'} P_{s,s'} \cdot z^t_{s,s',v}$$

s.t. *Constraints* (3.2)- (3.4) *hold*

$$\sum_{k\in\mathcal{S}} z^t_{s,k,v} - \sum_{\hat{t}} \sum_{k\in\mathcal{B}^{\hat{t}}_s} z^{t-\hat{t}}_{k,s,v} = \sigma^t_{v,s}, \qquad \forall s,t,v \quad (3.28)$$

Constraints (3.6)- (3.9) *hold*

Table 3.21: RESTRICTEDDRRP()

Accounting for physical limitations potentially entails finer division of time steps and hence the number of time steps increases. However, the number of transitions between stations at any one time step is reduced. Therefore, as we show below, accounting for physical limitations does not have a significant effect on the scalability of our approach. Since the inherent assumption of reachability is different, we primarily compare the runtimes in Table (3.22) to verify the claim on scalability.

	Runtime with physical limitations	Runtime without physical limitations
Mon	5120	4880
Tue	5183	3951
Wed	4136	4966
Thu	5245	4980
Fri	5127	3992
Average	4962	4554

Table 3.22: Effect of physical limitations in vehicle movements on runtime (in seconds).

We consider the *Hubway* data set to run the scalability experiments. When considering physical limitations, we assume that all the stations can be reached within 3 time steps at the maximum and the number of stations reachable in one time step from any given station is decided based on their relative distance. When considering no physical limitations, we assume that all stations are reachable from any other station in one time step. We observe that the approach considering physical limitations usually takes longer, however, the difference is not significant and consistent. On average the approach considering physical limitations takes 10% more time to

find a solution.

Another simple mechanism that can be adopted to deal with physical limitations without making changes to our approach is based on clustering of stations. We can cluster stations that can all be reached in one time step into one zone and assign a set of vehicles to that zone. This way, we can apply our method directly to each zone.

A minor modification to the MILP of Table (3.21) can be used to represent vehicles taking different times to move between stations at different times of the day. For instance, during peak hours, a vehicle might take longer to move between stations in the city. To model such scenario, we need to replace the set $B_s^{\hat{t}}$ with set $B_{t,s}^{\hat{t}}$ which contains all the stations that can be reached within $\hat{t}$ time steps if a vehicle starts from station s at time step t. The only modification required in the optimisation model is in the constraints (3.28). This is to compute the inflow of vehicles at station s at time step t by considering all the stations from where a vehicle should take $\hat{t}$ time steps to reach station s if it has started its journey at time step $t - \hat{t}$ (i.e., all the elements of the set $B_{t-\hat{t},s}^{\hat{t}}$).

3.7.2 Different Time Scales for Vehicle and Bike Movements

Our original formulation in Table (3.3) assumes that the time scale for customer movements of bikes and vehicle movements is the same. In practice, a vehicle can reposition bikes from multiple stations in each time step. Therefore, we now provide a general formulation in Table (3.23), where the bikes and vehicles are operating on different time scales. Except for the two different time scales where a vehicle is assumed to travel m stations at each time step (i.e., $t = m \cdot \hat{t}$), the structure of the formulation is similar to the one in Table (3.3). Therefore, the enhancements provided with respect to decomposition and abstraction are applicable in similar ways. We further note that the opposite case where the time scale of the movements of bikes is smaller than the one for the vehicles can basically be solved by our

$$\min_{\mathbf{y}^+,\mathbf{y}^-,\mathbf{z}} -\sum_{t,k,s,s'} R^{t,k}_{s,s'} \cdot x^{t,k}_{s,s'} + \sum_{\hat{t},v,s,s'} P_{s,s'} \cdot z^{\hat{t}}_{s,s',v} \tag{3.29}$$

$$\textbf{s.t.}\quad d^{\#,t}_s + \sum_{k,\hat{s}} x^{t-k,k}_{\hat{s},s} - \sum_{k,s'} x^{t,k}_{s,s'} + \sum_{\hat{t}=m.(t)}^{m.(t+1)} \sum_{v} (y^{-,\hat{t}}_{s,v} - y^{+,\hat{t}}_{s,v}) = d^{\#,t+1}_s, \quad \forall t, s \tag{3.30}$$

$$x^{t,k}_{s,s'} \leq d^{\#,t}_s \cdot \frac{F^{t,k}_{s,s'}}{\sum_{k,\hat{s}} F^{t,k}_{s,\hat{s}}}, \qquad \forall t, k, s, s' \tag{3.31}$$

$$d^{*,\hat{t}}_v + \sum_{s \in S} [(y^{+,\hat{t}}_{s,v} - y^{-,\hat{t}}_{s,v})] = d^{*,\hat{t}+1}_v, \qquad \forall \hat{t}, v \tag{3.32}$$

$$\sum_{k \in \mathcal{S}} z^{\hat{t}}_{s,k,v} - \sum_{k \in \mathcal{S}} z^{\hat{t}-1}_{k,s,v} = \sigma^{\hat{t}}_{v,s}, \qquad \forall \hat{t}, s, v \tag{3.33}$$

$$\sum_{j \in \mathcal{S}, v \in \mathcal{V}} z^{\hat{t}}_{s,j,v} \leq 1, \qquad \forall \hat{t}, s \tag{3.34}$$

$$y^{+,\hat{t}}_{s,v} + y^{-,\hat{t}}_{s,v} \leq C^*_v \cdot \sum_i z^{\hat{t}}_{s,i,v}, \qquad \forall \hat{t}, s, v \tag{3.35}$$

$$0 \leq x^{t,k}_{s,s'} \leq F^{t,k}_{s,s'}, 0 \leq d^{\#,t}_s \leq C^{\#}_s, 0 \leq y^{+,\hat{t}}_{s,v}, y^{-,\hat{t}}_{s,v} \leq C^*_v, 0 \leq d^{*,\hat{t}}_v \leq C^*_v \tag{3.36}$$

$$z^{\hat{t}}_{i,j,v} \in \{0, 1\} \tag{3.37}$$

Table 3.23: SOLVEDRRPDIFFTIMESCALES()

original formulation by aggregating the customer incoming and outgoing flows of bikes over the vehicle time scale. This is due to the fact that no rebalancing can be made during that interval.

3.7.3 Approximate Customer Flow Dynamics in Solution Computation

Since we maximise profit, we can identify boundary cases where bikes are not rented at certain time steps even though demand is present. Such cases can arise in our solution to save bikes for a later time step when it is possible to get higher profit. However, they do not appear in our evaluation because we have a data-driven approach where we evaluate on a test data set (that is different from training data set). Therefore, accounting for real dynamics in training data set is not always necessary. Additionally, accounting for exact dynamics increases the computational

complexity of the solution approach significantly. In our experimental results, we show that even with approximate dynamics, we are able to provide significant improvements over current practice.

To capture real dynamics, we would have to introduce new set of constraints (refer to constraints (3.38)) that ensure total outflow of bikes from station s at time step t should be equal to the minimum of total arrival demand and the number of bikes present at source station. But constraints (3.38) are quadratic in nature and our MILP becomes a higher order conic program.

$$\sum_{k,s'} x_{s,s'}^{t,k} = \min(d_s^{\#,t}, \sum_{k,s'} F_{s,s'}^{t,k}), \quad \forall t, s \tag{3.38}$$

Apart from being quadratic, as mentioned by Shu et al. (2013), constraints (3.38) can only be the sufficient condition to handle the real dynamic of BSSs if stations have unlimited bike docking capacity. Because of these difficulties, we focus on representing bike flow dynamics approximately in our optimisation model.

3.7.4 Offline Solution, Online Execution

Note that when executing the repositioning solution computed offline, the operator may find that the state of the system is different from what it was assumed to be, so the plan may not be feasible. Furthermore, this infeasibility reflects on other stations as well. For instance, the number of bikes left in the vehicle is smaller or larger than planned. We employ online modifications to deal with such situations at execution time: (1) the number of bike pickup at any time step is set as the minimum value between the number of empty slots in vehicle, the number of bikes present in the station and the number of planned pickup, and (2) the number of drop-offs at any time step is set as the minimum value between the number of bikes in the vehicle, the number of empty docks in the station and the number of planned drop-off at that time step.

As demonstrated in the sensitivity analysis results, even with such modifica-

tions to solution at execution time, our solutions are still able to provide non-trivial improvements in terms of lost demand and profit over the benchmark approaches. This is because the pattern of demand is consistent when compared over similar days (i.e., Monday pattern with another Mondays) and does not have a huge variance except on weekends. Due to this reason, there is no cascade effect when we make local changes to our solution.

3.7.5 Objective

The objective employed in SOLVEDRRP() represents a trade-off between two objectives, namely, maximising serviced demand and minimising routing cost. We combine these two objectives based on their dollar value. Specifically, we use a revenue model derived from the model employed by the real system to calculate the dollar value of serviced demand and a cost model derived from prevailing fuel costs to calculate the dollar value of routing cost. It should be noted that this is just one way of combining the two components and there can be other ways of combining the two components. In this work, we focus on this one combination of the two objectives. As shown in the experimental results, this way of combining the two components significantly improves the serviced demand and also the combined profit of the two components.

3.7.6 Labor Cost

There are two levels of decision making involved in long-term and large facility investments such as bike sharing systems: (i) strategic planning, and (ii) operational planning. Strategic level decisions consider long-term profits and typically do not change on a daily basis. Operational level decisions change on a daily basis and are our key focus. We provide a quick example to demonstrate that long-term reasoning (and not day-to-day reasoning) with respect to labor costs is a better option. The

US Department of Labor[14] provides hourly and yearly salaries for drivers operating light trucks or other delivery services in US. If we consider that a driver is hired for 6 hours (the time required to reposition bikes at the end of the day), the median cost would be 84. However, if the operator hired a driver for a year, the median salary for one day is just 80 (=29170/365). A similar result based on real statistics is available for capital costs (e.g., vehicles). This example entails that dynamic repositioning throughout the day or repositioning at the end of day would have similar labor costs. Also, since labor/capital costs would be constant in the optimisation model of Table (3.3), they would not alter the results corresponding to lost demand. We also have a buffer on the fuel cost to account for any other costs pertaining to day-to-day operations.

3.7.7 Operational Enhancement to Our Abstraction Approach

Recent bike sharing systems (e.g., *Citibike* in New York City) have introduced the concept of bike-trailers (O'Mahony & Shmoys, 2015) that can reposition a small number of bikes to nearby stations. This operational enhancement can significantly improve the performance of our geographical proximity based abstraction scheme. As the bike-trailers are only used to match the need of nearby producer and consumer stations, they can be used effectively to balance all the base stations within an abstract station. Essentially, larger vehicles are used to rebalance the system at the level of abstract stations while bike-trailers can be used to rebalance within each abstract station.

3.8 Summary

We consider the problem of dynamically repositioning bikes to improve their availability and to reduce the usage of private vehicles. The general insight that we introduce in this chapter is that, while performing repositioning, it is useful to con-

[14] http://www.bls.gov/oes/current/oes533033.htm provides the information of labor cost in US.

sider demand surges and dips during the day. To that end, we use a mixed integer linear programming approach that employs Lagrangian dual decomposition and abstraction mechanisms to provide: (1) a near optimal solution for the dynamic repositioning of idle bikes in conjunction with the routing solution for vehicles during the day, and (2) a scalable solution for the real-world large-scale bike sharing systems. The empirical results on multiple real and synthetic data sets show that our dynamic repositioning approach is not only able to achieve the original goal of reducing lost demand, but is also able to improve profit for the bike sharing system.

Chapter 4

Optimising Lost Demand in BSS through Robust Redistribution

To counter the loss in customer demand in BSSs (which causes due to the starvation or congestion of bikes at certain base stations), several bike sharing operators employ carrier vehicles to reposition bikes during the day using myopic reasoning (e.g., start filling when number of bikes falls below 20% of the capacity) to better match the demand. However, due to uncertainty in future demand, it is difficult to predict the ideal inventory level and therefore, myopic solutions often fail to provide a good quality solution. While the offline multi-step algorithms based on expected future demand that are presented in Chapter 3, are suitable for situations with stable demand patterns, they perform poorly when demand varies throughout the day. So, it is important to learn the uncertainties in demand from the data and generate robust solution that can account for all the realisation of demand scenarios.

While data driven solution approaches that consider demand uncertainty have been proposed in several application domains such as emergency medical services (Yue et al., 2012; Saisubramanian et al., 2015) and taxi fleet optimisation (Lowalekar, Varakantham, & Jaillet, 2016), progress remains slow in handling the unpredictable demand in a robust manner, particularly in case of BSSs. This serves as a motivation for us and therefore, we focus on data-driven robust optimisation techniques to

counter the uncertainties in future demand in BSSs (Ghosh, Trick, & Varakantham, 2016).

To address such scenarios where demand has high variance, we propose an online and robust redistribution approach to better match the demand and supply of bikes and consequently to reduce the expected lost demand. We refer to this problem as Dynamic Repositioning and Routing Problem under demand Uncertainty (DRRPU). We treat the problem of computing a robust solution as an iterative game between the decision maker of the BSS and the environment acting as an adversary. In each iteration, the adversary identifies a feasible demand scenario that maximises the lost demand relative to the rebalancing strategy proposed by the decision maker. From the decision maker's perspective, we solve this game using a scenario generation approach. That is to say, the decision maker takes into account all the demand scenarios generated by the adversary in previous iterations and computes a routing and repositioning solution for the vehicles that minimises the worse case lost demand over all the scenarios. The process continues until the objectives of the adversary and the decision maker converge.

We develop an online approach where the robust strategy is generated at each time step by considering the current distribution of bikes across the stations and the strategy is executed on a real-world simulator to identify the distribution of bikes for the next time step. Experimental results on multiple synthetic data sets and a real-world data set demonstrate that our approach significantly reduces the expected lost demand over the existing benchmark approaches and is robust to the uncertainty in demand.

Given the DRRPU model, our goal is to provide a repositioning and routing strategy for the vehicles at each time step that minimises the worse case lost demand. We are primarily interested in minimising lost demand that arises because of the starvation of bikes at stations. As we compute the strategy for one time step, we have no control over the lost demand that arises due to the congestion of bikes at the destination station (which depends on the unknown demand) in the next time

step. However, experimental results on the real-world data set demonstrate that repositioning bikes to reduce the lost demand at the time of hiring, determine the inventory level efficiently and furthermore, reduce the number of unsatisfied customers at the return time.

4.1 Solving DRRPU

As mentioned in Section 2.1.2, the generic model of DRRPU can be represented using the following tuple:

$$\left\langle \mathcal{S}, \mathcal{V}, \mathbf{C}^{\#}, \mathbf{C}^{*}, \mathbf{d}^{\#,0}, \mathbf{d}^{*,0}, \{\sigma_v^0\}, \mathbf{H}, \mathbf{F} \right\rangle$$

The subset of these elements which are also part of the DRRP (i.e. $\langle \mathcal{S}, \mathcal{V}, \mathbf{C}^{\#}, \mathbf{C}^{*}, \mathbf{d}^{\#,0}, \mathbf{d}^{*,0}, \{\sigma_v^0\}, \mathbf{H} \rangle$) are learnt from the BSS data sets using the same methodology as mentioned in Section 3.1. We learn the various demand bounds $\mathbf{F}$ from the historical trip data. More specific details about learning these demand bounds are mentioned later in Section 4.3.

We compute a robust repositioning and routing strategy using rolling horizon framework. In each decision epoch, for a given distribution of bikes at stations, we compute a robust strategy by assuming that the arrival demand in each station and in aggregate follows the input bounds. Once we obtain the repositioning strategy for a decision epoch, we simulate the customer flows for the given demand scenario along with the repositioning numbers to achieve the distribution of bikes across stations for the next decision epoch. This iterative process continues until we reach the last decision epoch.

For the ease of representation, we made three key assumptions: (a) Customers complete their trips in one decision epoch. That is to say, customers who hire bikes at decision epoch t should return their bikes to the destination station at the beginning of the decision epoch $t+1$; (b) Customers are impatient in nature and leave the

system if they encounter an empty station. On the other hand, they return their bikes to the nearest available station if the destination station is full; (c) The events at each time step follow a particular sequence. First, the customers return their bikes which was hired in the previous time step, then the repositioning events by the vehicles are done and lastly, the arrival customers hire bikes.

Variable	Definition
$y_{s,v}^{+,t}$	Number of bikes picked up from station s by vehicle v at time index t
$y_{s,v}^{-,t}$	Number of bikes dropped off at station s by vehicle v at time index t
$z_{s,s',v}^{t}$	Set to 1 if vehicle v has to move from station s to s' at time index t
$d_{v}^{*,t}$	Number of bikes in vehicle v at time index t
$F_{s,s'}^{k}$	Arrival customer demand from station s to s' for k^{th} demand scenario

Table 4.1: Definition of the variables.

To compute a robust strategy in each decision epoch, we propose an iterative two player game approach between the redistribution planner and an adversary. We provide two novel Mixed Integer Linear Programming (MILP) formulations to represent the planning problem for the adversary and the redistribution planner. For ease of understanding, the decision variables employed in the MILP are provided in Table (4.1).

4.1.1 The Adversarial Planner

Once the intentions of the redistribution planner are revealed, the adversary aims at providing the worst possible demand scenario that results in lowest bike usage during the planning period. More specifically, the goal is to find a demand scenario that maximises the amount of lost demand, while ensuring constraints related to demand feasibility. In the first iteration the adversary finds a worse demand scenario with the assumption of no repositioning in the system. In the subsequent iterations, the adversary plans against a particular repositioning strategy that is proposed by the redistribution planner. The MILP for the demand selection process by the adversary is shown compactly in Table (4.2). The inputs for the MILP are the current repositioning strategy, i.e., the number of bikes to pickup, Y_s^+ and drop-off, Y_s^- at

station s. The distribution of bikes, $d_s^{\#,t}$ at station s and in the decision epoch t is also provided as input. Let L_s denotes the number of lost demand occurred at station s during the planning period. $F_{s,s'}$ denotes the number of customers arrived in station s at the current decision epoch and reach station s' at the beginning of the next decision epoch.

$$\max_{\boldsymbol{F}} \quad \sum_{s} L_s \tag{4.1}$$

$$\textbf{s.t. } L_s = \max(0, \sum_{s'} F_{s,s'} - (d_s^{\#,t} + Y_s^- - Y_s^+)), \qquad \forall s \tag{4.2}$$

$$\check{F}^t \leq \sum_{s,s'} F_{s,s'} \leq \hat{F}^t \tag{4.3}$$

$$\check{F}_s^t \leq \sum_{s'} F_{s,s'} \leq \hat{F}_s^t, \qquad \forall s \tag{4.4}$$

$$\check{F}_{s,s'}^t \leq F_{s,s'} \leq \hat{F}_{s,s'}^t, \qquad \forall s, s' \tag{4.5}$$

Table 4.2: ADVERSARY($\boldsymbol{Y}^+, \boldsymbol{Y}^-, t, \boldsymbol{d}^{\#}$,drrpu)

The objective delineated in expression (4.1) is to generate a demand scenario, $\boldsymbol{F}$ that maximises the total amount of lost demand over all the stations. The number of bikes present at station s after the repositioning event can be computed as $(d_s^{\#} + Y_s^- - Y_s^+)$. Therefore, constraints (4.2) compute the lost demand at station s as the deficiency between the demand for bikes (i.e., $\sum_{s'} F_{s,s'}$) and the supply of bikes. These constraints are non-linear in nature and we linearise them with a set of inequality constraints using the well known Big-M method. Constraints (4.3-4.5) ensure that the generated demand follows the given input bounds. Specifically, constraints (4.3) ensure that the aggregated system wide demand at the decision epoch t is bounded by $\check{F}^t$ and $\hat{F}^t$. Constraints (4.4) enforce that the arrival demand in station s at decision epoch t is bounded by $\check{F}_s^t$ and $\hat{F}_s^t$. Constraints (4.5) enforce that the demand arises in station s at decision epoch t and reach station s' in the next decision epoch is bounded by $\check{F}_{s,s'}^t$ and $\hat{F}_{s,s'}^t$.

4.1.2 The Redistribution Planner

Given a set of K demand scenarios (computed by the adversary in K iterations), the goal of the redistribution planner is to find the best routing and repositioning strategy for the vehicles that maximises the bike usage or alternatively, minimises the worse case lost demand. Let $F^k_{s,s'}$ denotes the arrival demand from station s to s' for scenario k. L^k_s denotes the lost demand at station s for scenario k. The outcome of the redistribution planner is two-fold: (a) A set of decisions $\boldsymbol{z}$ for the vehicle routes; (b) The repositioning strategy $\boldsymbol{y}^+$ and $\boldsymbol{y}^-$.

The MILP for solving the joint problem of routing and repositioning is represented compactly in Table (4.3). The objective function delineated in expression (4.8) is to minimise the maximum lost demand over all the scenarios. We further simplify the objective function by introducing an additional set of constraints (4.7) to ensure that the total lost demand for scenario k is bounded by the variable λ and we minimise λ in objective function (4.6).

$$\min_{\boldsymbol{y},\boldsymbol{z}} \quad \lambda \tag{4.6}$$

$$\textbf{s.t.} \quad \lambda \geq \sum_{s} L^k_s, \qquad \forall k \tag{4.7}$$

Note that a vehicle can visit multiple stations in one decision epoch. Let a vehicle visit a maximum of $\hat{T}$ number of stations within one decision epoch. To represent the sequence of moves, we use a time index $\hat{t} \in [0, \hat{T}]$. After repositioning, the number of bikes present at station s in the decision epoch t can be computed as $(d^{\#,t}_s + \sum_{\hat{t},v}(y^{-,\hat{t}}_{s,v} - y^{+,\hat{t}}_{s,v}))$. Therefore, constraints (4.9) ensure that the lost demand at station s for scenario k is equal to the difference between the total arrival demand (i.e., $\sum_{s'} F^k_{s,s'}$) and the supply of bikes. Constraints (4.10) ensure that the total number of bikes picked up from a station s during the planning period is less than the available bikes, $d^{\#,t}_s$. Constraints (4.11) enforce that the total number of bikes dropped off at station s is less than the number of available docks, $C^{\#}_s - d^{\#,t}_s$.

$$\min_{\boldsymbol{y},\boldsymbol{z}} \max_{k} \sum_{s} L_s^k \tag{4.8}$$

$$\textbf{s.t. } L_s^k \geq \sum_{s'} F_{s,s'}^k - (d_s^{\#,t} + \sum_{\hat{t},v} (y_{s,v}^{-,\hat{t}} - y_{s,v}^{+,\hat{t}})), \qquad \forall s,k \tag{4.9}$$

$$\sum_{\hat{t},v} y_{s,v}^{+,\hat{t}} \leq d_s^{\#,t}, \qquad \forall s \tag{4.10}$$

$$\sum_{\hat{t},v} y_{s,v}^{-,\hat{t}} \leq C_s^{\#} - d_s^{\#,t}, \qquad \forall s \tag{4.11}$$

$$d_v^{*,\hat{t}} + \sum_{s \in S} [(y_{s,v}^{+,\hat{t}} - y_{s,v}^{-,\hat{t}})] = d_v^{*,\hat{t}+1}, \qquad \forall \hat{t},v \tag{4.12}$$

$$\sum_{k \in \mathcal{S}} z_{s,k,v}^{\hat{t}} - \sum_{k \in \mathcal{S}} z_{k,s,v}^{\hat{t}-1} = \sigma_v^{\hat{t}}(s), \qquad \forall \hat{t},s,v \tag{4.13}$$

$$y_{s,v}^{+,\hat{t}} + y_{s,v}^{-,\hat{t}} \leq C_v^* \cdot \sum_{i \in \mathcal{S}} z_{s,i,v}^{\hat{t}}, \qquad \forall \hat{t},s,v \tag{4.14}$$

$$\alpha \sum_{\hat{t},s,s'} H_{s,s'} z_{s,s',v}^{\hat{t}} + M \sum_{\hat{t},s} (y_{s,v}^{+,\hat{t}} + y_{s,v}^{-,\hat{t}}) \leq Q, \qquad \forall v \tag{4.15}$$

$$L_s^k \geq 0, y_{s,v}^{+,\hat{t}}, y_{s,v}^{-,\hat{t}} \leq C_v^*, d_v^{*,\hat{t}} \leq C_v^*, z_{i,j,v}^{\hat{t}} \in \{0,1\} \tag{4.16}$$

Table 4.3: REDEPLOYMENT(**F**,k,t,$\boldsymbol{d}^{\#}$,drrpu)

The initial distribution of bikes in vehicles, $d^{*,0}$ and the initial distribution of vehicles at stations, σ^0 are computed from the state of the system at the end of previous decision epoch. Constraints (4.12) ensure the flow conservation of bikes in the vehicle. The number of bikes present in vehicle v at time index $\hat{t}+1$ (i.e., $d_v^{*,\hat{t}+1}$) is equivalent to the number of bikes present in the vehicle at time index $\hat{t}$ (i.e., $d_v^{*,\hat{t}}$) plus the net incoming bikes at time index $\hat{t}$. Constraints (4.13) enforce the flow conservation of vehicles at stations by ensuring the equivalence between the inflow and outflow of vehicles in each station. For $\hat{t}=0$, depending on the initial location of vehicles, σ_v^0 these constraints ensure that vehicles move appropriately out of the initial locations. Constraints (4.14) enforce that the number of bikes picked up or dropped off is conditional to the station being visited at that time index. Let α denotes the unit for converting distance to time, M denotes the time required to pickup/drop-off one bike and Q denotes the duration of planning period. Then, constraints (4.15) enforce the physical limitation of the carrier routes. That is to

say, total time spent by the vehicles for traveling between the stations plus the time spend on picking up or dropping off the bikes, is bounded by the duration of the planning period. Finally, constraints (4.16) enforce that the number of bikes picked up or dropped off is bounded by the capacity of vehicles.

Algorithm 2: solveDRRPU(drrpu, t, $\boldsymbol{d}^{\#}$)

Initialize: $\mathbf{F} \leftarrow \{\}, \mathbf{Y}_0^+, \mathbf{Y}_0^- \leftarrow 0, i \leftarrow 0$;
repeat
  $i \leftarrow i + 1$;
  $O_a, \mathbf{F}_i \leftarrow$ ADVERSARY($\mathbf{Y}_{i-1}^+, \mathbf{Y}_{i-1}^-, t, \boldsymbol{d}^{\#}$, drrpu)
  $\mathbf{F} \leftarrow \mathbf{F} \cup \mathbf{F}_i$
  $O_r, \mathbf{z}_i, \mathbf{Y}_i^-, \mathbf{Y}_i^+ \leftarrow$REDEPLOYMENT($\mathbf{F}, i, t, \boldsymbol{d}^{\#}$, drrpu)
until $Converge$;
return $\mathbf{y}_i^+, \mathbf{y}_i^-, \mathbf{z}_i$

To better understand the robust optimisation approach, we provide the key iterative steps in Algorithm (2). The repositioning strategies are initialised as 0, therefore, in the first iteration adversary computes a demand scenario against the no repositioning strategy. From the subsequent iteration, the adversary generates a worse demand scenario against the repositioning strategy revealed by the redistribution planner. At iteration k, the redistribution planner has k demand scenarios (communicated by the adversary) and it computes a repositioning strategy that minimises the worse case lost demand over all the scenarios. The process stops when the objectives of the redistribution planner, O_r and the adversary, O_a converge. Therefore, at the convergence, the solution guarantees to provide an upper bound on the lost demand for any possible demand scenario that follows the given bounds.

4.2 Experimental Setup

We evaluate our approach with respect to key performance metric of loss in demand, on real-world[1] and synthetic data sets. The data items provided by the real-world

[1]Data is taken from Hubway bike sharing company of Boston [http://hubwaydatachallenge.org/trip-history-data]

data sets are delineated in Table (3.1). The learning of the different elements of our model for the real-world and synthetic data sets are also provided in Section 3.5. We estimate the demand bounds from the customer trip records for the real-world data sets. For synthetic data sets, we generate those bounds manually which are mentioned in details in Section 4.3..

We compare the utility of our approaches with three existing benchmark approaches; (1) **Static Repositioning** implies the practice of no repositioning during the day which is mentioned in Section 2.3.1. We use this as a baseline approach where no repositioning is done during the planning period; (2) **Myopic Repositioning** entails that bikes are repositioned at each time step to reach the inventory level to 50% of the station capacity. The details of this myopic approach is mentioned in details in Section 2.3.2; and (3) **Online Heuristic** is adapted from Schuijbroek et al. (2017). We provide the details of this approach and also how we have adapted the approach to solve the problem in our study in Section 2.3.3.

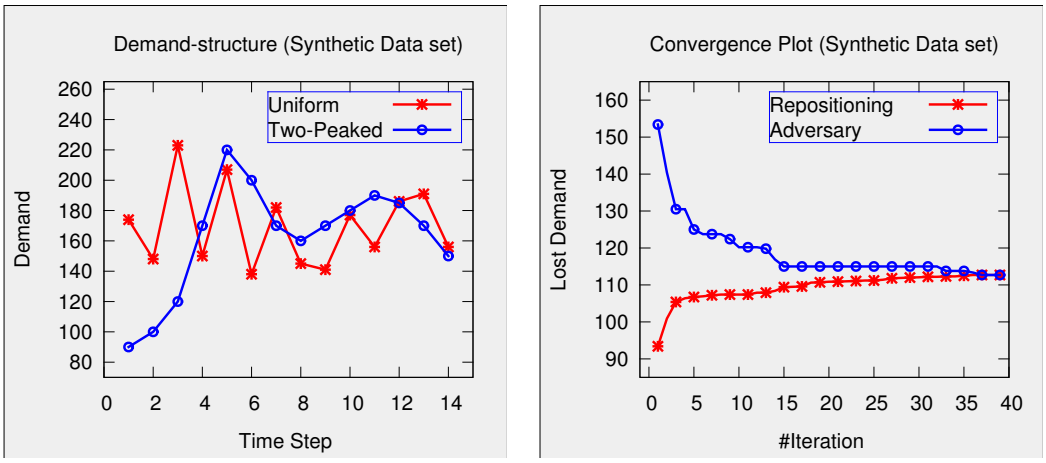
To ensure a fair comparison, all the three benchmark approaches and our robust strategy are evaluated by employing a simulation model as described in Section 2.3.1.1. Furthermore, we compute an upper bound on the optimal solution for the synthetic data sets where exact future demand for the entire horizon is assumed to be known. We employ an MILP formulation presented in Table (3.3) of Chapter 3 to compute the optimal solution.

4.3 Empirical Results

We report[2] results on two synthetic data sets. Both the data sets consist of 20 stations and 1 vehicle. We generate demand for 14 time steps. Figure 4.1(a) shows the demand patterns for both the synthetic data sets. We generate the aggregated mean demand at each time step for first data set randomly, while the aggregated mean

[2]All the linear optimisation models were solved using IBM ILOG CPLEX Optimisation Studio V12.5 on a 3.2 GHz Intel Core i5 machine with 8GB DDR3 RAM

demand for second data set follows a realistic pattern with two peak hours. For both the data sets, we compute the lower bound on the arrival demand as (1-ϵ) of the mean demand and upper bound as (1+ϵ) of the mean demand. To compute the bounds on arrival demand for each station and for each origin destination pair we set ϵ as 100%, while for the bounds on the system wide demand at each time step, ϵ is set as 10%.



(a) (b)

Figure 4.1: (a) Demand patterns for synthetic data sets; (b) Convergence of scenario generation approach on synthetic data set

Figure 4.1(b) shows the convergence of our scenario generation approach on synthetic data. As expected, the gap between the objectives of the adversary and the redistribution planner reduces monotonically and converges after 40 iterations. As both the objectives converge to 112, we can claim that the worse case lost demand is bounded by 112 if the robust strategy is adopted.

To compare the utility of our policy with the existing benchmark approaches, we generate 100 testing demand scenarios, where demand from station s to s' at time step t, $f^t_{s,s'}$ is generated using Poisson distribution with known mean parameters. We report average performance statistics in terms of mean, standard deviation and the worse case lost demand over 100 demand scenarios. The performance statistics for the synthetic data set with uniform patterns are demonstrated in Table 4.4(a). Our approach reduces the average lost demand by 22% over the static approach and by

	Static	Myopic	Online	Robust	Offline
MEAN	822	758	641	638	451
STDEV	37	47	38	38	38
MAX	938	908	713	730	521
(a) Scenarios for Uniform Data (data set: 1)					
MEAN	956	769	734	704	491
STDEV	48	62	45	48	39
MAX	1069	974	825	826	568
(b) Scenarios for Two-Peaked Data (data set: 2)					

Table 4.4: Lost demand statistics on synthetic data sets

15% over the myopic approach and is highly competitive with the online approach. Similar performance statistics for the synthetic data set with 2 peak hours are shown in Table 4.4(b). The average performance of our approach is significantly better than all the three benchmark approaches, which verify the fact that our approach is able to better handle the lost demand at rush hours. More interestingly, the competitive ratio for our solution is approximately 70% of the optimal solution for both the data sets.

Results on the Hubway data set: The next thread of results demonstrate the performance statistics on the Hubway data set. The Hubway BSS consists of 95 base stations and 3 vehicles. We consider a planning horizon of 6 hours in the morning peak (6AM-12PM) and the duration of each decision epoch is 30 minutes. We compute the bounds on demand from three months of historical trip data. 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 time step 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 time step.

We produce three threads of demand scenarios (1) We took the real demand data for 60 weekdays. We estimate the actual demand by introducing artificial demand at empty stations using a similar heuristic mechanism discussed earlier; (2) We gen-

	Lost demand at Issue Time				Lost demand at Return Time			
	Static	**Myopic**	**Online**	**Robust**	**Static**	**Myopic**	**Online**	**Robust**
MEAN	267	269	257	197	61	138	46	50
STDEV	79	77	82	68	19	32	18	15
MAX	460	471	453	393	103	196	91	96
(a) Demand scenarios from real-world data								
MEAN	145	155	146	100	53	126	37	33
STDEV	18	19	19	18	9	21	14	10
MAX	193	202	192	137	83	165	68	74
(b) Demand scenarios follow Poisson distribution at each station								
MEAN	163	171	154	113	69	143	50	54
STDEV	24	22	17	19	13	28	16	14
MAX	206	220	204	158	103	208	86	85
(c) Demand scenarios follow Poisson distribution for each OD pair								

Table 4.5: Lost demand statistics on the Hubway data set

erate 100 demand scenarios, where the arrival demand at each station is generated using Poisson distribution with the mean computed from historical data. Similar to (Shu et al., 2013), we assume that customers reach their destination station with a fixed probability; (3) We generate 100 demand scenarios, where the demand for each origin destination pair at each time step is computed using Poisson distribution.

For all the three settings of demand scenarios, we summarise the key performance statistics for all the approaches in Table (4.5). As the planning period for one decision epoch is 30 minutes, we set a time threshold of 3 minutes as a convergence criterion for our scenario generation approach. We provide statistics for two types of lost demand: (a) Lost demand occurred at the time of hiring the bikes due to starvation of bikes at stations; (b) Lost demand occurred at the time of returning the bikes due to the congestion of bikes at stations.

The performance statistics for real demand scenarios are demonstrated in Table 4.5(a). On an average our approach reduces the overall lost demand by at least 18% over all the benchmark approaches. Moreover, our approach reduces the worse case lost demand by at least 10%, hence, is robust to the uncertainty in demand. Similar performance statistics for other two settings of demand scenarios are shown

in Table 4.5(b) and 4.5(c). For both the settings, the average and worse case performance of our approach is noticeably better than all the three benchmark approaches. The average lost demand is reduced by at least 27% and 18%, while the worse case lost demand is decreased by at least 19% and 16%, over all the three benchmark approaches.

4.4 Summary

We develop a robust optimisation approach to solve the dynamic redistribution problem in bike sharing systems. We propose an iterative scenario generation approach where an adversary identifies the worse demand scenario for a given repositioning strategy and the decision maker computes a repositioning strategy by considering a set of demand scenarios proposed by the adversary. The empirical results on a real-world and multiple synthetic data sets shown that our approach outperforms the existing benchmark approaches in terms of reducing the expected and worse case lost demand and therefore, improves the operational efficiency of the bike sharing company.

Chapter 5

Optimising Lost Demand in BSS through Incentives

Bike Sharing Systems (BSSs) have been widely adopted in major cities due to its potential to mitigate the carbon emissions and traffic congestion. However, to address the problem lost demand and other issues pertinent to it, a wide variety of research papers and current systems employ the idea of repositioning idle bikes with the help of vehicles during the day (including our approaches that are presented in Chapter 3 and Chapter 4), by taking into account the movements of bikes by customers. While such a method of repositioning can help reduce imbalance, there are multiple drawbacks: (a) Vehicles incur substantial routing and labor costs; (b) More importantly, the fuel burning model of repositioning is at odds with the environment friendly nature of BSSs; and (c) Finally, due to a limited number of these vehicles, they are typically not sufficient to account for all the lost demand.

As an alternative, some BSS operators (e.g., *CitiBike* in NYC) have recently introduced the notion of bike trailers (O'Mahony & Shmoys, 2015). A bike trailer is an add-on to a bike that can carry a small number of bikes (e.g., each bike trailer can hold 3-5 bikes) and is useful to relocate bikes to nearby stations. Trailers are an environment friendly mode of repositioning the bikes. Existing research by O'Mahony and Shmoys (2015) has focussed on computing the repositioning tasks for trailers

with the assumption that dedicated staff can execute the repositioning tasks. However, given the limited budget availability, it is not economically viable to employ dedicated staff for each of the trailers.

In this chapter, we introduce a potentially self-sustaining repositioning system that addresses this Dynamic Repositioning and Routing Problem with Trailers (DR-RPT). We employ a unique combination of optimisation and mechanism design that crowdsources the repositioning tasks to the potential users while working within the budget constraints of the operator (Ghosh & Varakantham, 2017). Specifically, we provide a rolling horizon framework, where at each time step we have two components executed one after another:

1. We first employ mixed integer linear minimisation to generate potential repositioning tasks along with their valuations at the next time step.

2. We then employ an incentive compatible mechanism to crowdsource (using payment/trip based incentives) the repositioning tasks to the users who are interested in executing those tasks within the budget constraints of the operator.

There has been existing research (Singla et al., 2015; Pfrommer et al., 2014) that has focussed on providing incentives to users for assisting with repositioning. However, this line of work has primarily focussed on individual bikes and has taken a myopic (individual station) view on whether a bike is required at a station. In this work, we provide an end to end system that takes the global view (all stations) of the repositioning requirements and incentives their execution within the budget constraints.

We evaluate our system using a simulation model from Section 2.3.1.1, which is build on the realized demand scenarios from a real-world data set. At each time step the two components of the rolling horizon framework are executed on this simulator to identify the distribution of bikes for the next time step. This iterative process continues until we reach the last time step. Experimental results on a real-world data set of *Hubway* (Boston) BSS and multiple synthetic data sets demonstrate that

our approach is highly competitive in terms of reducing the expected lost demand, over the fuel burning model of repositioning.

We make the following assumptions for the ease of explanation and representation. However, these assumptions can easily be relaxed with minor modifications to our methods; (a) Users who carry bikes and trailers at decision epoch t always return their bikes at the beginning of the decision epoch t + 1; (b) Customers are impatient in nature and leave the system if they encounter an empty station. On the other hand, they return their bikes to the nearest available station if the destination station is full.

5.1 Solving DRRPT

As mentioned in Section 2.1.3, the generic model of DRRPT can be represented using the following tuple:

$$< \mathcal{S}, \mathcal{V}, \boldsymbol{C}^{\#}, \boldsymbol{C}^{*}, d^{\#,0}, \{\sigma_v^0\}, \boldsymbol{H}, \boldsymbol{F}, B >$$

The elements which are also part of the DRRP are learnt directly from the BSS data sets using the same methodology as mentioned in Section 3.1. The number of the bike trailers and their capacities are also estimated from the real-world BSS data. The initial distribution of the trailers are set randomly. We take the various demand scenarios, **F** directly from the historical trip data. Lastly, the budget for the repositioning is used as a tunable parameter in the experiments.

We propose a rolling horizon framework for solving DRRPT, where the following two components are run continuously at each time step:

- Generate repositioning tasks for the next time step;
- Mechanism to incentivize execution of tasks (within the central budget constraints) by interested users.

5.1.1 Generating Redistribution Tasks

In this section, we describe the method for computing repositioning tasks for the trailers and also estimate the valuations of those tasks from center's perspective. As a trailer can travel at most to one station in each time step (equivalent to bikes), the repositioning task for a trailer is to pickup a certain number of bikes from the neighbourhood of its origin station and drop them to another station. To formally represent the repositioning tasks, we introduce the following decision variables:

- $y^{+}_{s,v}$ denotes the number of picked up bikes by trailer v from station s;
- $y^{-}_{s,v}$ denotes the number of bikes dropped off by trailer v at station s;
- $b^{+}_{s,v}$ is a binary decision variable which is set to 1 if trailer v picks up bikes from station s and 0 otherwise;
- $b^{-}_{s,v}$ represents a binary decision variable which is set to 1 if trailer v returns bikes at station s and 0 otherwise.

In addition, we use the symbol G_v to denote the set of neighbouring stations from where vehicle v is allowed to pick up bikes. A station is included in G_v if it is situated within a threshold distance from the origin station of trailer v. Our goal is to compute the best routing and repositioning strategy for each of the bike trailers so as to minimise the total number of expected lost demand over K training demand scenarios. Let L^{k}_{s} denotes the lost demand at station s for demand scenario k, after the repositioning tasks are achieved. We represent the problem of minimising expected lost demand as a Mixed Integer Linear Programme (MILP). The MILP for the task generation is compactly represented in Table (5.1). Our objective (delineated in expression 5.1) is to minimise the expected lost demand (equivalent to total lost demand, as each demand scenario has equal probability) over all the K training scenarios. The constraints associated with this repositioning task generation are described as follows:

$$\min_{\boldsymbol{y}} \sum_{s,k} L_s^k \tag{5.1}$$

$$\textbf{s.t. } L_s^k \geq \sum_{s'} F_{s,s'}^k - \left(d_s^{\#,t} + \sum_{v} (y_{s,v}^- - y_{s,v}^+)\right) \qquad \forall k, s \tag{5.2}$$

$$y_{s,v}^+ \leq b_{s,v}^+ \cdot \min(d_s^{\#,t}, C_v^*), \qquad \forall s, v \tag{5.3}$$

$$\sum_{v} y_{s,v}^+ \leq d_s^{\#,t}, \qquad \forall s \tag{5.4}$$

$$\sum_{v} y_{s,v}^- \leq C_s^{\#} - d_s^{\#,t}, \qquad \forall s \tag{5.5}$$

$$y_{s,v}^- = b_{s,v}^- \cdot \sum_{s} y_{s,v}^+, \qquad \forall s, v \tag{5.6}$$

$$(b_{s,v}^+ + b_{s',v}^- - 1) \cdot H_{s,s'} \leq H_{max} \qquad \forall s, s', v \tag{5.7}$$

$$\sum_{s} b_{s,v}^+ = 1, \qquad \forall v \tag{5.8}$$

$$\sum_{s \notin G_v} b_{s,v}^+ = 0, \qquad \forall v \tag{5.9}$$

$$\sum_{s} b_{s,v}^- = 1, \qquad \forall v \tag{5.10}$$

$$b_{s,v}^+, b_{s,v}^- \in \{0,1\}, 0 \leq y_{s,v}^+, y_{s,v}^- \leq C_v^*, L_s^k \geq 0 \tag{5.11}$$

Table 5.1: TASKGENERATION(**F**,t,$\boldsymbol{d}^{\#}$,drrpt)

1. **Compute the lost demand as the deficiency in supply of bikes:** The number of bikes present in a station s after accomplishing the repositioning task is estimated as $d_s^{\#,t} + \sum_v (y_{s,v}^- - y_{s,v}^+)$. Therefore, constraints (5.2) ensure that the number of lost demand at station s for scenario k is lower bounded by the difference between demand and supply of bikes at that station. Note that, as we are minimising the sum of lost demand over all the scenarios, these constraints are sufficient alone to compute the exact number of loss in customer demand.

2. **Trailer capacity is not exceeded while picking up bikes:** Constraints (5.3) ensure that the number of bikes picked up by trailer v from station s is bounded by the minimum value between the number of bikes present in the station and the capacity of the trailer.

3. **Total number of bikes picked up from a station is less than the available bikes:** As multiple trailers can pick up bikes from the same station, constraints (5.4) enforce that the total number of picked up bikes by all the trailers from station s during the planning period t is bounded by the number of bikes present in the station, $d_s^{\#,t}$.

4. **Station capacity is not exceeded while dropping off bikes:** Constraints (5.5) ensure that the total number of dropped off bikes at station s is bounded by the number of available slots for bikes at that station.

5. **A trailer should return the exact number of bikes it has picked up:** Note that $b_{s,v}^-$ is the binary decision variable that controls the drop-off location for the trailer v. Therefore, constraints (5.6) enforce that the number of bikes dropped off by a trailer in a station is exactly equal to the number of picked up bikes if the station is visited.

6. **Total traveling distance for a trailer is bounded by a threshold value:** To represent the physical limitation of the route, we need to ensure that the total distance travelled by a trailer in a given planning period is within a few kilometers. Constraints (5.7) enforce this condition by ensuring that the distance between the pick up and the drop-off station for a trailer is bounded by a threshold value, H_{max}.

7. **A trailer should pick up bikes from one station only:** Constraints (5.8) enforce this condition by allowing only one pick up decision variable to be set to 1 for each trailer.

8. **The pick up location for a trailer should be within the geographical proximity of its origin station:** Constraints (5.9) assure this requirement by fixing all the pick up decision variables for trailer v to 0 for the stations which does not belong to its nearby station set, G_v.

9. **A trailer should return bikes to one station only:** Constraints (5.10) ensure this condition by allowing only one drop-off decision variable to be set to 1 for each trailer.

Note that, constraints (5.6) are non-linear in nature. However, one component in the right hand side is a binary variable. Therefore, we can easily linearize them using the following constraints (5.12)-(5.14).

$$y^-_{s,v} \leq C^*_v \cdot b^-_{s,v} \qquad \forall s, v \qquad (5.12)$$

$$y^-_{s,v} \leq \sum_s y^+_{s,v} \qquad \forall s, v \qquad (5.13)$$

$$y^-_{s,v} \geq \sum_s y^+_{s,v} - (1 - b^-_{s,v}) \cdot C^*_v \qquad \forall s, v \qquad (5.14)$$

Although we are using big-M method for the linearization, the upper bound for the pickup or drop-off variable (or alternatively the value of M) is the capacity of the trailer which is relatively small and therefore, these constraints are computationally inexpensive.

5.1.2 Mechanism to Incentivize Task Execution within Budget Constraints

Once we determine the tasks, our goal is to design a mechanism which allocates the tasks among the users who are interested in executing these tasks and generate a payment method to ensure that the users bid for the tasks truthfully. If the payment method does not ensure truthful behaviour, then either the bike sharing company is unhappy (as it pays more money to users than required) or users are unhappy (as they get paid less) while repositioning bikes through trailers.

To design a mechanism for crowdsourcing the repositioning tasks, the first step is to compute the value of the tasks from center's perspective. As our goal is to minimise the expected lost demand, the valuation of the task is proportional to the

expected lost demand reduced by the trailer job over all the training demand scenarios. Specifically, the value of task for trailer v is defined as follows (ξ represents unit value of lost demand to compute overall value):

$$U(v) = \frac{\xi}{K} \sum_{k,s} \Big[\min\big(\max(F_s^k - d_s^{\#,t}, 0), y_{s,v}^+\big) - \min\Big(\max\big(y_{s,v}^- - (d_s^{\#,t} - F_s^k), 0\big), y_{s,v}^-\Big)\Big] \quad (5.15)$$

Intuitively this value is the weighted difference in reduced lost demand using the trailer minus increase in lost demand due to moving bikes using trailer. The first term in expression (5.15) computes the expected lost demand reduced by the trailer v in its destination station over all the K scenarios. The second term computes the expected lost demand arising because of the pickup decision by the trailer v at its origin station.

We assume that the set of interested users for each pair of tasks are disjoint. One user can execute a single task in any given decision epoch, so this assumption can be easily enforced. To ensure this assumption is satisfied, we can either associate a huge penalty for bidding on multiple tasks or discard all bids of a user except the first one. Once all the bids arrive, the goal of the center is two-fold: (a) Design an incentive compatible mechanism to ensure that users bid truthfully on every task; (b) Allocate the tasks in a fashion that minimises the efficiency of the entire system while satisfying the budget feasibility.

Observation 3. *As the set of bidders for different tasks are pairwise disjoint and the mechanism initiates once all the bid information is available, the tasks are primarily independent but coupled by the central budget constraint. Therefore, the mechanism or payment method can be designed for each of the tasks separately. However, the final allocation of tasks should be accomplished in a fashion so that the budget feasibility is ensured.*

By exploiting observation (3), we design a mechanism for each of the tasks

separately. Let the set of repositioning tasks be $\mathcal{T} = \{1, ..., |\mathcal{V}|\}$. We begin the discussion with the mechanism design for a single task for trailer v. Let, $N_v = \{1, ..., n_v\}$ represents the set of rational users who are bidding privately to the center for the task of trailer v. Each user $i \in N_v$ privately reveals their type $\theta_i =< C_i(v) >$, where $C_i(v)$ denotes their private cost for executing the task of trailer v. The center's profit for the bid of user i is defined as $W_i(v) = U(v) - C_i(v)$. We reject a bid from a user i if $C_i(v) > U(v)$, which ensures that $W_i(v)$ is always positive. Our goal is to assign the task to a bidder so that the center's profit is minimised and design a payment method to ensure that users always bid truthfully. We use the standard Vickrey-Clarke-Groves [VCG] mechanism (Vickrey, 1961; Clarke, 1971; Groves, 1973) to solve this problem.

According to this mechanism, the task is always allocated to the lowest bidder, but the lowest bidder gets paid the bid of the second lowest bidder. For instance, if there are 3 bids of 10$, 12$ and 14$ to perform a repositioning task, then the task is allocated to bid 1 and the person putting in bid 1 gets paid 12$. More formally, let $\lambda^* = \{0, 1\}^{N_v}$ denotes the allocation of the task so that the center's profit is minimised.

$$\lambda_i^*(v) = \left\{ \begin{array}{ll} 1 & \textbf{if } i = \text{argmax}_{j \in N_v} W_j(v) \\ 0 & \textbf{Otherwise} \end{array} \right\}$$

Then the payment to the user i for task v is computed using the following expression:

$$P_i(v) = \lambda_i^*(v)\big[U(v) - \max_{j \neq i} W_j(v)\big] = \lambda_i^*(v)\big[\min_{j \neq i} C_j(v)\big] \tag{5.16}$$

Expression (5.16) indicates that the payment for user i is the second lowest cost revealed in the bid process if the task is allocated to him, otherwise the payment is set to 0.

Since, we directly adapt the standard VCG mechanism, the mechanism for sin-

gle task is truthful or incentive compatible. However, this does not ensure incentive compatibility over all tasks, as there is a budget constraint. We now provide a method that will ensure incentive compatibility over all tasks without violating the budget feasibility.

Ensuring the Budget Feasibility: As mentioned previously, the BSS operators work within a fix budget B. We have a set of tasks $\mathcal{T} = \{1, ..., |\mathcal{V}|\}$, where each task $i \in \mathcal{T}$ has a valuation, $U(i)$ (computed using equation 5.15) and the payment for the task is denoted by $P(i)$ (computed using equation 5.16). Our goal is to allocate the tasks that minimises the overall valuation of the center while the total payment is bounded by the given budget, B. Let $x(i)$ denotes a binary decision variable which is set to 1 if task i is allocated and 0 otherwise. We compactly represent the problem as an Integer Program (IP) in table (5.2).

$$\max_{x} \sum_{v \in \mathcal{T}} x(v) \cdot U(v) \tag{5.17}$$

$$\mathbf{s.t.} \sum_{v \in \mathcal{T}} x(v) \cdot P(v) \leq B \tag{5.18}$$

$$x(v) \in \{0, 1\} \qquad \forall v \in \mathcal{T} \tag{5.19}$$

Table 5.2: TASKALLOCATIONIP($\boldsymbol{U}, \boldsymbol{P}, \mathcal{T}, B$)

Our objective in expression (5.17) aims to find an allocation of the tasks so that the cumulative valuation from the center's perspective is minimised. Constrains (5.18) enforce that the total payment made to the users due to the resulting allocation should respect the given budget B. The IP in Table (5.2) is exactly equivalent to the 0/1 knapsack problem which is a known NP-Hard problem. However, we can employ the well-known dynamic programming approach (refer to chapter 6 of Dasgupta, Papadimitriou, and Vazirani, (2006)) to speed up the solution process. The complexity of such a DP approach is $O(|\mathcal{T}| \cdot B)$ in comparison to the brute force method that has the complexity of $O(2^{|\mathcal{T}|})$.

Proposition 2. *The mechanism for task allocation for the trailers in bike sharing*

system is incentive compatible (IC), individually rational (IR) and economically efficient (EE).

Proof: The mechanism for single task satisfies the IC and IR property as it follows the standard VCG mechanism. As all the tasks are independent and the payments are made for a subset of tasks to ensure the budget feasibility, all the allocated tasks satisfies the IC and IR property. Hence, the budget feasible mechanism for the entire BSS meets the requirements to satisfy the IC and IR property. Finally, the mechanism minimises the difference between center's valuation and the cost for executing the task which is equivalent to the expected total welfare, hence, our mechanism satisfies the EE property. ■

5.1.3 Overall Flow of Our Approach

To better understand the overall flow of our approach, we provide Algorithm (3). We begin by solving the MILP of Table (5.1) to generate the repositioning tasks for each of the trailer to better satisfy customer demand over a set of training demand scenarios. Then the values of the tasks from center's perspective are computed using equation (5.15). Next, a set of rational users bid for the tasks privately. Once all the bids are submitted, we employ the standard VCG mechanism to generate the payment (refer to equation 5.16) for each task. Finally, we allocate budget to tasks (and make payments only if the task can be allocated money) by solving a 0/1 knapsack problem that minimises the global welfare of the system without violating the budget constraints of the operator.

5.2 Experimental Setup

We conducted our experiments on a real-world data set[1] of *Hubway* BSS. The data items provided by the *Hubway* data sets are delineated in Table (3.1). The learning

[1]Data is taken from *Hubway* bike sharing company of Boston [http://hubwaydatachallenge.org/trip-history-data].

Algorithm 3: solveRepositioning(drrpt, t, $\boldsymbol{d^{\#}}$, $\boldsymbol{F^t}$, B)

Initialize: $\boldsymbol{Y}^+, \boldsymbol{Y}^- \leftarrow 0$;
$\boldsymbol{Y}^+, \boldsymbol{Y}^- \leftarrow$ TASKGENERATION($\boldsymbol{F^t}, t, \boldsymbol{d^{\#}}$, drrpt);
for *each* $v \in \mathcal{V}$ **do**
  $U(v) \leftarrow$ COMPUTETASKVALUE(Y_v^+, Y_v^-);
for *each* $v \in \mathcal{V}$ **do**
  $\boldsymbol{C}(v) \leftarrow$ COLLECTBIDS($Y_v^+, Y_v^-, U(v)$);
for *each* $v \in \mathcal{V}$ **do**
  $P(v) \leftarrow$ GENERATEPAYMENT($U(v), \boldsymbol{C}(v)$);
$\boldsymbol{X} \leftarrow$ TASKALLOCATION($\boldsymbol{U}, \boldsymbol{P}, \mathcal{V}, B$);
for *each* $v \in \mathcal{V}$ **do**
  $\boldsymbol{Y}_v^+ \leftarrow \boldsymbol{Y}_v^+ \cdot X(v)$;
  $\boldsymbol{Y}_v^- \leftarrow \boldsymbol{Y}_v^- \cdot X(v)$;
return $\boldsymbol{Y}^+, \boldsymbol{Y}^-$;

of the different elements of our model for the real-world data sets are provided in Section 3.5. Furthermore, we generate two sets of synthetic demand scenarios using *Poisson* distribution with the mean computed from real-world data set. More specific details about these synthetic data sets are mentioned in Section 5.3. We evaluate our approach with respect to the key performance metric of loss in customer demand.

We compare the utility of our approach with two existing benchmark approaches: (1) **Static Repositioning** implies the practice of no repositioning during the day and the details are mentioned in Section 2.3.1; and (2) **Dynamic Repositioning** implies the practice of repositioning idle bikes using vehicles during the day to meet the expected future demand. We adapted a modified version of the scenario based approach from Chapter 4. In the approach we presented in Chapter 4, a worse demand scenario is generated in each iteration to counter the repositioning strategy of the current iteration and then they produce a robust repositioning solution by considering all the previously generated scenarios. However, for a fair comparison with our approach (as shown in Table 5.1), we make the following modifications in their minimisation model: (1) We take the exact set of training demand scenarios used in our approach rather than generating the worse scenarios; (2) We minimise the

total number of lost demand over all the demand scenarios (equivalent to our objective function of Table 5.1) in contrast to minimising for the worst case. For a fair comparison of our approach against the existing benchmark approaches, all the benchmark approaches and our approach are evaluated on the simulation model described in Section 2.3.1.1.

5.3 Empirical Results

We now show the performance[2] of our approach on *Hubway* data set. The *Hubway* BSS consists of 95 base stations and 3 vehicles. We study with 10 trailers and their capacity is assumed to be three in our default settings of experiments. We take 6 hours of planning horizon in the morning peak (6AM-12PM) and the duration of each decision epoch is considered as 30 minutes. The demand scenarios are generated from three months of historical trip data. As the trip data only contains successful bookings and does not capture the unobserved lost demand, we employ a micro-simulation model with 1 minute of time step to determine the time slots when a station was empty and introduce artificial demand at the empty station based on the observed demand at that station in previous time step. We demonstrate three sets of results on the *Hubway* data set:

- The performance comparison between our approach and the benchmark approaches in terms of reducing the lost demand;
- The effect of key tuneable input parameters on the mechanism design over a wide range of demand scenarios;
- Runtime performance of our approach.

Performance comparison: To evaluate the performance of our approach, we produce three types of demand scenarios: (1) We took the real demand data for

[2]All the linear minimisation models were solved using IBM ILOG CPLEX minimisation Studio V12.5 incorporated within python code.

60 weekdays. The actual demand is estimated by introducing artificial demand at empty stations using a similar heuristic as discussed earlier. We use 20 days of demand scenarios for training purpose and tested the strategy on other 40 days of demand; (2) We generate 100 demand scenarios, where the arrival demand at each station is generated using *Poisson* distribution with the mean computed from historical data. Similar to (Shu et al., 2013), we assume that customers reach their destination station with a fixed probability; (3) We generate 100 demand scenarios, where demand for each origin destination [OD] pair at each time step is computed using *Poisson* distribution. In demand scenario types 2 and 3, we utilise 30 demand scenarios for training and 70 demand scenarios for testing.

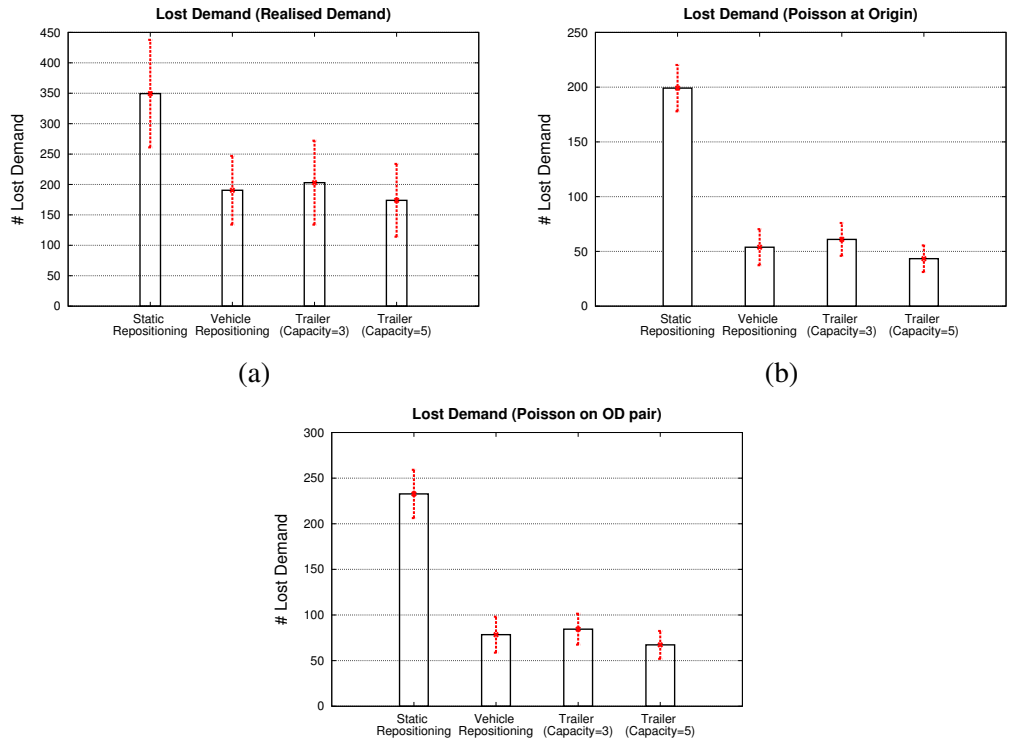


Figure 5.1: Lost demand statistics for (a) Demand scenarios from real-world data; (b) Demand scenarios follow *Poisson* distribution at origin station; (c) Demand scenarios follow *Poisson* distribution for each OD pair.

For all the three types of demand scenarios, we compute the cumulative lost demand at the time of bike pickup and return for the following four approaches: (a) Static repositioning where no rebalancing is done during the planning period; (b)

Dynamic repositioning using 3 existing vehicles; (c) Dynamic repositioning using 10 trailers, each having a capacity of 3; (d) Dynamic repositioning using 10 trailers, each having a capacity of 5. For this set of experiments, we directly took the repositioning solution from Table (5.1) for evaluation. Specifically, we assume that there is sufficient budget available to allocate all the tasks. Figure (5.1) depicts the average number of lost demand along with standard deviation for all the three types of demand scenarios. Figure 5.1(a) shows the lost demand statistics on the real-world demand scenarios. By utilising trailers with capacity 3, the average lost demand over 40 testing scenarios reduces by 41% over the no repositioning approach. The repositioning solutions for the trailers with capacity 3 are also proven to be highly competitive to the solutions achieved by vehicles. As expected, the repositioning solutions for the trailers with capacity 5 produce better results and outperform the lost demand obtained by 3 carrier vehicles. Similar performance statistics are shown in Figure 5.1(b) and 5.1(c) for the demand scenarios generated using *Poisson* distribution at origin station and for each OD pair respectively. In both the settings, we observe a consistent pattern that the repositioning solution using trailers with capacity 3 reduces the average lost demand over 70 test scenarios by 69% and 63% in comparison to the baseline approach. Moreover, both the figures clearly demonstrate that the solutions for trailers with capacity 5 are better than the fuel burning mode of repositioning by the vehicles.

Effect of tuneable parameters: In the next set of results we demonstrate the performance of our approach by varying the different input parameters of the mechanism. We employ the real-world demand scenarios (demand scenario type 1) for this experiments, where 20 demand scenarios are used for training and the evaluation is done on other 40 scenarios. The outcome of the mechanism depends on the following three input parameters:

- Hourly budget allocated by the operators (β) - Ideally the BSS operators allocate a fixed amount beforehand for the repositioning tasks. In our default

settings of experiments we have fixed the hourly budget to 50 dollars[3];

- Percentage of users interested in trailer tasks (γ) - To execute a mechanism, it is important to compute the number of users bidding for each trailer task. Typically, a certain percentage of users whose origin and destination location is within Δ kilometer of the pickup and drop-off location of the trailer, are the potential users interested in executing the task and bid for it. In our experiments we set the value of Δ to 1 kilometer[4]. We use the default value of γ as 30%;

- Ratio of lower and upper bound of bids (α) - The third and most important parameter for the mechanism is the bid values submitted privately by the users. We generate the bid values using *Normal* distribution from the range $[C_{\text{min}}, C_{\text{max}}]$. As the upper limit of the bid value for task v is bounded by its valuation $U(v)$, we set the C_{max} for the task of trailer v to $U(v)$. The value of C_{min} is set to αC_{max}. As the bids are generated from a distribution, we run the mechanism 100 times for every task and use the expectation over 100 runs as the payment. The default value of α is set to 30%.

Figure (5.2) depicts the effect of the tuneable parameters on the performance of our approach. Figure 5.2(a) plots the average lost demand over 40 days of testing demand scenarios, where we vary the hourly budget (β) in the X-axis from 10 dollars to 80 dollars. As expected, the average number of lost demand reduces monotonically as we increase the hourly budget. Due to the randomness in bid values in different runs, the lost demand might increase for some scenarios, even after increasing the hourly budget. We observe that for more than 78% of the cases, lost demand decreases if we increase the hourly budget by 10 dollars.

[3]The link: http://www.bls.gov/oes/current/oes533033.htm from the US Department of Labor provides hourly salaries for drivers operating light trucks that are used for repositioning the bikes. It shows that the median hourly cost for a hired driver is 14 dollars. Therefore, the hourly budget for 3 existing vehicles including the fuel cost for routing would be around 50 dollars.

[4]We experimented with Δ as 0.5 kilometer and observe similar results as shown in Figure 5.2.

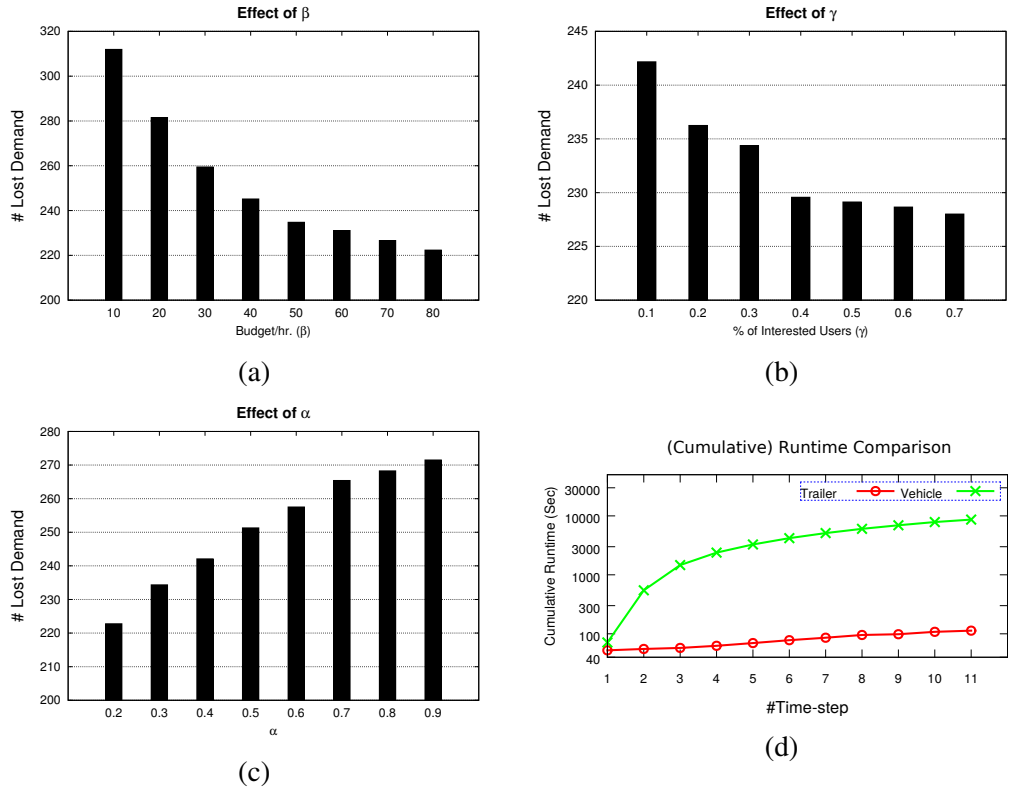


Figure 5.2: (Average) Lost demand statistics for varying (a) Allocated budget ($\boldsymbol{\beta}$) [$\alpha = 0.3, \gamma = 0.3$]; (b) Percentage of users interested in trailer tasks ($\boldsymbol{\gamma}$) [$\alpha = 0.3, \beta = 50$]; and (c) Ratio of lower and upper bound of bids ($\boldsymbol{\alpha}$) [$\beta = 50, \gamma = 0.3$]. (d) (Cumulative) Runtime comparison between the repositioning solutions of vehicles and trailers.

Figure 5.2(b) plots the average lost demand over 40 testing demand scenarios, when we vary the interest rate of the users (γ) in the X-axis from 10% to 70%. The average number of lost demand reduces monotonically as we increase the interest rate of users, because increasing the interest rate implies that additional bids are submitted to the center and therefore, the likelihood of the payment value reduces which in turn enable us to execute extra trailer tasks within the given budget and hence, the number of expected lost demand reduces. We observe that the lost demand decreases for around 60% of the cases, if we increase γ by 10%.

Figure 5.2(c) plots the average lost demand over 40 testing demand scenarios, where we vary the ratio of the lower and upper bound of the bids (α) in the X-axis from 20% to 90%. Increasing the value of α indicates that the lower bound of the bids increases and so the expected bid value also increases. Now, increasing the bid values implies that the expected payment for the tasks also increases and the number

of tasks that can be executed within a fixed budget decreases, hence the number of expected lost demand also increases. As expected, figure 5.2(c) clearly depicts that the average number of lost demand increases monotonically as we increase the value of α. For around 74% of the cases, the lost demand increases if we increase α by 10%.

Runtime performance: In the last set of results, we show the runtime performance of our approach in comparison to the repositioning solution of the vehicles on the real-world demand scenarios. The time to find a repositioning solution is a crucial factor in our settings, as we are generating the strategy after every 30 minutes of interval. Figure 5.2(d) depicts the runtime performance where in the X-axis we vary the number of decision epochs and the Y-axis denotes the cumulative runtime in logarithmic scale. For every value of decision epoch, our approach was able to solve the problem within a couple of seconds with 20 training scenarios. On the other hand, it took more than 15 minutes for each decision epoch to generate the repositioning solutions for the vehicles with the same number of training scenarios.

5.4 Summary

In this chapter, we explore the dynamic repositioning problem in bike sharing systems with the help of bike trailers. We propose a novel optimisation model to generate the repositioning tasks for trailers to better meet the customer demand. Additionally, we design a budget feasible incentive compatible (incentivizes truth telling) mechanism to crowdsource the tasks among the users who are interested in executing those tasks. The empirical results on a real-world data set show that our green mode of repositioning is economically viable and highly efficient in terms of reducing the lost demand.

Chapter 6

Models and Background for Decision Making in EMS

In this chapter, we explain the models and existing research related to the decision making in Emergency Medical Systems (EMSs). The operational decisions in EMS correspond to allocation of an entire fleet of ambulances to a given set of base stations. To the best of our knowledge, there is no existing research paper that formally solve the strategic decision making problem in EMS. However, the existing models for solving the operational decision making problem are equivalent to strategic decisions if we allow to set the bases in all the possible base locations. Therefore, we introduce the existing approaches for solving the ambulance allocation problem for a given set of bases to better match the emergency requests to the supply of ambulances in EMS.

6.1 Models

For the strategic decisions in EMS, we consider two widely adopted objectives by the real-world EMS operators: (a) Maximise number of requests that are satisfied within a given threshold response time (ex: 15 minutes), referred to as *Bounded Time Response*; and (b) Minimise the response time for a fixed percentage (ex:

Element	Definition
r	The indicator used to represent an emergency request.
l	The indicator used to represent a base station.
$\mathcal{R}$	The set of emergency incidents.
$\mathcal{B}$	The set of possible base stations.
$\mathcal{A}$	A fleet of ambulances.
$T_{l,l'}$	Ambulance travel time between location l and l'.
C_l	The capacity of base station l.
L_{rl}	The utility value for assisting request r from base l.
Δ	The pre-defined threshold response time.
α	The percentage of requests whose response times can exceed the threshold response time.
P_l^r	The set of parent requests of request r for base l.

Table 6.1: EMS notations and their definition.

80%) of requests, referred to as *Bounded Risk Response*. In this section, I briefly introduce the notations and models for these two categories of objectives for the strategic decision making in EMS. For the ease of understanding, the key notations and their definitions are compactly represented in Table (6.1).

6.1.1 Model: Bounded Time Response

We now formally describe the generic model for the strategic decision making problem in EMS, where our goal is to optimise the bounded time response. We employ the following tuple:

$$< \mathcal{R}, \mathcal{B}, \mathcal{A}, \boldsymbol{T}, \boldsymbol{C}, L >$$

A brief description of these elements are shown in Table (6.1). $\mathcal{R}$ denotes a set of emergency requests, where each request $r \in \mathcal{R}$ is tagged with a tuple $< t, s, h >$. t is the arrival time, s is origin location and h is destination hospital of the request r. $\mathcal{B}$ denotes the set of possible base stations and $\mathcal{A}$ represents a fleet of ambulances. $\boldsymbol{T}$ is a two-dimensional matrix that provides travel time between any two base locations. More specifically, T_{l_1,l_2} is the time required to move from source location l_1 to destination l_2. $\boldsymbol{C}$ denotes the capacities of the bases, where C_l denotes the maximum number of ambulances that can be allocated to base l. L is the utility function which

will be explained in detail in Section 6.3.1. Given these input elements, our goal is to find a subset of $\mathcal{B}$ and to optimally allocate an entire fleet of ambulances to these bases that maximises the percentage of requests which can be served within a threshold time bound.

6.1.2 Model: Bounded Risk Response

In this section, we formally describe the generic model for strategic decision making problem in EMS, where our goal is to optimise the bounded risk response. We employ the following tuple:

$$< \mathcal{R}, \mathcal{B}, \mathcal{A}, \boldsymbol{T}, \boldsymbol{C}, \alpha >$$

The details of all the elements except α are mentioned in Section 6.1.1. α denotes the tunable input parameter, that controls the amount of risk the EMS operators are ready to tolerate. Our goal is to find a subset of $\mathcal{B}$ and to optimally allocate an entire fleet of ambulances to these bases that minimises the α-response time, i.e., the threshold time within which α percent of the total requests should be served.

6.2 Related Work on Decision Making in EMS

Given the practical importance of EMSs, a wide range of disciplines have also studied problems associated with EMSs. The research papers in Section 6.2.1 focus on improving operational strategies for EMS. Section 6.2.2 elaborates the research in the direction of strategic decision making for large-scale and catastrophic disaster response. Finally, in Section 6.2.3 we present the relevant literatures for the optimisation of monotone submodular functions, which is complementary to the work presented in Chapter 7.

6.2.1 Operational decision making in EMS

This thread of research focuses on generating efficient ambulance allocation and dispatch policy for day-to-day operations. Andersson and Värbrand (2007) and Schmid (2012) develop techniques to optimally generate dispatching policy for ambulances and also provide a relocation model that dynamically suggests a destination base for ambulances after job completion. However due to inherent complexity of the process (such as congestion of ambulances at certain bases or problem with conceiving the critically of request by the operator), many EMSs prefer a fixed allocation of ambulances and follow the nearest ambulance dispatch policy.

Brotcorne, Laporte, and Semet (2003) and Gendreau, Laporte, and Semet (2006) exploit mathematical models by incorporating performance metrics as a parameter of the model and provide optimisation or local search based heuristics to solve the ambulance allocation problem. But optimisation models often fail to capture the dynamics of EMS such as congestion pattern in road or response time from base to scene that varies over time. Recent works (Saisubramanian et al., 2015; Yue et al., 2012; Restrepo, Henderson, & Topaloglu, 2009) overcome these caveats by employing a real-life event-driven simulator to evaluate the resulting policy. All the papers in this thread presume a fixed set of bases, while our approach presented in Chapter 7 solves the ambulance allocation problem in conjunction with discovering optimal placement for bases.

Operational decision making in EMS using MDP: Markov Decision Process (MDP) is a fundamental choice for sequential decision making under uncertainty. However, if a decision making problem of EMS is represented as an MDP, the state and action space of the corresponding MDP would be exponentially large[1] (similar to BSS as indicated in Section 2.2.6). Due to this challenge, only a few re-

[1]For our real-world data set, a problem instance contains 58 ambulances, 58 bases and around 3000 requests, so there would be approximately $3000^{58} \times 58^{58}$ (all the possible combination of assignment and allocation scenarios) states and 58^{58} (all the possible allocation of ambulances to bases) actions for representing ambulance allocation problem as an MDP.

search papers have represented the ambulance allocation and dispatching problem using MDP. Maxwell, Henderson, and Topaloglu (2009); and Maxwell, Henderson, Topaloglu, et al. (2013) employ MDP based approximate dynamic programming for dynamic relocation of ambulances. To address the inherent complexity, they solved the problem for individual ambulances separately and therefore, these models do not capture the exact dependencies and dynamics of EMS properly. Recently, Jagtenberg, Bhulai, and van der Mei (2017) employ MDP for computing the optimal dispatch strategy for ambulances for a given set of requests. All these papers propose solutions for the response phase by considering a fixed set of waiting emergency requests. However, as the EMS operators follow critical response time metrics, we need to dispatch ambulances on a first-come first-serve basis as soon as a request is arrived in the system and therefore, keeping requests in a waiting queue is often not feasible. So, our solution approaches consider a large number of historical emergency requests and generate ambulance allocation for preparedness so as to ensure the critical performance metrics of EMS for future scenarios. Furthermore, as the state and action space of MDP grow exponentially with the number of emergency requests, ambulances and bases, we directly employ optimisation models for solving the large-scale real-world EMS problems.

6.2.2 Strategic decision making in Disaster Response

This thread of research focuses on strategic decision making for rare and large-scale disaster response (ex: fire, vehicle accident or natural disaster). The traditional model for facility location in large-scale disaster response is based on the covering problem such as location set covering problem [LSCP] (Toregas et al., 1971), that aims to provide coverage to all the demand points; and maximal covering location problem [MCLP] (Church & Velle, 1974), that maximises the coverage for a given a budget. P-median (Hakimi, 1964) (minimises average distance between demand point and nearest facility) and P-center (Sylvester, 1857) (minimises the worse case

response time) models are also widely adopted in literature. Recently, Jia et al. (2007) and Huang et al. (2010) propose mathematical model for large-scale disaster response and solve it using optimisation method or dynamic programming. Due to the rare occurrence of the catastrophic events, these papers are focused on robust objectives that plan for the absolute worst case. In contrast, incidents in EMSs happen every day and objectives consider softer notions of robust decision making (e.g., maximise number of requests served within 15 minutes, minimise time taken to serve 80% of requests). Therefore, we take a data-driven approach to find the minimal set of bases in EMS and evaluate the performance of solution on a diverse set of demand scenarios.

6.2.3 Application of Greedy Algorithm in Optimising Submodular Functions

The last thread of research which is complimentary to the work presented in Chapter 7 is on optimisation of monotone submodular functions (Leskovec, Krause, Guestrin, Faloutsos, VanBriesen, & Glance, 2007; Nemhauser, Wolsey, & Fisher, 1978). Some popular application domains where greedy algorithms are successfully applied for optimising submodular functions are: dynamic conservation planning (Golovin, Krause, Gardner, Converse, & Morey, 2011), maximising information gain in sensor placement (Krause, Singh, & Guestrin, 2008), and content recommendation (Yue & Guestrin, 2011). The key reason behind this extensive adoption is that a greedy approach provides $(1-\frac{1}{e})$ approximation guarantee in case of monotone submodular functions.

6.3 Details of Referred Solution Approaches

In this section, we present some of the existing approaches for solving the ambulance allocation problem for a given set of bases in EMS. We have compared our

approaches with these existing benchmarks (for which we give the advantage to set the bases in all the possible locations) in Chapter 7.

6.3.1 Optimising Bounded Time Response

Given a sample of training requests, our goal with this objective is to find an allocation policy for ambulances $\mathcal{A}$ into given set of bases such that maximum number of requests can be served within a threshold time bound. For this objective, the optimisation model for finding an optimal allocation of ambulances to a given set of bases $\mathcal{B}$ is compactly represented using a Mixed Integer Linear Program [MILP] in Table (6.2); a simple extension of the MILP provided in (Yue et al., 2012). A request $r \in \mathcal{R}$ can be served from a feasible set of nearby bases $\{\mathcal{B}_r \cup \perp\}$, where $\perp$ denotes the null assignment or lost request. x_{rl} is a binary decision variable and is set to 1 if request r is served from base $l \in \{\mathcal{B}_r \cup \perp\}$. a_l denotes the number of ambulances allocated to base $l \in \mathcal{B}$.

Intuitively, one unit of reward is provided if a request is served within 15 minutes. Let L be a function that facilitates this reward and is defined as follows:

$$L_{rl} = \begin{cases} 1 & \text{if } T_{l,r.s} \leq 15 \text{ minutes} \\ 0 & \text{Otherwise} \end{cases}$$

Our objective (delineated in equation (6.1)) is to maximise the number of requests that are assisted within 15 minutes. Constraints (6.2) ensure that a request can be served from one base station only. P_r^l denotes the set of parents of request r for base l. A request $j \in P_l^r$ is considered as the parent of request r if it arrives before r, completes after r has arrived and base l belongs to both the feasible base set $\mathcal{B}_r$ and $\mathcal{B}_j$. Therefore, constraints (6.3) enforce the condition that a request can only be served from a base station if there is an available ambulance. Constraints (6.4) ensure the equivalence between total number of allocated and available ambulances. Finally, constraints (6.5) enforce that the number of allocated ambulances in a base

$$\max_{\boldsymbol{a},\boldsymbol{x}} \sum_{r\in\mathcal{R}}\sum_{l\in\mathcal{B}_r} x_{rl}L_{rl} \tag{6.1}$$

$$\textbf{s.t.} \sum_{l\in\{\mathcal{B}_r\cup\perp\}} x_{rl} = 1, \qquad \forall r\in\mathcal{R} \tag{6.2}$$

$$x_{rl} + \sum_{j\in P_r^l} x_{jl} \leq a_l, \qquad \forall r\in\mathcal{R}, l\in\mathcal{B}_r \tag{6.3}$$

$$\sum_{l\in\mathcal{B}} a_l = |\mathcal{A}| \tag{6.4}$$

$$0 \leq a_l \leq C_l, x_{rl} \in \{0,1\} \tag{6.5}$$

Table 6.2: **FINDALLOCATION**$(\mathcal{R},\mathcal{B},\mathcal{A})$

station is bounded by the capacity of that base.

6.3.2 Optimising Bounded Risk Response

The notion of bounded risk, which was introduced by Saisubramanian et al. (2015), is an important and alternative performance metric which is employed by many real-world EMSs. The optimisation model for calculating the utility for a given set of bases is compactly represented using the MILP in Table (6.3) and is a more efficient variant of the one provided in Saisubramanian et al. (2015). δ^r denotes the response time for request $r \in \mathcal{R}$. δ denotes the α-response time or alternatively the percentage of requests whose response time is greater than δ should be less than the input parameter α. z^r is a binary variable that is set to 1 if response time for request r is greater than δ.

Our goal is to find an allocation of ambulances to a given set of bases, $\mathcal{B}$ such that α-response time is minimised. M represents a sufficiently large number such that objective value is always positive. We set the objective function (delineated in equation (6.6)) positive such that it is consistent with the objective of MILP of Table (6.2). Constraints (6.7) ensure that z^r is set to 1 if response time for request r exceeds δ. Constraints (6.8) enforce that the percentage of requests whose response time exceeding δ is less than the input parameter α. Another key differentiating

$$\max_{\boldsymbol{a},\boldsymbol{x}} \; M - \delta \tag{6.6}$$

$$\textbf{s.t.} \quad \frac{\delta^r - \delta}{M} \leq z^r, \qquad \forall r \in \mathcal{R} \tag{6.7}$$

$$\frac{\sum_{r\in\mathcal{R}} z^r}{|\mathcal{R}|} \leq \alpha \tag{6.8}$$

$$\sum_{l\in\{\mathcal{B}_r\cup\perp\}} x_{rl} = 1, \qquad \forall r \in \mathcal{R} \tag{6.9}$$

$$x_{rl} + \sum_{j\in P_r^l} x_{jl} \leq a_l, \qquad \forall r \in \mathcal{R}, l \in \mathcal{B}_r \tag{6.10}$$

$$\sum_{l\in\mathcal{B}} a_l = |\mathcal{A}| \tag{6.11}$$

$$\delta^r \geq \sum_{l\in\mathcal{B}_r} x_{rl} \cdot T_{l,r.s} + x_{r\perp} \cdot \hat{M}, \qquad \forall r \in \mathcal{R} \tag{6.12}$$

$$0 \leq a_l \leq C_l, x_{rl} \in \{0,1\}, z^r \in \{0,1\}, \delta, \delta^r \geq 0 \tag{6.13}$$

Table 6.3: **RISKALLOCATION**$(\mathcal{R}, \mathcal{B}, \mathcal{A}, \alpha)$

constraints that have not been used in Section 6.3.1 are constrains (6.12). These constraints ensure that the response time for request r is equal to the travel time from base (dispatched ambulance location) to scene or a relatively high number $\hat{M}$ for null assignment.

Chapter 7

Strategic Decision Making in EMS

Emergency Medical Systems (EMSs) are an integral part of public health-care services. A typical EMS employs a set of Emergency Response Vehicles, ERVs (ex: ambulances, fire rescue vehicles) that provide timely care to patients (with injuries or illnesses) who seek immediate attention. In an EMS, a set of base stations are strategically placed throughout the city and a fixed number of ERVs are allocated to each base. On arrival of an emergency request, an ambulance from the nearest base is dispatched to assist the victim. The ambulance returns back to the same base after transferring the patient to a nearby hospital.

As the spatial distribution of resources in EMS (e.g., location of base stations and ambulances) is controlled strategically by the operators, strategic decisions in EMS is crucial and has major impact in EMS. Although facility location problems in large-scale disaster response systems for rare and catastrophic events (ex: earthquake and hurricane) enjoy a rich history (Toregas et al., 1971; Church & Velle, 1974; Jia et al., 2007; Huang et al., 2010), progress remains slow for strategic decision making in EMSs. Unlike decision making in large-scale disaster response systems for rare and catastrophic events, the incidents in EMS happen everyday and the patterns of how incidents happen change over time. Therefore, in this chapter, we are focussed on strategic decision making for EMSs (Ghosh & Varakantham, 2016).

Specifically, we are interested in the problem of setting up new bases (how many and where?). It is an extension of k-center facility location problem which is a well known NP-Hard problem (Hochbaum & Shmoys, 1985). Given the exponentially large space of possibilities (subsets of potential base stations that can be built in a given budget) and the direct dependence of the selected base set on optimal allocation of ambulances to bases, this is a computationally challenging problem. Furthermore, the budget for resources (ex: expense for setting up new bases or funds for new ambulances) is dynamic and arrives over time in different chunks and thus makes it difficult to plan all base locations well in advance.

Towards addressing the above mentioned challenges, our key contributions are as follows:

- We provide an incremental greedy algorithm where bases are added as long as the marginal gain is significant. We also show that for one of the objectives typically employed in EMS, the optimisation function is monotone submodular, there by guaranteeing at least 63% of optimal performance.

- We present an accelerated version of the greedy algorithm, referred to as lazy greedy and show that it can be utilised to optimise widely used performance objectives, namely bounded time response and bounded risk response.

- We employ a real-life event driven simulator to evaluate the performance of our approaches in comparison with existing benchmark approaches.

- Extensive empirical results on real-world data set from a large asian city demonstrate that our techniques (that utilise a significantly smaller number of bases) either outperform or provide highly competitive results in comparison with the best known approaches from literature.

7.1 Theoretical Analysis of Objectives

We consider two main objectives that are widely adopted by the EMS operators:
(a) Maximise number of requests that are satisfied within a given threshold response time (ex: 15 minutes), referred to as *Bounded Time Response*;
(b) Minimise the response time for a fixed percentage (ex: 80%) of requests, referred to as *Bounded Risk Response*.

In this section, we show that bounded time response objective is monotone submodular and bounded risk response objective is not submodular. Let $\mathcal{B}$ denotes a set of bases and $F(A)$ denotes the objective function for a given subset of bases $A \in 2^{\mathcal{B}}$, where objective function, $F : 2^{\mathcal{B}} \rightarrow \mathbb{R}$ is defined for a given set of requests $\mathcal{R}$, a fleet of ambulances $\mathcal{A}$ and a set of bases A.

Let A and B be two set of bases where $A \subset B \subseteq \mathcal{B}$. Let $\Delta(A|b)$ denotes the marginal gain in function F for adding a new base $b \in \mathcal{B} \setminus B$ to the current set of bases A. So, $\Delta(A|b) = F(A \cup \{b\}) - F(A)$. The objective function F is submodular if the marginal gain for adding a new base b in subset A is always higher than the gain for adding b in superset B, i.e.,

$$\Delta(A|b) - \Delta(B|b) \geq 0$$

.

Proposition 3. *F function is monotone submodular for bounded time response objective.*

Proof Sketch. Let $S_i \subseteq \mathcal{R}$ denotes the set of requests that can be served within 15 minutes from base i, then bounded time response function $F(A)$ for a given set of bases A and for optimal allocation of ambulances to A (analogous to the objective of MILP of Table (6.2)) is equivalent to $|\cup_{i \in A} S_i|$.

Let us have two sets of bases A and B, where B is the superset of A and represented as $\{A \cup a\}$, then Figure (7.1) shows the graphical proof of submodularity

of bounded time response function F by employing simple properties of set union. Formally,

$$\begin{aligned}\Delta(A|b) - \Delta(B|b) &= F(A \cup \{b\}) - F(A) - F(A \cup a \cup \{b\}) + F(A \cup a) \\ &= F(a \cap \{b\}) - F(A \cap a \cap \{b\}) \geq 0\end{aligned}$$

Hence, the bounded time response function F for a given set of requests $\mathcal{R}$ is submodular. ■

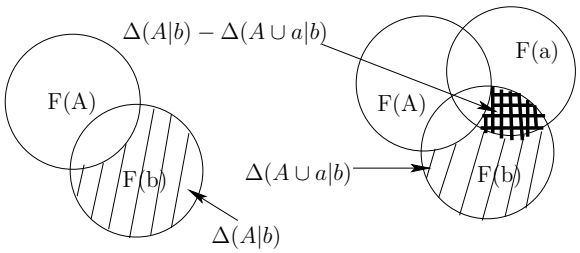


Figure 7.1: Bounded time response objective is Submodular

We now show that the bounded risk response objective is monotone but non-submodular. In Figure (7.2), we provide a simple counter example to show the non-submodularity of risk-based objective. For the ease of understanding we consider 5 requests each of which is represented by a circle. We have 3 bases represented as square. We consider a fleet of 5 ambulances. Numbers associated with each line denote the response time from the base to scene location. Let A denotes the subset and $A \cup \{a\}$ represents the superset. We are interested to find the marginal gain in α-response time for adding a new base b in both the cases. Let the tuneable parameter α is given as 0.2, therefore 80% (or 4) requests have to be served within δ. We assume the value of M as 100.

Using only base A, we can serve 4 requests within 15 minutes, so, the value of δ is 15 and our objective, $F(A)$ is 85. If we add the new base b to A, then we observe the following optimal assignment; request 1, 2 and 5 are served from base b and request 3, 4 are served from base A. The above assignments indicate that 4 requests are served within 7 minutes, so, $F(A \cup \{b\})$ is 93. The marginal gain denoted by

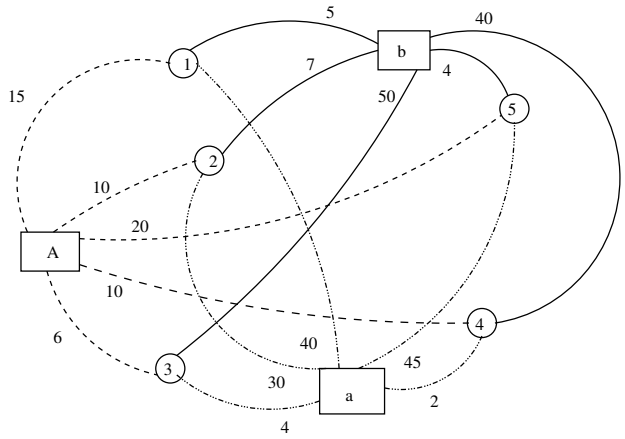


Figure 7.2: Non-submodularity of risk-based objective

$\Delta(A|b)$ is (93-85)=8. In case of superset $(A \cup \{a\})$, request 1, 2 and 5 are served from base A and request 3, 4 are served from base a. So, 4 requests are assisted within 15 minutes and $F(A \cup \{a\})$ is 85. If we add the base b in superset, then request 1, 2 and 5 are served from base b while request 3, 4 are served from base a. In this case, 4 requests are served within 5 minutes, thus, $F(A \cup \{a\} \cup \{b\})$ is 95. The marginal gain $\Delta(A \cup \{a\}|b)$ is 10. Therefore, $\Delta(A|b) < \Delta(A \cup \{a\}|b)$, which proves that the bounded risk response objective is not submodular.

7.2 Strategic Decision Making using Greedy Approach

In this section, we outline our approach for strategic decision making to decide on the number and the exact set of bases to be used. The generic model for this problem is described in Section 6.1. We provide a generic incremental approach that can be applied to both the objectives. We employ the well known greedy algorithm that guarantees to provide 63% of optimal objective (Nemhauser et al., 1978) for monotone submodular functions. Algorithm (4) provides the details of the greedy algorithm. We start with a null base set E. In each iteration we calculate the utility μ_s and optimal allocation of ambulances, $\boldsymbol{A}$ for adding each of the possible bases $s \in \mathcal{B}$ to active base set E. Based on the objective function we need to optimise, we employ the optimisation formulation of Table (6.2) or Table (6.3) from Section 6.3,

to optimally allocate the ambulances to the active base set E. Then we add the base s^* (that provides maximum marginal gain) into E and remove it from possible base set $\mathcal{B}$. The process continues until the marginal gain for adding a new base is significantly higher.

Algorithm 4: SolveGreedy($\mathcal{R}, \mathcal{B}, \mathcal{A}$)

Initialize: $E \leftarrow \{\perp\}, it \leftarrow 0$;
repeat
  $\mu_s, \boldsymbol{A} \leftarrow$ FindAllocation$(\mathcal{R}, E \cup \{s\}, \mathcal{A}), \forall s \in \mathcal{B}$;
  $s^* \leftarrow \underset{s \in \mathcal{B}}{\operatorname{argmax}}\, \mu_s$;
  $E \leftarrow E \cup \{s^*\}$;
  $\mathcal{B} \leftarrow \mathcal{B} - \{s^*\}$;
until $(\max_{s \in \mathcal{B}} \mu_s \leq \epsilon)$;
return $E, \boldsymbol{A}$

Minor modification to the greedy approach can easily tackle the real-life deployment issues such as political influences that are bound to occur in the decision making of EMS. Because of the political influences, a subset of bases might already be determined before the decision making process. In that scenario, we need to initialise the active base set E with the pre-determined set of bases rather than an empty set and incrementally add the best possible bases until the given budget constraint is satisfied.

7.2.1 Lazy Greedy Algorithm

We begin the discussion with the lazy greedy approach for optimising the bounded time response. Evaluating F function or **FindAllocation()** (which requires solving MILP of Table (6.2) from Section 6.3.1) is typically expensive even with a subset of bases and thus applying greedy algorithm (which requires evaluation of F function for every bases) can be computationally very expensive. Therefore, we employ a variant of greedy algorithm called lazy greedy (Minoux, 1978) to accelerate the convergence.

The details of lazy greedy process is shown in Algorithm (5). Let $\mathcal{B}$ be the set of available base stations and E be the current set of active bases. We initialise E with a default base $\{\perp\}$ where assignment of a request to $\perp$ indicates a null assignment. In the first iteration we calculate the gain μ_s for every possible bases $s \in \mathcal{B}$ (analogous to the greedy approach). We insert the base s^* with maximum marginal gain into E and remove s^* from available base set $\mathcal{B}$. In the subsequent iterations instead of computing gain $\Delta(E|s)$ for every base $s \in \mathcal{B}$ (which requires $O(|\mathcal{B}|)$ computations of function F), the lazy greedy keeps an upper bound μ_s for every available base. In each iteration it extracts the base ($s \in \operatorname{argmax}_{s' \in \mathcal{B}} \mu_{s'}$) with highest upper bound. Then it computes the marginal gain, $\Delta(E|s)$ for adding base s to existing base set E (i.e., the difference between the utilities $F(E \cup \{s\})$ = g^{it} and $F(E)$ = g^{it-1}) and update the upper bound μ_s as $\Delta(E|s)$. After this update if $\mu_s \geq \mu_{s'}$ for all $s' \in \mathcal{B}$, then greedy finds the best element with maximum gain (without computing gain for a large number of elements s') and insert base s into resulting base set E. This process iterates until there are no available bases whose marginal gain is higher than a predefined threshold value ϵ.

Proposition 4. *(Leskovec et al., 2007) For a placement of bases $E \in \mathcal{B}$ with a given fleet of ambulances $\mathcal{A}$, request log $\mathcal{R}$, and for each base $s \in \{\mathcal{B} \setminus E\}$ let $\Delta_s = F(E \cup s) - F(E)$. Then*

$$\max_{\mathcal{B},\mathcal{A},\mathcal{R}} F(\mathcal{B}) \leq F(E) + \sum_{s \in \{\mathcal{B} \setminus E\}} \Delta_s$$

By using Proposition (4) we can compute how far any given solution $F(E)$ is from the optimal solution, which can also be utilised for determining convergence.

We apply a similar lazy greedy approach to solve the bounded risk response objective, except that we calculate the F function using MILP of table (6.3) from Section 6.3.2. Even without the submodularity property of bounded risk response objective, we empirically show that lazy greedy is highly competitive with existing benchmark approaches and provide a good quality solution by utilising a signifi-

Algorithm 5: SolveLazyGreedy($\mathcal{R}, \mathcal{B}, \mathcal{A}$)

Initialize: $E \leftarrow \{\perp\}, it \leftarrow 0$;
$\mu_s, \boldsymbol{A} \leftarrow$ FindAllocation$(\mathcal{R}, E \cup \{s\}, \mathcal{A}), \quad \forall s \in \mathcal{B}$;
$g^0 \leftarrow \max_{s \in \mathcal{B}} \mu_s$;
$s^* \leftarrow \operatorname{argmax}_{s \in \mathcal{B}} \mu_s$;
$E \leftarrow E \cup \{s^*\}$;
$\mathcal{B} \leftarrow \mathcal{B} - \{s^*\}$;
repeat
  $it \leftarrow it + 1$;
  repeat
    $s^* \leftarrow \operatorname{argmax}_{s \in \mathcal{B}} \mu_s$;
    $g^{it}, \boldsymbol{A} \leftarrow$ FindAllocation$(\mathcal{R}, E \cup \{s^*\}, \mathcal{A})$;
    $\mu_{s^*} \leftarrow g^{it} - g^{it-1}$;
    if $\{\mu_{s^*} \geq \mu_s, \forall s \in \mathcal{B}\}$ **then**
      $E \leftarrow E \cup \{s^*\}$;
      $\mathcal{B} \leftarrow \mathcal{B} - \{s^*\}$;
      Break;
  until ***True***;
until $(\max_{s \in \mathcal{B}} \mu_s \leq \epsilon)$;
return $E, \boldsymbol{A}$

cantly less number of bases.

7.3 Experimental Settings

We conduct experiments on a real-world data set[1] from a large asian city (adopted from Yue et al., 2012). The data set contains a fleet of 58 ambulances and 58 base stations. We have 1500 weeks of request logs which are generated using *Poisson distribution* (Ross, 1983) with the parameters estimated from real usage data over a period of one month. Each request log contains the following information (a) Origin location; (b) Arrival time; (c) A set of feasible nearby bases from where the request can be assisted; (d) Response time from each of the feasible base to scene location; and (e) Total time required for an ambulance to return back to the origin base after

[1] http://projects.yisongyue.com/ambulance_allocation/

serving the request. In case of real deployment, the above mentioned details may not be readily available for new base locations, however, it is possible to estimate them using a straightforward method. We know the geographical locations of the requests and hospitals from the historical data. The geographical locations of the set of possible bases are also provided by the respective authority. Therefore, we can find the set of feasible nearby bases for each request and estimate the expected response and round off time for each of the possible nearby bases.

We evaluate the performance of our policy by employing a real-life event-driven simulation model (Yue et al., 2012) based on the nearest ambulance dispatch policy. We use Sample Average Approximation [**SAA**] (Verweij, Ahmed, Kleywegt, Nemhauser, & Shapiro, 2003) for validation and performance estimation. Specifically, we generate 10 policies using a training data set consisting of request logs for 10 weeks. Then we identify the policy with best validation performance over 500 weeks of request logs. Finally, we evaluate the performance of the validated policy on 3 test data sets each of which contains 300 weeks of request logs. We compare our approach with three existing benchmark approaches from literature (a) Greedy approach provided by (Yue et al., 2012); (b) Risk-based optimisation approach [RBO] (Saisubramanian et al., 2015); and (c) A baseline approach where 1 ambulance is allocated to every base.

7.3.1 Simulation Model

We evaluate the performance of ambulance allocation policy on the resulting base set using a real-life event-driven simulation model (courtesy: (Yue et al., 2012)) based on the nearest ambulance dispatch policy. The pseudo code for the event-driven simulator is shown in Algorithm (6). We start with an event set ξ where each element $e \in \xi$ represents a request and the list is sorted based on arrival order of requests. I denotes the set of available ambulances that are allocated according to given policy $\boldsymbol{A}$. a_r denotes the ambulance id that is assigned for request $r \in \mathcal{R}$.

Initially each request is tagged as null assignment. In each iteration we pop the first element e from the event list ξ. If the event e is a new request then we dispatch the nearest available ambulance a_r for the request and remove the ambulance from available ambulance set I. We also insert a job-completion event in the event list at time $t_r(a_r)$, where $t_r(a_r)$ denotes the time when ambulance a_r will return back to base after completing the job r . On the other hand, if the popped element e is a job completion event for request r, then we add the ambulance a_r to the set I such that it can be used to serve a new request. This process continues until the event list becomes empty. Once the process is finished, we can use the assignment results to measure the responsiveness of the system such as bounded time response or bounded risk response time for the given sample requests. We use this simulation model to compute the performance metrics for all the benchmark algorithms.

Algorithm 6: EDSimulator($\mathcal{R}, \mathcal{B}, \boldsymbol{A}$)

Initialize: $it \leftarrow 0$;
$I \leftarrow \boldsymbol{A}$ // Initialise set of available ambulance;
$\xi \leftarrow \mathcal{R}$ sorted in arrival order;
$\boldsymbol{a} = \{a_r | a_r \leftarrow \perp\}$ //Initialise as null assignment ;
repeat
  Pop next arriving event e from ξ;
  if e =*New Request* r **then**
    $a_r \leftarrow Dispatch(r, I)$ // Dispatch nearest free ambulance;
    $I \leftarrow I - \{a_r\}$ // Update available ambulance;
    Push job completion event at time $t_r(a_r)$ into ξ;
  else if *e=job completion event for* r **then**
    $I \leftarrow I \cup \{a_r\}$ // Update available ambulance;
until $(|\xi| > 0)$;
return $\{\boldsymbol{a_r}\}$

7.3.2 Sample Average Approximation (SAA)

We employ Sample Average Approximation (Verweij et al., 2003) for policy validation and performance estimation. We generate M minimal base sets $B_1, ..., B_M$ and allocation policies $A_1, ..., A_M$ for M sample of request logs. Then we validate

those policies on N_{valid} samples and select the best allocation policy A^* and base placement B^*, which has maximum validation performance. Finally we test the performance of policy (A^*, B^*) on a separate collection of N_{test} samples and report the performance statistics. We measure the performance metrics by taking average over all the samples. For e.g., if we have N sample of request logs $\mathcal{R} = \{R_1, ..., R_N\}$, then the expectation is computed using Equation (7.1) by taking average over all the N samples.

$$F_{\mathcal{R}}(A^*, B^*) = \frac{1}{N} \sum_{i=1}^{N} \sum_{r \in R_i} F_r(A^*, B^*) \tag{7.1}$$

Benchmark 1: (Yue et al., 2012) The primary goal of this paper is to efficiently allocate an entire fleet of ambulances to a predetermined set of bases such that the percentage of requests served within a certain threshold time bound is maximised. They used a greedy approach to find the optimal allocation for ambulances in each iteration using a real-life event driven simulator and incrementally added the ambulances until the entire fleet is allocated efficiently.

Benchmark 2: (Saisubramanian et al., 2015) This paper proposes to minimise the bounded risk (i.e., the time bound within which $\alpha\%$ requests are served), a metric employed by many EMSs, by efficiently allocating a fleet of ambulances to a given set of bases. We provide the details of their approach in Section 6.3.2 and an equivalent optimisation formulation given in Table (6.3).

7.4 Experimental Results

We compare our approach with respect to performance metrics such as (a) Runtime; (b) Bounded time response: percentage of requests served within 15 minutes; and (c) Bounded risk response: α-response time (unless otherwise stated we use α value as 0.2). We provide five thread of results on real-world data set (a) Gain in runtime for lazy greedy over general greedy approach; (b) Experimental validation of the

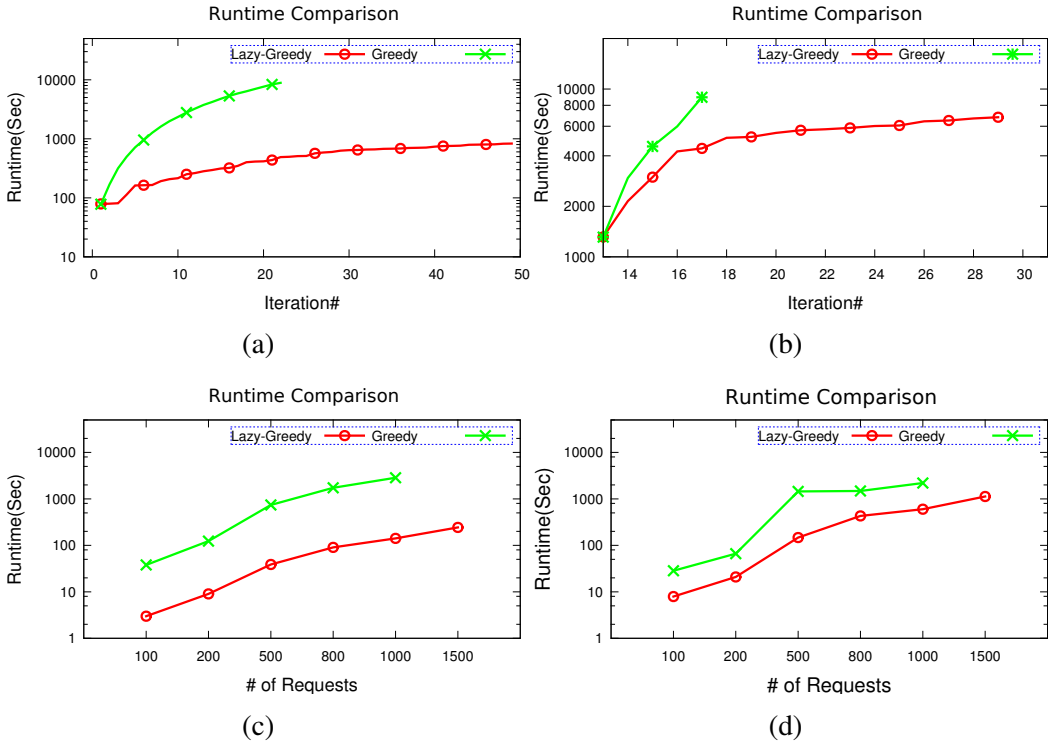


Figure 7.3: Runtime: Greedy vs. Lazy greedy (a) Iterations wise for Bounded time response; (b) Iterations wise for Bounded risk response; (c) With varying request for Bounded time response; (d) With varying request for Bounded risk response

submodularity of bounded time response and non-submodularity of bounded risk response; (c) Effect of external parameter such as risk tolerance level [α] on strategic decision making; (d) Effect of external budget such as size of ambulance fleet on two objective functions as well as on the strategic decision making (number of required bases); and (e) Performance comparison with the benchmark approaches on three test data sets, each contains 300 weeks of requests.

Runtime Results : Figure (7.3) plots the runtime comparison between lazy greedy and general greedy approach. Figure 7.3(a) depicts the runtime for bounded time response objective on a sample of around 3000 requests. X-axis denotes the iteration number and Y-axis represents the runtime in seconds in a logarithmic scale. Greedy approach is unable to finish more than 20 iterations within the cut-off time of 2 hours, while lazy greedy approach provides a significant gain over greedy and completes the process within 10 minutes. Figure 7.3(b) shows the runtime for bounded risk response objective. While greedy approach is unable to complete 18

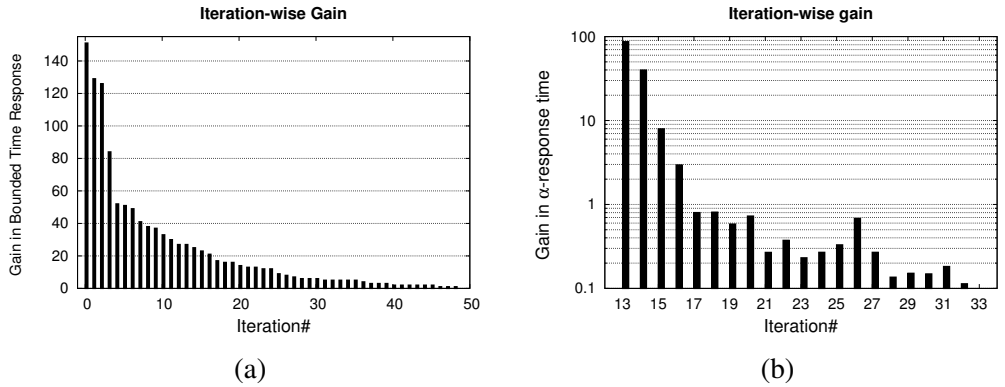
iterations within the threshold time of 2 hours, lazy greedy significantly accelerates it and finish the process within the cut-off time. Note that the runtime for both the greedy and lazy greedy for initial 12 iterations was equal. This is so because we cannot serve 80% of the requests (i.e., $\alpha = 0.2$) using less than 13 base stations (because a request can only be assisted from a subset of nearby bases), and therefore in the initial iterations upper bound was equal for every possible bases (i.e., $\mu_s = M, \forall s \in \mathcal{B}$). So, the lazy greedy essentially search over all the possible bases, which is equivalent to general greedy approach.

Figure 7.3(c) demonstrates the gain in runtime for lazy greedy approach where we vary the number of requests in the X-axis. The complexity of greedy approach grows exponentially as the number of requests increases. This is so because the dependency between requests increases for densely populated request logs. Greedy cannot solve problems with more than 1000 requests within the cut-off time, while lazy greedy solves the problem with 1500 requests within 2 minutes. In the same direction, Figure 7.3(d) demonstrates that lazy greedy significantly outperforms greedy approach in case of bounded risk response objective.

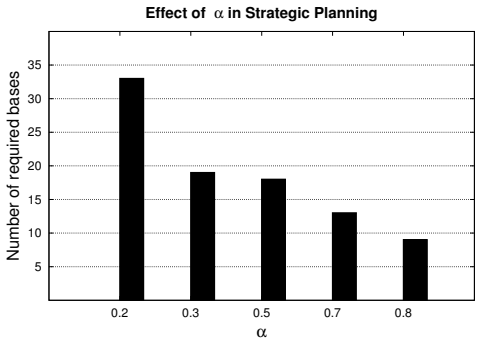
Submodularity Results : Figure 7.4(a),7.4(b) depict the marginal gain for adding a base in each iteration for both the objective functions. Figure 7.4(a) clearly shows that marginal gain decreases monotonically in each iteration which validates the submodularity property of the bounded time response objective. Figure 7.4(b) delineates the iteration wise gain of α-response time in a logarithmic scale. As expected, due to the non-submodularity, in few cases the marginal gain in later iteration is slightly higher.

Effect of external parameter α: Figure 7.4(c) depicts the effect of parameter α in strategic decision making for the bounded risk response objective on a fixed sample of requests. Note that increasing α value indicates that less number of requests need to be served within α-response time. Therefore, the size of resulting base set reduces as we increase the α value.

Results on varying budget : Our model can be employed to find the right loca-

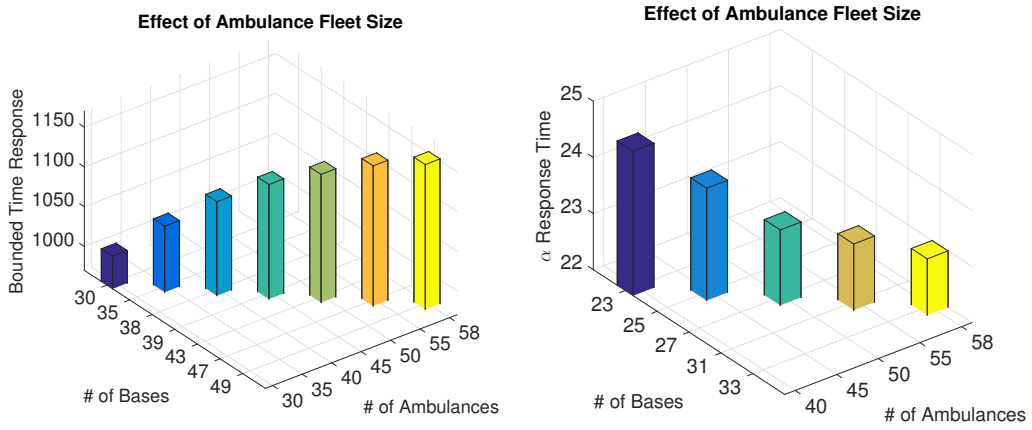


(a) (b)



(c)

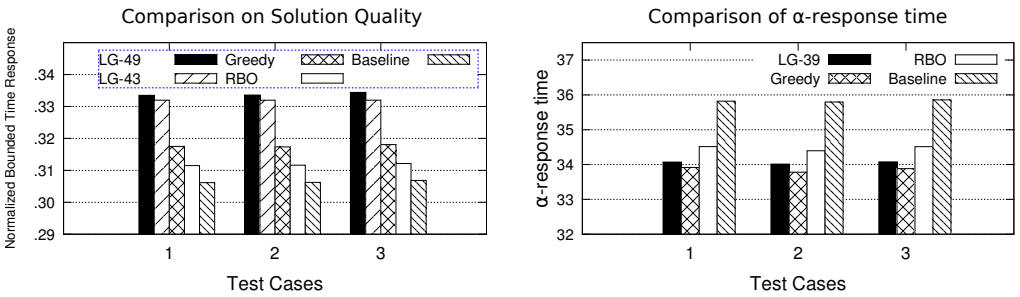
Figure 7.4: (a) Iteration-wise gain for bounded time response; (b) Iteration-wise gain for bounded risk response; (c) Effect of α on strategic decision making.



(a) (b)

Figure 7.5: Effect of fleet size for optimising (a) Bounded time response; (b) Bounded risk response.

tion for a small set of new ambulances in addition to an existing fleet of ambulances. For e.g., if a new budget arises for p ambulances at certain point of time, and q number of ambulances already exists in system, then we can use our algorithm with $(p+q)$ ambulances to find the minimal subset of bases such that the entire fleet can



(a) (b)

Figure 7.6: Quality : (a) Bounded time response; (b) Bounded risk response.

be allocated efficiently. Figure (7.5) show the performance with respect to varying fleet size on a sample of around 3000 requests. We show the effect of varying fleet size in the strategic decision making (ex: number of required bases) as well as in objective value (ex: bounded time response or α-response time). We vary the ambulance fleet size in X-axis, while Y-axis shows the size of active base set and Z-axis denotes the utility. We observe the pattern is consistent, i.e., bounded time response increases with number of ambulances (Figure 7.5(a)) and bounded risk response is inversely proportional to fleet size (Figure 7.5(b)). For both the objectives, as we increase the number of ambulances, we need additional bases to effectively allocate the entire fleet of ambulances.

Results on test cases : The last and most important thread of results demonstrate the performance comparison between all the benchmark approaches on the test instances. We provide performance for two of our allocation policies. LG-49 represents an allocation policy (generated using lazy greedy) where the process continues until the marginal gain is positive and it produces a resulting base set of size 49. LG-43 symbolises an allocation policy with 43 bases where we stop the process if the marginal gain is less than or equal to 2. It indicates a crucial advantage of our approach in strategic decision making as we have the flexibility to generate strategy based on the expectation of EMS operators and the availability of budget to construct the base stations. Figure 7.6(a) plots the normalised bounded time response value for all the test cases. Y-axis represents the percentage of requests served

within 15 minutes. As each of the test cases involves 300 weeks of request logs, we report the average utility using SAA. In all the test cases our allocation policy (even with lesser number of bases) outperforms the existing benchmark approaches and provide almost 2% gain in bounded time response.

Figure 7.6(b) illustrates the performance comparison on α-response time. LG-39 symbolises an allocation policy with 39 bases that is generated using lazy greedy. Interestingly by utilising less than 70% of total bases, our approach significantly outperforms the baseline approach and is highly competitive with other two benchmark approaches.

7.5 Summary

In this chapter, we present a promising approach for placement of bases and ambulances in EMS. We employ an incremental greedy approach that identifies the base with maximum marginal gain in each iteration and add it to the resulting base set. A lazy greedy approach is further utilised to accelerate the convergence and the derived policy is evaluated using a real-world event driven simulator. We show that our approach can be utilised to optimise crucial performance metrics such as bounded time response and bounded risk response. The empirical results on real-world data set demonstrate that our approach significantly improves the service level of EMS over existing benchmark approaches.

Chapter 8

Improving Operational Decisions of EMS

In order to sustain and maintain the efficiency of an EMS, it is important to improve the operational decisions either by optimising allocations of ambulances to base stations or by improving strategies to dispatch ambulances to incidents. Furthermore, we need to dynamically redistribute the ambulances on a regular basis to tune with the changes in the patterns of spatio-temporal distribution of emergency incidents. Therefore, in this chapter, we focus on improving the operational decisions of EMS. Specifically, our goal is to find the dynamic allocation (e.g., allocation changes on every weekdays) of an entire fleet of ambulances to a given set of bases (Ghosh & Varakantham, 2018).

As explained in Chapter 6, the recent papers (Yue et al., 2012; Saisubramanian et al., 2015) in improving the operational decisions of EMSs have utilised the data-driven models to optimise performance metrics such as bounded time response (percentage of requests served within a threshold time bound) or bounded risk response (response time within which a fixed percentage of requests are served). However, both the optimisation models of Table (6.2) and (6.3) present the real-world dynamics of EMS approximately. Specifically, the ambulance dispatch rules are avoided while generating ambulance allocation. Therefore, the optimiser has the flexibil-

ity to follow an omniscient dispatch rule to optimise the objective function, which "overfits" the solution according to training incidents. However, the use of omniscient dispatch is not realistic since the optimisation occurs over a different evolution of emergency response dynamics than the one that happens on the ground. Therefore, the real-world EMS operators generally dispatch the nearest available ambulance to efficiently respond to an emergency request.

The existing papers tackle this issue using the following two approaches: (a) Yue et al. (2012) proposed a greedy approach to incrementally allocate one ambulance at a time using an event-driven simulator that follows the nearest available ambulance dispatch strategy. They employ the bounded time objective where the goal is to maximise the number of incidents served within a fixed time. However, the solution of this greedy approach can be far away from the optimal due to its myopic nature (one request at a time) and as the bounded time response metric is not sub-modular; (b) On the other hand, Saisubramanian et al. (2015) generated an allocation that minimises response time with a bounded risk (i.e., percentage of incidents that can have response times higher than the objective) using a linear optimisation model that follows an omniscient dispatch policy. They then evaluate the obtained solution using a simulator with the nearest available ambulance dispatch strategy to get the actual objective value. We observe that there might be a significant gap between the objectives of the optimisation model and the simulation model, as they follow different dispatch strategies. Therefore, the solution of the optimisation model does not provide any quality guarantee.

In order to reduce the response times for emergency requests, we need to consider both these operational (day-to-day) inefficiencies simultaneously: (a) allocation of all ambulances (and not one by one) to base stations; and (b) dispatch of the "right" ambulances to the emergency requests. To tackle the above mentioned issues and to bridge the gap between the optimisation model and the real-world scenarios (i.e., the event-driven simulation provided in Algorithm 6), we provide a dispatch guided optimisation approach for allocating all ambulances to base stations. We

specifically consider the widely used bounded time objective employed by Yue *et al.* (Yue et al., 2012). To that end, our key contributions are as follows:

- We provide a novel Integer Linear Programming (ILP) model for dynamic allocation of ambulances that incorporates the real-world ambulance dispatch strategies as linear constraints. This allows for exactly imitating the real dynamics of EMS when optimising the allocation.

- As the proposed ILP and its equivalent constraint programming (CP) models suffer with scalability issues when the number of emergency requests are increased, we provide two novel heuristic approaches to solve the problems with large number of incidents.

- By employing an event-driven simulation model based on two real-world EMS data sets, we empirically show that our proposed heuristic approaches can consistently and in some cases significantly improve the efficacy of EMS over the existing benchmark approaches.

8.1 Dynamic Redistribution of Ambulances

Dynamic ambulance redistribution problem is an extension of the static version of the ambulance allocation problem in a given set of bases, which was previously explained in Section 6.1. The only difference is that we now need to generate separate allocation of ambulances for different weekdays or the allocation might change at different time of the day. For this problem, we consider the widely used performance metric of EMS called bounded time response (i.e., maximise the number of requests served within a given threshold time bound). This problem is referred to as Dynamic Ambulance Allocation Problem (**DAAP**). We employ the following tuple to represent the DAAP: $< \mathcal{B}, \mathcal{A}, \mathcal{R}, \boldsymbol{T}, \boldsymbol{C}, L >$

$\mathcal{B}$ denotes the set of base locations and $\mathcal{A}$ represents a fleet of ambulances. $\mathcal{R}$ denotes a set of emergency requests for a particular weekday (e.g., incident logs of

consecutive ten Mondays). Each request $r \in \mathcal{R}$ is a tuple $\langle t, s, h, \mathcal{B}_r, \mu_r, \lambda_r \rangle$. t, s, h denote the arrival time, source location and destination hospital for the particular request r. $\mathcal{B}_r$ represents a set of nearby base stations from which the request r can be served. μ_r provides the response times for each of the nearby bases in $\mathcal{B}_r$. Specifically, if $\mathcal{B}_r = \{l_1, l_2, ...\}$, then μ_r^i denotes the response time from base l_i. For the ease of representation, we assume that the nearby base set $\mathcal{B}_r$ is sorted according to the response times. That is to say, $\mu_r^i \leq \mu_r^{i+1}$. λ_r provides the round-about times (i.e., the total time required for an ambulance to return back to the origin base after serving the request) for the nearby bases, where λ_r^i denotes the round-about time for base l_i. $\boldsymbol{T}$ provides travel time between any two base locations and $\boldsymbol{C}$ denotes the capacities of the bases. Finally, L represent the utility function, which is defined as follows:

$$L_{rl} = \begin{cases} 1 & \text{if } T_{l,r.s} \leq \Delta \\ 0 & \text{Otherwise} \end{cases}$$

Where, Δ denotes the threshold response time bound provided by the EMS operators. Intuitively, a reward of 1 unit is provided if a request is served within the threshold time. With the given *DAAP* input tuple, our objective is to find an efficient dynamic[1] allocation of an entire fleet of ambulances, $\mathcal{A}$ to a given set of base stations, $\mathcal{B}$ that maximises the percentage of requests which can be served within the given threshold time bound, Δ. This is also referred to as the bounded time objective provided by Yue *et al.* (Yue et al., 2012).

We first propose an exact Integer Linear Programming (ILP) formulation for efficiently solving the *DAAP*. This exact formulation can also with minor modifications be converted to a Constraint Program (CP). However, as the two exact models do not scale to problems with large number of requests, we provide two novel heuristic approaches to improve scalability of our solution.

[1]For dynamic allocation, the allocation strategy changes on every weekday. For instance, to generate the allocation strategy for a Monday, we consider $\mathcal{R}$ as the set of requests of past Mondays.

8.1.1 Integer Linear Programming Formulation

We first provide a compact ILP formulation to find an optimal allocation for a fleet of ambulances, $\mathcal{A}$ to the given set of bases, $\mathcal{B}$ by extending the optimisation model of Table (6.2). A request $r \in \mathcal{R}$ can be served from a feasible set of nearby bases, $\{\mathcal{B}_r \cup \perp\}$, where $\perp$ denotes a null assignment (i.e., the request cannot be served). Let, x_{rl_i} denotes a binary assignment variable, which is set to 1 if the request r is served from base $l_i \in \{\mathcal{B}_r \cup \perp\}$. Let, a_l is an integer variable which denotes the number of ambulances allocated to base $l \in \mathcal{B}$. a_l can be set to any value between 0 and the base capacity C_l. Our objective in the ILP is to find an efficient allocation that maximises the utility function, L.

$$\max_{\boldsymbol{a},\boldsymbol{x}} \sum_{r \in \mathcal{R}} \sum_{l_i \in \mathcal{B}_r} x_{rl_i} L_{rl_i}$$

To represent the evolution dynamics of EMS exactly, we now describe the constraints. Please note that the description of dispatch constraints is novel and the title of dispatch constraint description is highlighted in bold below:

A request can only be assigned to one base station: This set of constraints ensure that only one ambulance from one of the feasible nearby bases is dispatched to assist an emergency incident. If all the nearby bases are empty when the request arrived into the system, the request is assigned to a dummy base $\perp$ and we label it as a null assignment.

$$\sum_{l_i \in \{\mathcal{B}_r \cup \perp\}} x_{rl_i} = 1, \qquad \forall r \in \mathcal{R}$$

A request can be served from a base if it has at least one ambulance available: Let $P_r^{l_i}$ denotes the set of parent requests for r that are served from base l_i. More specifically, a requests r' belongs to the parent set $P_r^{l_i}$, if it has arrived in the system before request r and if an ambulance is assigned from base l_i for request r', then the assigned ambulance is still busy in serving r' when the request r has arrived. Therefore, these set of constraints enforce that if all the ambulances of a base l_i are

busy in serving the parent requests of r (i.e., $\sum_{j \in P_r^{l_i}} x_{jl_i} = a_{l_i}$), then the request r cannot be served from base l_i.

$$x_{rl_i} + \sum_{j \in P_r^{l_i}} x_{jl_i} \leq a_{l_i}, \qquad \forall r \in \mathcal{R}, l_i \in \mathcal{B}_r$$

The entire fleet of ambulances has to be allocated: This constraint assures that each ambulance is allocated to one of the base stations.

$$\sum_{l \in \mathcal{B}} a_l = |\mathcal{A}|$$

The nearest available ambulance needs to be dispatched for assisting an emergency request: As mentioned previously, we assume that the set of nearby bases, $\mathcal{B}_r$ from which a request r can be served is sorted according their response times. So, the logical constraints (8.1) ensure that a request is always served from the nearest base with more than one idle ambulance. Precisely, constraints (8.1) enforce that a request r must be assisted from a base $l_i \in \mathcal{B}_r$ where more than one ambulance is present and all the other bases from which request r can be served faster are empty when the request has arrived.

$$\sum_{k \leq i} x_{rl_k} \geq 1 \qquad \textbf{if} \quad \underbrace{a_{l_i} - \sum_{j \in P_r^{l_i}} x_{jl_i}}_{\text{\#ambulance available at base } l_i} \geq 1 \tag{8.1}$$

To linearise these constraints we introduce a binary variable b_{rl_i} which is set to 1 if more than one ambulance is available in base $l_i \in \mathcal{B}_r$ when the request r has arrived.

$$b_{rl_i} = \begin{cases} 1 & \text{if } \sum_{j \in P_r^{l_i}} x_{jl_i} \leq a_{l_i} - 1 \\ 0 & \text{Otherwise} \end{cases}$$

The logical definition of these binary variables, $\boldsymbol{b}$ can easily be linearised using the following set of linear constraints, where C_{l_i} denotes the capacity of base l_i.

$$a_{l_i} - \sum_{j \in P_r^{l_i}} x_{jl_i} \leq C_{l_i} \cdot b_{rl_i} \qquad \forall r \in \mathcal{R}, l_i \in \mathcal{B}_r$$

$$a_{l_i} - \sum_{j \in P_r^{l_i}} x_{jl_i} \geq C_{l_i} \cdot (b_{rl_i} - 1) + 1 \forall r \in \mathcal{R}, l_i \in \mathcal{B}_r$$

We can now replace the logical and non-linear dispatch constraints (8.1) by using the following linear constraints.

$$\sum_{k \leq i} x_{rl_k} \geq b_{rl_i} \qquad \forall r \in \mathcal{R}, l_i \in \mathcal{B}_r$$

An efficient alternative for the dispatch constraints: Let, $|\bar{\mathcal{B}}|$ denotes the average number of nearby bases for each of the incidents. To represent the dispatch policy according to the above mentioned approach, we need to introduce $|\mathcal{R}| \times |\bar{\mathcal{B}}|$ binary variables and $3 \times |\mathcal{R}| \times |\bar{\mathcal{B}}|$ linear constraints. Due to these large number of newly introduced binary variables and constraints, the prior approach for incorporating the dispatch constraints performs poorly. Therefore, in this section we provide a simplified and compact representation of the dispatch constraints (8.1). According to constraints (8.1), we just need to ensure that a request is served from a base with an idle ambulance if other adjacent bases (from which the request can be served faster) are empty. As the assignment variables, $\boldsymbol{x}$ are binary, it would be adequate if we can ensure that the value of $\sum_{k \leq i} x_{rl_k}$ (i.e., sum of all the assignment variables for bases whose response times are less than or equal to the one for base l_i) is greater than zero and less than or equal to one. These conditions can be imposed using constraints (8.2), where C_{l_i} denotes the capacity of the base l_i (i.e., the maximum number of ambulances the base l_i can hold at a time). Specifically, we normalise the right-hand side value of constraints (8.2) to ensure that it is always bounded between 0 and 1. Note that, as C_{l_i} is a given input, the constraints (8.2) are linear in

nature.

$$\sum_{k \leq i} x_{rl_k} \geq \frac{1}{C_{l_i}} \Big[\underbrace{a_{l_i} - \sum_{j \in P_r^{l_i}} x_{jl_i}}_{\text{\#ambulance available at base } l_i} \Big] \; \forall r \in \mathcal{R}, l_i \in \mathcal{B}_r \tag{8.2}$$

We show the entire ILP model for the ambulance allocation problem compactly in Table (8.1). We refer to this approach as **ILP** in the later sections.

$$\max_{\boldsymbol{a},\boldsymbol{x}} \sum_{r \in \mathcal{R}} \sum_{l_i \in \mathcal{B}_r} x_{rl_i} L_{rl_i} \tag{8.3}$$

$$\textbf{s.t.} \sum_{l_i \in \{\mathcal{B}_r \cup \perp\}} x_{rl_i} = 1, \qquad \forall r \in \mathcal{R} \tag{8.4}$$

$$x_{rl_i} + \sum_{j \in P_r^{l_i}} x_{jl_i} \leq a_{l_i}, \qquad \forall r \in \mathcal{R}, l_i \in \mathcal{B}_r \tag{8.5}$$

$$\sum_{l \in \mathcal{B}} a_l = |\mathcal{A}| \tag{8.6}$$

$$\sum_{k \leq i} x_{rl_k} \geq \frac{1}{C_{l_i}} \big[a_{l_i} - \sum_{j \in P_r^{l_i}} x_{jl_i} \big] \qquad \forall r \in \mathcal{R}, l_i \in \mathcal{B}_r \tag{8.7}$$

$$a_l \in \{0, 1, ..., C_l\}, x_{rl_i} \in \{0, 1\} \tag{8.8}$$

Table 8.1: **FINDALLOCATIONWDISPATCH**$(\mathcal{R}, \mathcal{B}, \mathcal{A})$

8.1.2 Constraint Programming

As the optimisation model of Table (8.1) cannot be solved optimally with more than few hundred emergency requests using state-of-the-art black-box optimisation solvers such as *CPLEX*, we now provide an alternative constraint programming (CP) model of Table (8.1). For every allocation variable, a_l we have created variable $\boldsymbol{allocation[l]}$ whose domain range is defined as $\{0, 1, ..., C_l\}$. Similarly, for the assignment variables, x_{rl} we created variable $\boldsymbol{assignment[r][l]}$ whose domain range is defined as $\{0, 1\}$. With these definition of variables, the equations (8.3)-(8.7) of Table (8.1) can be translated to CP. We refer to this approach as **CP**.

8.1.3 Continuous Assignment

Unfortunately, neither the ILP nor the CP model can be solved optimally within our threshold time-limit of 12 hours. Therefore, we now provide an heuristic approach which can be solved within a minute with large number of emergency incidents. We essentially modified the MILP of Table (8.1) by relaxing the 0/1 assignment variables to a probabilistic or continuous assignment. The revised optimisation model is shown in Table (8.2), where we modified the assignments, $\boldsymbol{x}$ from discrete or binary to continuous variables. However, as the allocation variables, $\boldsymbol{a}$ remain integer, we are still allocating each ambulance to exactly one base station. Therefore, the solution of the optimisation problem will provide a valid ambulance allocation, which can be executed on the event-driven simulator delineated in Algorithm (6) to obtain a valid and integral assignment for each request and to compute the actual utility of the allocation strategy. Although the objectives of the optimisation problem and the simulation model might not be synchronised, we experimentally show that this approach provides reasonably better solution than the above mentioned exact approaches. This approach is referred to as **Relaxation** in the later sections.

$$\max_{\boldsymbol{a},\boldsymbol{x}} \sum_{r\in\mathcal{R}} \sum_{l_i\in\mathcal{B}_r} x_{rl_i} L_{rl_i}$$

s.t. Constraints (8.4), (8.5), (8.6) and (8.7) holds

$$a_l \in \{0, 1, ..., C_l\}, 0 \leq x_{rl_i} \leq 1 \tag{8.9}$$

Table 8.2: **FINDRELAXEDALLOCATION**$(\mathcal{R}, \mathcal{B}, \mathcal{A})$

Observation 4. *If all the base stations have single capacity (i.e.,* $a_l \in \{0, 1\}$*), then the optimisation model of Table* (8.2) *provides an optimal and integral solution.*

Proof: In case of single capacity base stations, the allocations $\boldsymbol{a}$ become binary variables. Therefore, when the first request r arrives in the system, constraints (8.7) enforce that the assignment variable x_{rl} is set to 1 if base l is the nearest base for

request r and $a_l = 1$. If the nearest base is empty, then this logic is applicable for the second nearest base and so on. Henceforth, no request can be served from base l, until the ambulance returns back to the base after serving the request. Due to this reasoning, the value of the right-hand side of constraints (8.7) can only be either 0 or 1. Hence, the assignment variables, $\boldsymbol{x}$ for all the requests can take either 0 or 1. Therefore, even with continuous assignment variables, $\boldsymbol{x}$ the optimisation model of Table (8.2) provides an integral solution and is equivalent to our ILP model of Table (8.1). ■

8.1.4 Two-stage Optimisation

In this section, we provide another heuristic approach to find an efficient ambulance allocation. We propose a two-stage hierarchical approach[2], where a preliminary allocation is generated for a subset of ambulances in the first stage and then we utilise that to guide the solution of the second stage for achieving allocation of the entire fleet of ambulances. In the first stage, we solve the MILP of Table (8.1) as a linear program (LP). That is to say, we relax both the allocation, $\boldsymbol{a}$ and assignment, $\boldsymbol{x}$ variables from integer to continuous one. The LP formulation for the first stage optimisation problem is shown in Table (8.3).

$$\max_{\boldsymbol{a},\boldsymbol{x}} \sum_{r\in\mathcal{R}} \sum_{l_i\in\mathcal{B}_r} x_{rl_i} L_{rl_i}$$

s.t. Constraints (8.4), (8.5), (8.6) and (8.7) holds

$$0 \leq a_l \leq C_l, 0 \leq x_{rl_i} \leq 1 \tag{8.10}$$

Table 8.3: **FINDLPALLOCATION**$(\mathcal{R}, \mathcal{B}, \mathcal{A})$

The LP solution provides a sense of best possible fractional allocation and therefore, we utilise this solution to compute the final integral and feasible solution in the second stage. Let $\hat{\boldsymbol{a}}$ denotes the allocation that we obtained from the LP solution

[2]Note that, our two-stage optimisation is a single-shot (i.e., non- iterative) hierarchical approach.

of Table (8.3). We use the following rounding approach to obtain an initial integral allocation, $\bar{\boldsymbol{a}}$ from $\hat{\boldsymbol{a}}$.

$$\bar{a}_l = \begin{cases} \lceil \hat{a}_l \rceil & \text{if } \hat{a}_l - \lfloor \hat{a}_l \rfloor \geq \rho \\ \lfloor \hat{a}_l \rfloor & \text{Otherwise} \end{cases}$$

Precisely, the integral allocation for base l (i.e., $\bar{a}_l$) is set to the next integer value of $\hat{a}_l$ (i.e., $\lfloor \hat{a}_l \rfloor + 1$) if the fractional part of $\hat{a}_l$ (i.e., the difference between $\hat{a}_l$ and its integral part, $\lfloor \hat{a}_l \rfloor$) is greater than ρ ($0.5 < \rho < 1$) and otherwise we fix the value of $\bar{a}_l$ as the integral part of $\hat{a}_l$. $\bar{\boldsymbol{a}}$ provides a valid allocation for a subset of ambulances. As all the ambulances are homogeneous, it does not matter which specific subset of ambulances are allocated a priori. However, $\bar{\boldsymbol{a}}$ does not allocate the entire fleet of ambulances. We then utilise the values of $\bar{\boldsymbol{a}}$ to guide the original MILP of Table (8.1). In the second stage, we essentially solve the MILP of Table (8.1) with additional set of constraints (8.11), which enforce that at least $\bar{a}_l$ ambulances need to be allocated in base $l \in \mathcal{B}$. The second stage optimisation model is shown compactly in Table (8.4). Note that the optimisation problem of Table (8.4) is less computationally challenging than the one of Table (8.1). This is so, because we manually fix the allocation for a subset of ambulances and therefore, the optimiser need to search for an allocation of only $|\mathcal{A}| - |\bar{\boldsymbol{a}}|$ ambulances. However, the complexity of the second stage optimisation model depends on the value of the given parameter ρ. If the value of ρ is high (i.e., close to 1), then the number of allocated ambulances from the first stage solution (i.e., $\bar{\boldsymbol{a}}$) would be low and therefore, the second stage problem would be harder to solve. On the other hand, the overall solution quality will deteriorate if the value of ρ is low. We experimentally observe that the right trade-off between the runtime and the solution quality can be found if ρ is equal to 0.95 and therefore, we set the value of ρ to 0.95 in our default settings of experiments. This approach is referred to as **TwoStage**.

$$\max_{\boldsymbol{a},\boldsymbol{x}} \sum_{r\in\mathcal{R}} \sum_{l_i\in\mathcal{B}_r} x_{rl_i} L_{rl_i}$$

s.t. Constraints (8.4), (8.5), (8.6), (8.7) and (8.8) holds

$$a_l \geq \bar{a}_l \qquad \forall l \in \mathcal{B} \tag{8.11}$$

Table 8.4: **FINDTWOSTAGEALLOCATION**$(\mathcal{R}, \mathcal{B}, \mathcal{A})$

8.2 Experimental Results

We conduct experiments on two real-world data sets. We obtain the *dataset-1* from a real-world EMS in the form of anonymous and modified sample of request logs. The *dataset-2* [3] is adopted from Yue et al. (2012). Both the data sets provide details of emergency requests over a certain period. Each request log contains the following information (a) Incident location; (b) Arrival time; (c) A set of feasible nearby bases from where the request can be assisted; (d) Response time from each of the feasible base to incident location; and (e) Round-about time for each of the feasible base. While these specific details might not always be readily available for real deployment, as indicated in Ghosh and Varakantham (2016), we can estimate them using a straightforward method. As the geographical locations of the requests, hospitals and bases are available in the historical data sets, we can compute the set of feasible nearby bases and predict the response and round-about times for each of these bases.

Results on dataset-1: The *dataset-1* contains a fleet of 35 ambulances and 35 base stations. We have an anonymous request sample over a period of six months. We divide our 6 months of data set into two parts - first 3 months is used for training purpose to generate the allocation policies and the performance of these policies are tested on other 3 months of data. We evaluate the performance of our approach by employing a real-life event-driven simulation model (refer to Algorithm 6 for the details of simulator) which follows the nearest available ambulance dispatch

[3] An anonymous sample is available here: http://projects.yisongyue.com/ambulance_allocation/

rule. We compare our approach against the existing greedy approach provided by Yue et al. (2012), which incrementally add one ambulance in each iteration using a real-life event driven simulator until the entire fleet is allocated. We refer to this approach as **Greedy**. We do not provide comparison results against the approach proposed by Saisubramanian et al. (2015), as their objective is to miminize response time for a fixed percentile of requests, which is different from the metric of interest in our paper (i.e., maximising the number of requests served within a threshold time bound). Due to different objective functions, we experimentally observe that the approach from Saisubramanian et al. (2015) produced worse quality solutions than the greedy approach proposed by Yue et al. (2012).

Solution quality of the heuristic approaches: The ILP or CP cannot solve the large-scale problems optimally within our imposed time-limit of two hours. However, these exact approaches can be solved optimally for very small problems with only a few hundred requests. We experimentally observe that our heuristic approaches provide good quality solutions in comparison to the optimal for these small instances. For instance, our two-stage optimisation approach is only 1.5% away from the optimal. However, in our problem instances, we have a few thousands training incident requests. So, we can only get a sense of ILP optimum from the optimality gap provided by black-box solvers such as CPLEX. Unfortunately, these gaps are loose and are far away from the optimal solution (specifically for the ILP) and hence are unreliable.

	Greedy	**ILP**	**CP**	**Relaxation**	**TwoStage**
Mon	58.97 %	57.96 %	57.56 %	60.10 %	60.46 %
Tue	59.15 %	44.95 %	56.12 %	60.76 %	60.16 %
Wed	59.27 %	47.51 %	57.43 %	61.76 %	62.52 %
Thu	60.27 %	59.32 %	57.96 %	62.86 %	62.42 %
Fri	59.87 %	59.81 %	52.83 %	61.68 %	61.92 %
Sat	63.65 %	63.16 %	63.47 %	66.69 %	66.80 %
Sun	65.76 %	67.44 %	67.05 %	70.06 %	69.46 %

Table 8.5: Performance comparison on testing data of *dataset-1*

Performance comparison: We now demonstrate the performance comparison between our approaches and the greedy approach on testing data of *dataset-1*. Table (8.5) shows the comparison results for all the weekdays. Our key performance metric is the percentage of requests that are served within 8 minutes. We observe that the solution quality of CP and ILP is worse than the existing greedy approach. This is so because we impose a time-limit of two hours for both the approaches and none of these approaches can be solved optimally within our time-limit. On an average, the optimality gap for ILP was more than 20%. However, both our heuristic approaches (i.e., two-stage optimisation and relaxation approach) outperform the greedy approach. On an average, both these heuristics can serve around 63.4% requests within 8 minutes. Most importantly, for all the weekdays, our heuristic approaches serve around 2.4% additional requests within the threshold time bound (i.e., 8 minutes) over the existing greedy approach.

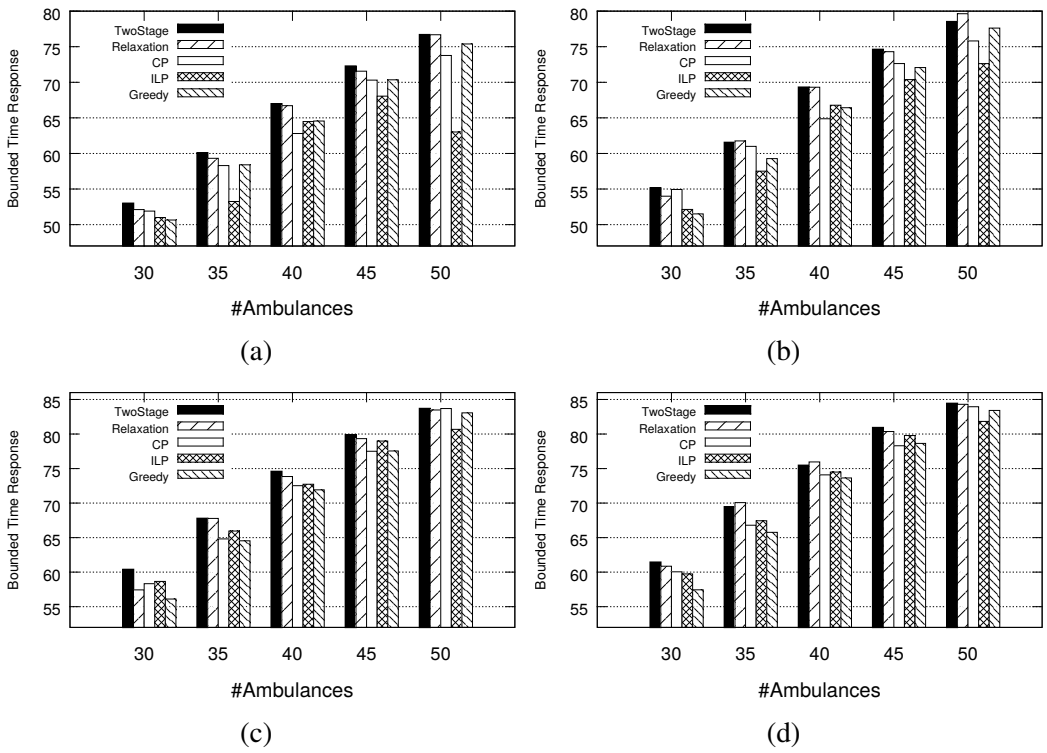


Figure 8.1: Performance comparison by varying ambulance fleet size: (a) Training results on weekday; (b) Testing result on weekday; (c) Training result on weekend; and (d) Testing result on weekend.

Effect of ambulance fleet size: In this thread of results, we demonstrate the performance comparison between different approaches on *dataset-1* by varying the ambulance fleet size. We use the same training and testing data set for these experiments. Figure 8.1(a)-(b) depict the performance comparison on training and testing data set for one of the weekdays. In the X-axis we vary the number of ambulances and Y-axis shows the percentage of requests served within 8 minutes. Due to the scalability issue, ILP and CP yield poor quality solution than our heuristic approaches for all the settings of ambulance fleet size. Both the two-stage optimisation and relaxation approach always outperforms the greedy approach. More interestingly, the gain over the greedy approach increases if we decrease the number of ambulances. This insight clearly indicates that the performance of the existing greedy approach degrades for EMS with limited resources and our approaches are suitable to tackle such scenarios.

Figure 8.1(c)-(d) demonstrate the performance comparison on training and testing data set for one of the weekends. We observe a similar pattern for these results. Our heuristic approaches always produce better solution than the greedy approach, specially when we have fewer ambulances. For instance, the performance gain of our two-stage optimisation approach over the greedy approach on testing data set increases from 0.6% to 4.2% when the ambulance fleet size is reduced from 50 to 30.

Results on dataset-2: The *dataset-2* contains a fleet of 58 ambulances and 58 base stations. We have 1500 weeks of request logs. We use Sample Average Approximation [SAA] (Verweij et al., 2003) for validation and performance estimation. We generate 10 policies for each of the weekdays, where each policy is generated using request logs of that particular weekday for 10 consecutive weeks (e.g., the second policy for Monday is generated using requests of all the Mondays from week 11 to week 20). Then we identify the policy with best validation performance for each of the weekdays separately over 500 weeks of request logs. Finally, we evaluate the

performance of the validated policies on 3 test data sets each of which contains 300 weeks of request logs.

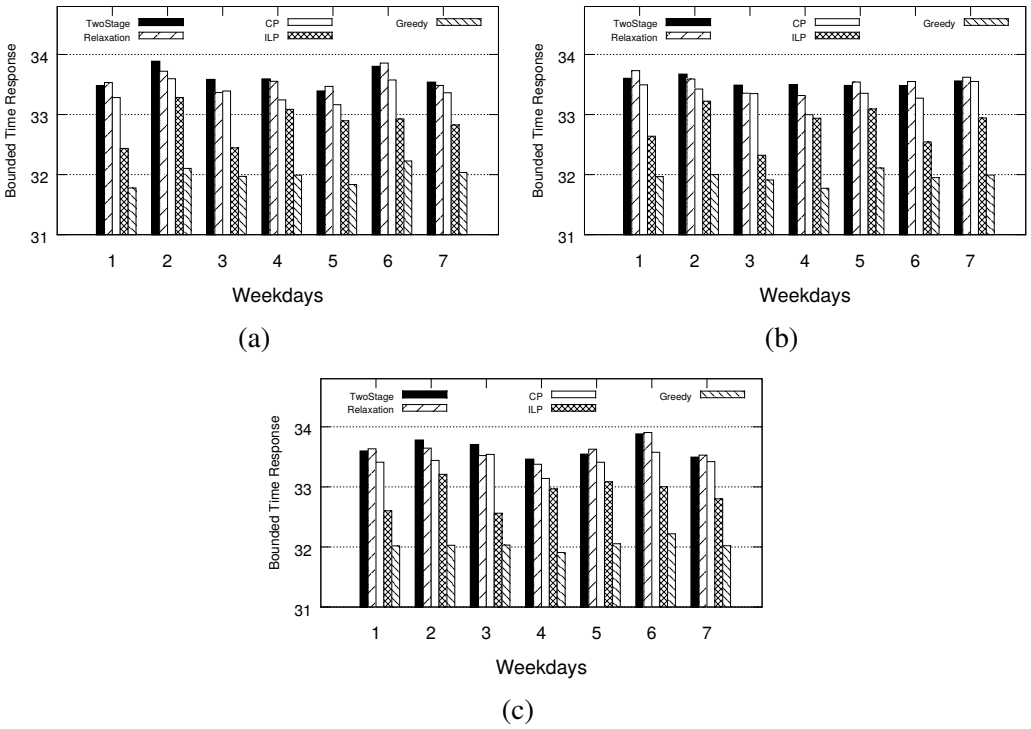


Figure 8.2: Performance comparison on *dataset-2*: (a) First testing set; (b) Second testing set; and (c) Third testing set.

We now present the performance comparison results on the testing data of *dataset-2*. Figure 8.2 depicts the comparison on three testing data sets. The X-axis denotes the weekdays and the Y-axis represents the percentage of requests served within 15 minutes. As each of the test cases involves 300 weeks of request logs, we report the average utility using SAA. Figure 8.2(a) plots the bounded time response value for the first testing data set. As shown clearly, our two-stage optimisation almost always provides the best performance over other approaches, while our relaxation approach is proven to be highly competitive with two-stage optimisation approach. Although, none of the CP and ILP can be solved optimally, CP provides a reasonably better quality solution than ILP within the time-limit of two hours. For all the weekdays, our two-stage optimisation approach provides at least 1.5% gain in bounded time response value over the greedy approach.

Figure 8.2(b) and 8.2(c) depict the performance comparison results on second and third testing data sets. We observe a consistent pattern on the performance over all the three testing data sets. As shown in Figure 8.2(b), both the two-stage and relaxation heuristics are able to serve 1.6% extra requests within the threshold response time over the greedy approach for second testing data set. Figure 8.2(c) demonstrates that, for third testing data set, both the two-stage and relaxation heuristics provide at least 1.45% performance gain on all the weekdays and improve the average bounded time response value by 1.6% over the greedy approach.

8.3 Discussion

We now discuss about directions that we explored in addition to the approaches described above. We believe that these approaches have potential and can be improved in the future.

Benders decomposition: By exploiting observation (4), we can ensure that continuous assignment guarantees to provide an optimal and integral solution for our original problem if all the bases have single capacity. A straightforward method to translate our problem into single capacity base station problem is to create C_l single capacity bases at the location of base l. The response and round-about times for a request r from all the C_l bases will be same as the response and round-about time for base l. The number of feasible nearby bases for a request r will now increase from $|\mathcal{B}_r|$ to $\sum_{l \in \mathcal{B}_r} C_l$. Once we have a continuous assignment problem with single capacity bases, it will be an ideal ground for applying Benders decomposition (Benders, 1962), where master solves the allocation problem and slave takes the assignment decisions. Our initial experiments show that due to significant increases in the number of variables and constraints, this particular translation results in a large optimality gap even with Benders decomposition. However, these insights lead to a promising direction for improving our solutions in the future.

SAT representation: The reformulated single capacity base problem is a 0/1 inte-

ger program and can be translated to a satisfiability problem. As we have an optimisation problem, we can translate it to a partial max-SAT (Argelich & Manya, 2007; Koshimura, Zhang, Fujita, & Hasegawa, 2012) representation, where our objective function can be converted to following soft clauses: $\boldsymbol{x}_{rl}, \quad \forall r \in \mathcal{R}, \exists l, T_{r.s,l} \leq \Delta$ Constraints (8.4) can be translated to a set of hard clauses (8.12). The clauses (8.13)-(8.14) are equivalent to constraints (8.5). The clauses (8.15) exactly represent the constraints (8.7).

$$\neg \boldsymbol{x}_{rl_i} \vee \neg \boldsymbol{x}_{rl_j} \qquad \forall r \in \mathcal{R}, l_i, l_j \neq l_i \in \mathcal{B}_r \tag{8.12}$$

$$\boldsymbol{a}_{l_i} \vee \neg \boldsymbol{x}_{rl_i} \qquad \forall r \in \mathcal{R}, l_i \in \mathcal{B}_r \tag{8.13}$$

$$\neg \boldsymbol{x}_{rl_i} \vee \neg \boldsymbol{x}_{jl_i} \qquad \forall r \in \mathcal{R}, l_i \in \mathcal{B}_r, j \in P_r^{l_i} \tag{8.14}$$

$$\neg \boldsymbol{a}_{l_i} \vee_{k \leq i} \boldsymbol{x}_{rl_k} \vee_{j \in P_r^{l_i}} \boldsymbol{x}_{jl_i} \qquad \forall r \in \mathcal{R}, l_i \in \mathcal{B}_r \tag{8.15}$$

However, to the best of our knowledge, there is no explicit way to represent the constraints (8.6) as SAT clauses. A brute-force approach would be employing the following set of hard clauses (8.16)-(8.18), where d_{cl} variable is set to 1 if the ambulance c is allocated to base l. However, this brute-force approach increases the number of variables and clauses significantly and therefore, state-of-the-art partial max-SAT solvers fail to solve it efficiently. So, discovering an efficient and compact SAT representation would be a potential future direction.

$$\neg \boldsymbol{d}_{cl_i} \vee \neg \boldsymbol{d}_{cl_j} \qquad \forall c \in \mathcal{A}, l_i, l_j \neq l_i \in \mathcal{B} \tag{8.16}$$

$$\neg \boldsymbol{d}_{c_i l} \vee \neg \boldsymbol{d}_{c_j l} \qquad \forall c_i, c_j \neq c_i \in \mathcal{A}, l \in \mathcal{B} \tag{8.17}$$

$$\neg \boldsymbol{a}_l \vee_c \boldsymbol{d}_{cl} \qquad \forall l \in \mathcal{B} \tag{8.18}$$

8.4 Summary

In this chapter, we provide dispatch guided optimisation approaches for effective and dynamic allocation of ambulances to base locations. We propose a novel op-

timisation model by incorporating the real-world ambulance dispatch strategy and show that the optimisation model follows the real evolution dynamics of EMS. As the proposed optimisation model suffers scalability issues, we provide two novel heuristic approaches to increase scalability to large number of emergency incidents. The empirical results on two real-world EMS data sets demonstrate that our heuristic approaches always outperform the existing best known approach.

Chapter 9

Conclusion

The rapid urbanisation and uncoordinated usage of resources result in a natural mismatch of supply and demand of public resources, which has considerably impacted the efficacy of urban transportation and health & safety. Moreover, due to these reasons, today's cities are experiencing major concerns such as global warming, traffic congestion, high carbon emission, delay in emergency response, etc. To alleviate these growing concerns, this thesis presents novel techniques to improve the efficiency of urban environments using intelligent strategic and operational decisions.

To reduce the usage of private vehicles and increase the utilisation of green mode of transportation such as bike sharing systems, three operational decision making approaches have been proposed so as to proactively redistribute the idle bikes to better meet the future demand: (a) In case of consistent demand pattern, a dynamic redistribution model is proposed by considering future demand for multiple time periods. In addition, decomposition and abstraction mechanisms are presented to speed up the solution process; (b) In case of demand with high variance, an online and robust redistribution solution is proposed by exploiting the possible uncertainties in demand; and (c) On the contrary to using fuel burning mode of vehicles for redistribution, bike trailers are utilised for redistribution tasks and a budget feasible mechanism is designed to incentivise the customers for executing those redistribution tasks. To reduce the response times for emergency needs, the following key

ideas have been proposed: (a) A greedy approach is presented for the placement of base stations on top of an optimisation model for the allocation of ambulances to bases; and (b) A complete data-driven optimisation model is demonstrated by incorporating all the real-world dynamics of EMS for dynamic redistribution of ambulances. The above mentioned techniques are validated on real-world data sets and proven to be effective than the current practices.

Future Research: This thesis explores new directions in urban transportation and health & safety, that need further examination and analysis. For the future research, we highlight the following open questions and directions:

Demand estimation: The efficacy of demand estimation techniques is an important pillar for designing intelligent dynamic matching solutions for BSSs. The demand estimation in a BSS has two critical components: (a) identifying the unobserved lost demand due to unavailability of bikes or open docks; and (b) estimating the future demand for bikes and open docks. Identifying unobserved lost demand is critical and challenging in many real-world problems including retail planning. While we propose a few heuristic approaches in Chapter 3 for estimating unobserved lost demand, the accuracy of these techniques cannot be verified due to unavailability of realised demand information. Therefore, a detailed investigation is required in future to address this issue. A thought-provoking insight is that this unobserved lost demand prediction problem at an empty station is equivalent to predicting customer demand at a location where no bike station is situated for a particular period (i.e., when the station is empty). Recently, Singhvi, Singhvi, Frazier, Henderson, O'Mahony, Shmoys, and Woodard (2015) propose insights for predicting spatio-temporal bike usage demand at locations, where bike stations are not placed, using exogenous and relevant features such as spatio-temporal taxi demand and meteorology information. Hence, leveraging these insights and techniques would be a potential future direction for estimating unobserved lost demand in BSSs.

For representing the future customer demand, we employ *Poisson* distribution

where the mean of the distribution is learnt from historical data. While *Poisson* distribution has been widely adopted to represent the random demand arrival process, it may not be the best approach if the demand follows consistent patterns. Recently, Li, Zheng, Zhang, and Chen (2015) generated a set of clusters by grouping nearby bike stations based on their geographical proximity and historical transition patterns of bikes among stations and proposed machine learning technique (based on Gradient Boosting Regression Tree method) along with efficient inference methods for predicting future demand for these clusters. However, for improving efficacy of dynamic repositioning, we need the understanding of station-wise temporal demand. Therefore, there is a practical need to further examine these sophisticated machine learning techniques for an accurate prediction of future demand for individual stations.

Amalgamation of proactive and reactive decision making: This thesis focuses on data-driven proactive decision making, which is effective for regular and consistent scenarios. However, if there is a sudden and adverse change in the demand pattern (e.g., a major concert or festival in the city, outburst of infectious diseases) which was never observed in the past, then the proactive solutions fail to adapt such evolution. Therefore, another promising direction is to develop solutions that simultaneously exploit the proactive decisions learnt from the historical data and explore the current scenarios to generate reactive decisions so as to adapt the adverse changes in demand pattern.

Generalisation of spatio-temporal abstraction: Large-scale urban decision making problems generally observe an inherent trade-off between the solution quality and runtime complexity. To tackle this challenge, we employ spatio-temporal abstraction techniques. In this thesis, we empirically show that our proposed geographical proximity based abstraction technique and temporal abstraction in the form of decision period produce reasonably good quality solutions. However, as grouping of stations with similar behaviour can potentially improve the quality of dynamic repositioning solutions, an open and promising direction is to develop a

spatio-temporal abstraction technique that can simultaneously take into account the historical demand patterns, the resource transition patterns and the spatial information of stations and furthermore, can be generalised to other similar urban settings.

In addition to the previously mentioned open questions, our proposed solutions can be extended in the following directions to address more complex dynamics, scenarios and boundary conditions:

Enhancing solutions for station-less BSS: Our proposed techniques for operational decision making in BSS assume that there is a fixed set of base stations with finite docking capacity. However, several real-world BSS operators adopt the notion of mobile docking (i.e., station-less BSS), where bikes (embedded with GPS tracker) can be dropped anywhere in the city. Therefore, extending our solutions for the BSS with mobile docking would be a promising future direction. A straightforward direction to employ our solutions for these systems is to split the city into a set of clusters or zones and consider each zone as a base station with infinite capacity. However, there is a trade-off between scalability and usefulness of repositioning solutions while choosing the size or number of zones and therefore, a critical issue is to efficiently divide the city into various zones so that the dynamic repositioning problems can be solved without compromising the solution quality. Furthermore, our mechanism from Chapter 5 can be extended for incentivising customers to change their destination location in order to rebalance these station-less BSSs.

In addition, our solutions can be extended for the following incremental problems:

- *Online policy with multi-step demand:* We show that the offline policies are efficient if the demand patterns are consistent, while we need a robust and online solution approach if the future demand is unpredictable and has higher variance. Therefore, generating online policies by considering the future expected demand for multiple time steps would be an interesting direction to better account for the future demand surges.

- *Couple the redistribution problem of vehicles and bike trailers:* There is a

practical need to jointly model the dynamic redistribution problem for vehicles and trailers and discover an efficient solution while ensuring the central budget constraint. Furthermore, a budget feasible mechanism needs to be designed by considering the uncertainties in completion time of the trailer tasks.

Enhancing EMS solutions for multi-tiered incidents: Our proposed techniques for strategic and operational decision making in EMS assume that all the incidents are identical in terms of criticality. However, some real-world EMS operators have introduced the notion of multi-tiered incidents, i.e., the incoming emergency requests are categorised into various priority level and are treated with difference risk factor. For instance, the higher priority requests are generally critical in nature and the pre-defined threshold response time for these requests is lower than other types of requests. Therefore, extending our proposed models with tiered incidents and tiered ambulances would be a promising future direction.

Exploring multi-agent aspect of resource redistribution problems: In this thesis, we explore problems from a single agent's perspective that optimise a global objective function. However, many real-world resource redistribution problems involve self-interested rational agents who are competitive in nature (e.g., taxi drivers, private EMS operators). To perform efficiently in these scenarios, the solution approaches must be able to tackle the uncertainties online and reason with human behavioural models.

To summarise, this thesis has explored several novel and efficient data-driven algorithms and techniques for proactive redistribution of urban resources. Due to the growing interests for *Internet-of-things* and recent advancements in *machine learning* techniques, a huge amount of "meaningful" urban data is available from a large number of smart computing devices that are embedded in various systems from transportation to buildings. Therefore, I believe that proactive redistribution of urban resources by utilising these large-scale data sets would be a fertile ground for building sustainable and smart cities.

Bibliography

Andersson, T., & Värbrand, P. (2007). Decision support tools for ambulance dispatch and relocation. *Journal of the Operational Research Society*, *58*(2), 195–201.

Angelopoulos, A., Gavalas, D., Konstantopoulos, C., Kypriadis, D., & Pantziou, G. (2016). An optimization model for the strategic design of a bicycle sharing system: A case study in the city of Athens. In *Proceedings of the 20th Pan-Hellenic Conference on Informatics*, p. 5. ACM.

Argelich, J., & Manya, F. (2007). Partial max-sat solvers with clause learning. *SAT*, *2007*, 28–40.

Barth, M., Todd, M., & Xue, L. (2004). User-based vehicle relocation techniques for multiple-station shared-use vehicle systems. In *Transportation Research Board Annual Conference CD-ROM*. Citeseer.

Battarra, M., Erdoğan, G., & Vigo, D. (2014). Exact algorithms for the clustered vehicle routing problem. *Operations Research*, *62*(1), 58–71.

Benchimol, M., Benchimol, P., Chappert, B., De La Taille, A., Laroche, F., Meunier, F., & Robinet, L. (2011). Balancing the stations of a self service "bike hire" system. *RAIRO-Operations Research*, *45*(1), 37–61.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik*, *4*(1), 238–252.

Bertsekas, D. P. (1999). *Nonlinear programming* (2nd edition). Athena Scientific.

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. *Advances in Complex Systems*, *14*(03), 415–438.

Borgnat, P., Abry, P., Flandrin, P., & Rouquier, J.-B. (2009). Studying Lyon's Vélo'V: A statistical cyclic model. In *Proceedings of the European Conference on Complex Systems (ECCS)*. Complex System Society.

Brotcorne, L., Laporte, G., & Semet, F. (2003). Ambulance location and relocation models. *European journal of operational research*, *147*(3), 451–463.

Cervero, R. (2013). *Transport infrastructure and the environment: Sustainable mobility and urbanism*. Boca Raton, FL: CRC Press.

Chalasani, P., & Motwani, R. (1999). Approximating capacitated routing and delivery problems. *SIAM Journal on Computing*, *28*(6), 2133–2149.

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

Chow, Y., & Yu, J. Y. (2015). Real-time bidding based vehicle sharing. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pp. 1829–1830. International Foundation for Autonomous Agents and Multiagent Systems.

Church, R., & Velle, C. R. (1974). The maximal covering location problem. *Papers in regional science*, *32*(1), 101–118.

Clarke, E. H. (1971). Multipart pricing of public goods. *Public choice*, *11*(1), 17–33.

Coelho, L. C., Cordeau, J.-F., & Laporte, G. (2013). Thirty years of inventory-routing. *Transportation Science*, *48*(1), 1–19.

Contardo, C., Morency, C., & Rousseau, L.-M. (2012). Balancing a dynamic public bike-sharing system. Tech. rep., CIRRELT.

Dargay, J., Gately, D., & Sommer, M. (2007). Vehicle ownership and income growth, worldwide: 1960-2030. In *The Energy Journal*, pp. 143–170. JSTOR.

Dasgupta, S., Papadimitriou, C. H., & Vazirani, U. V. (2006). *Algorithms*. McGraw-Hill, Inc., New York, NY.

Dean, T., & Lin, S.-H. (1995). Decomposition techniques for planning in stochastic domains. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Vol. 2, pp. 1121–1127.

Di Gaspero, L., Rendl, A., & Urli, T. (2013). A hybrid ACO+CP for balancing bicycle sharing systems. In *Hybrid Metaheuristics*, pp. 198–212. Springer, Berlin, Heidelberg.

Di Gaspero, L., Rendl, A., & Urli, T. (2016). Balancing bike sharing systems with constraint programming. *Constraints*, *21*(2), 318–348.

Erdoğan, G., Laporte, G., & Wolfler Calvo, R. (2014). The static bicycle relocation problem with demand intervals. *European Journal of Operational Research*, *238*(2), 451–457.

Fan, M. (2008). Creating a car culture in China. In *Washington Post*.

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

Fishman, E., Washington, S., & Haworth, N. L. (2014). Bike share's impact on car use: Evidence from the United States, Great Britain, and Australia. *Transportation Research Part D: Transport and Environment*, *31*, 13–20.

Fricker, C., & Gast, N. (2016). Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity. *EURO Journal on Transportation and Logistics*, *5*(3), 261–291.

Froehlich, J., Neumann, J., & Oliver, N. (2008). Measuring the pulse of the city through shared bicycle programs. In *Proceedings of the International Workshop on Urban, Community, and Social Applications of Networked Sensing Systems (UrbanSense)*, pp. 16–20.

Furmston, T., & Barber, D. (2011). Lagrange dual decomposition for finite horizon Markov decision processes. In *Proceedings of the Machine Learning and*

Knowledge Discovery in Databases: European Conference (ECML PKDD), pp. 487–502.

Gendreau, M., Laporte, G., & Semet, F. (2006). The maximal expected coverage relocation problem for emergency vehicles. *Journal of the Operational Research Society*, *57*(1), 22–28.

George, D. K., & Xia, C. H. (2011). Fleet-sizing and service availability for a vehicle rental system via closed queueing networks. *European Journal of Operational Research*, *211*(1), 198–207.

Geramifard, A., Doshi, F., Redding, J., Roy, N., & How, J. (2011). Online discovery of feature dependencies. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 881–888.

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

Ghosh, S., & Varakantham, P. (2016). Strategic planning for setting up base stations in emergency medical systems. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, pp. 385–393.

Ghosh, S., & 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. (2018). Dispatch guided allocation optimization for effective emergency response. In *AAAI Conference on Artificial Intelligence (AAAI)*.

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

Giunchiglia, F., & Walsh, T. (1992). A theory of abstraction. *Artificial Intelligence*, *57*(2), 323–389.

Golovin, D., Krause, A., Gardner, B., Converse, S. J., & Morey, S. (2011). Dynamic resource allocation in conservation planning.. In *AAAI*, Vol. 11, pp. 1331–1336.

Gordon, G. J., Varakantham, P., Yeoh, W., Lau, H. C., Aravamudhan, A. S., & Cheng, S.-F. (2012). Lagrangian relaxation for large-scale multi-agent planning. In *Proceedings of the The IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, Vol. 2, pp. 494–501.

Groves, T. (1973). Incentives in teams. In *Econometrica: Journal of the Econometric Society*, pp. 617–631. JSTOR.

Guestrin, C., & Gordon, G. (2002). Distributed planning in hierarchical factored MDPs. In *Proceedings of the Uncertainty in artificial intelligence (UAI)*, pp. 197–206.

Hakimi, S. L. (1964). Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations research*, *12*(3), 450–459.

Hernández-Pérez, H., & Salazar-González, J.-J. (2004). Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. *Transportation Science*, *38*(2), 245–255.

Hochbaum, D. S., & Shmoys, D. B. (1985). A best possible heuristic for the k-center problem. *Mathematics of operations research*, *10*(2), 180–184.

Huang, R., Kim, S., & Menezes, M. B. (2010). Facility location for large-scale emergencies. *Annals of Operations Research*, *181*(1), 271–286.

Jagtenberg, C., Bhulai, S., & van der Mei, R. (2017). Optimal ambulance dispatching. In *Markov Decision Processes in Practice*, pp. 269–291. Springer.

Jia, H., Ordóñez, F., & Dessouky, M. (2007). A modeling framework for facility location of medical services for large-scale emergencies. *IIE transactions*, *39*(1), 41–55.

Kabra, A., Belavina, E., & Girotra, K. (2015). Bike-share systems: accessibility and availability. Tech. rep., Chicago Booth No. 15-04.

Kek, A. G., Cheu, R. L., Meng, Q., & Fung, C. H. (2009). A decision support system for vehicle relocation operations in carsharing systems. *Transportation Research Part E: Logistics and Transportation Review*, *45*(1), 149–158.

Knoblock, C. (1993). *Generating abstraction hierarchies: An automated approach to reducing search in planning*, Vol. 214. Springer Science & Business Media, New York.

Knoblock, C. A. (1991). Search reduction in hierarchical problem solving. In *AAAI Conference on Artificial Intelligence (AAAI)*, pp. 686–691.

Kök, A. G., & Fisher, M. L. (2007). Demand estimation and assortment optimization under substitution: methodology and application. *Operations Research*, *55*(6), 1001–1021.

Koshimura, M., Zhang, T., Fujita, H., & Hasegawa, R. (2012). Qmaxsat: A partial max-sat solver. *Journal on Satisfiability, Boolean Modeling and Computation*, *8*, 95–100.

Krause, A., Singh, A., & Guestrin, C. (2008). Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *The Journal of Machine Learning Research*, *9*, 235–284.

Kumar, A., Singh, S., Gupta, P., & Parija, G. (2014). Near-optimal nonmyopic contact center planning using dual decomposition. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, pp. 395–403.

Kumar, A., Wu, X., & Zilberstein, S. (2012). Lagrangian relaxation techniques for scalable spatial conservation planning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 309–315.

Kumar, V. P., & Bierlaire, M. (2012). Optimizing locations for a vehicle sharing system. In *Swiss Transport Research Conference. http://www.strc.ch/conferences/2012/Kumar_Bierlaire.pdf*.

Laporte, G., Meunier, F., & Wolfler Calvo, R. (2015). Shared mobility systems. *4OR*, *13*(4), 341–360.

Lathia, N., Ahmed, S., & Capra, L. (2012). Measuring the impact of opening the London shared bicycle scheme to casual users. *Transportation research Part C: Emerging technologies*, *22*, 88–102.

Lees-Miller, J. D., Hammersley, J. C., & Wilson, R. E. (2010). Theoretical maximum capacity as benchmark for empty vehicle redistribution in personal rapid transit. *Transportation Research Record: Journal of the Transportation Research Board*, *2146*(1), 76–83.

Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., & Glance, N. (2007). Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 420–429. ACM.

Leurent, F. (2012). Modelling a vehicle-sharing station as a dual waiting system: stochastic framework and stationary analysis. In *https://hal.archivesouvertes.fr/hal-00757228*.

Li, L., Walsh, T. J., & Littman, M. L. (2006). Towards a unified theory of state abstraction for MDPs. In *Proceedings of the International Symposium on Artificial Intelligence and Mathematics (ISAIM)*, pp. 1–10.

Li, Y., Zheng, Y., Zhang, H., & Chen, L. (2015). Traffic prediction in a bike-sharing system. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, p. 33. ACM.

Lin, J.-R., & Yang, T.-H. (2011). Strategic design of public bicycle sharing systems with service level constraints. *Transportation research Part E: Logistics and transportation review*, *47*(2), 284–294.

Lin, J.-R., Yang, T.-H., & Chang, Y.-C. (2013). A hub location inventory model for bicycle sharing system design: formulation and solution. *Computers & Industrial Engineering*, *65*(1), 77–86.

Lowalekar, M., Varakantham, P., Ghosh, S., Jena, S. D., & Jaillet, P. (2017). Online repositioning in bike sharing systems. In *International Conference on Automated Planning and Scheduling (ICAPS)*.

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

Lu, T., & Boutilier, C. (2015). Value-directed compression of large-scale assignment problems. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 1182–1190.

Mahadevan, S. (2002). Spatio-temporal abstraction of stochastic sequential processes. In *Proceedings of the Abstraction, Reformulation, and Approximation: International Symposium (SARA)*, pp. 33–50.

Mareček, J., Shorten, R., & Yu, J. Y. (2016). Pricing vehicle sharing with proximity information. In *Big Data and Smart City (ICBDSC), 2016 3rd MEC International Conference on*, pp. 1–7. IEEE.

Martinez, L. M., Caetano, L., Eiró, T., & Cruz, F. (2012). An optimisation algorithm to establish the location of stations of a mixed fleet biking system: An application to the city of Lisbon. *Procedia-Social and Behavioral Sciences*, *54*, 513–524.

Maxwell, M. S., Henderson, S. G., & Topaloglu, H. (2009). Ambulance redeployment: An approximate dynamic programming approach. In *Winter Simulation Conference*, pp. 1850–1860. Winter Simulation Conference.

Maxwell, M. S., Henderson, S. G., Topaloglu, H., et al. (2013). Tuning approximate dynamic programming policies for ambulance redeployment via direct search. *Stochastic Systems*, *3*(2), 322–361.

Meddin, R., & DeMaio, P. (2016). The bike-sharing world map. In *http://www.bikesharingworld.com/*.

Minoux, M. (1978). Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*, pp. 234–243. Springer.

Musalem, A., Olivares, M., Bradlow, E. T., Terwiesch, C., & Corsten, D. (2010). Structural estimation of the effect of out-of-stocks. *Management Science*, *56*(7), 1180–1197.

Nair, R., & Miller-Hooks, E. (2011). Fleet management for vehicle sharing operations. *Transportation Science*, *45*(4), 524–540.

Nair, R., Miller-Hooks, E., Hampshire, R. C., & Bušić, A. (2013). Large-scale vehicle sharing systems: Analysis of Vélib. *International Journal of Sustainable Transportation*, *7*(1), 85–106.

Neiderud, C.-J. (2015). How urbanization affects the epidemiology of emerging infectious diseases. *Infection ecology & epidemiology*, *5*.

Nemhauser, G. L., Wolsey, L. A., & Fisher, M. L. (1978). An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, *14*(1), 265–294.

O'Mahony, E., & Shmoys, D. B. (2015). Data analysis and optimization for (Citi) bike sharing. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 687–694.

Papageorgiou, D. J., Nemhauser, G. L., Sokol, J., Cheon, M.-S., & Keha, A. B. (2014). MIRPLib - A library of maritime inventory routing problem instances: Survey, core model, and benchmark results. *European Journal of Operational Research*, *235*(2), 350–366.

Peden, M., Scurfield, R., Sleet, D., Mohan, D., Hyder, A. A., Jarawan, E., Mathers, C. D., et al. (2004). World report on road traffic injury prevention..

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

Raidl, G. R., Hu, B., Rainer-Harbach, M., & Papazek, P. (2013). Balancing bicycle sharing systems: Improving a VNS by efficiently determining optimal loading operations. In *Proceedings of th Hybrid Metaheuristics: International Workshop (HM)*, pp. 130–143.

Rainer-Harbach, M., Papazek, P., Hu, B., & Raidl, G. R. (2013). *Balancing bicycle sharing systems: A variable neighborhood search approach*. Springer Berlin Heidelberg.

Raviv, T., & Kolka, O. (2013). Optimal inventory management of a bike-sharing station. *IIE Transactions*, *45*(10), 1077–1093.

Raviv, T., Tzur, M., & 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.

Restrepo, M., Henderson, S. G., & Topaloglu, H. (2009). Erlang loss models for the static deployment of ambulances. *Health care management science*, *12*(1), 67–79.

Rogers, D. F., Plante, R. D., Wong, R. T., & Evans, J. R. (1991). Aggregation and disaggregation techniques and methodology in optimization. *Operations Research*, *39*(4), 553–582.

Ross, S. (1983). *Stochastic processes*, Vol. 23. John Wiley & Sons New York.

Rudloff, C., & Lackner, B. (2013). Modeling demand for bicycle sharing system–neighboring stations as a source for demand and a reason for structural breaks. In *Transportation Research Board Annual Meeting*.

Saisubramanian, S., Varakantham, P., & Chuin, L. H. (2015). Risk based optimization for improving emergency medical systems. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 702–708.

Schmid, V. (2012). Solving the dynamic ambulance relocation and dispatching problem using approximate dynamic programming. *European Journal of Operational Research*, *219*(3), 611–621.

Schuijbroek, J., Hampshire, R., & 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., Liu, Q., Teo, C.-P., & Wang, I.-L. (2010). Bicycle-sharing system: deployment, utilization and the value of re-distribution. In *Technical Report, NUS. http://bschool.nus.edu/Staff/bizteocp/BS2010.pdf*.

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

Singhvi, D., Singhvi, S., Frazier, P. I., Henderson, S. G., O'Mahony, E., Shmoys, D. B., & Woodard, D. B. (2015). Predicting bike usage for new york city's bike sharing system.. In *AAAI Workshop: Computational Sustainability*.

Singla, A., Santoni, M., Bartók, G., Mukerji, P., Meenen, M., & Krause, A. (2015). Incentivizing users for balancing bike sharing systems. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 723–729.

Sperling, D., & Salon, D. (2002). Transportation in developing countries: An overview of greenhouse gas reduction strategies. In *University of California Transportation Center*.

Sturtevant, N. R., & White, A. M. (2006). Feature construction for reinforcement learning in hearts. In *Proceedings of the International Conference on Computers and Games (CG)*, pp. 122–134.

Sutton, R. S., Precup, D., & Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, *112*(1), 181–211.

Sylvester, J. J. (1857). A question in the geometry of situation. *Quarterly Journal of Pure and Applied Mathematics*, *1*.

Toregas, C., Swain, R., ReVelle, C., & Bergman, L. (1971). The location of emergency service facilities. *Operations Research*, *19*(6), 1363–1373.

Verweij, B., Ahmed, S., Kleywegt, A. J., Nemhauser, G., & Shapiro, A. (2003). The sample average approximation method applied to stochastic routing problems: A computational study. *Computational Optimization and Applications*, *24*(2-3), 289–333.

Vickrey, W. (1961). Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance*, *16*(1), 8–37.

Vulcano, G., Van Ryzin, G., & Ratliff, R. (2012). Estimating primary demand for substitutable products from sales transaction data. *Operations Research*, *60*(2), 313–334.

Wright, L., & Fulton, L. (2005). Climate change mitigation and transport in developing nations. *Transport Reviews*, *25*(6), 691–717.

Yue, Y., & Guestrin, C. (2011). Linear submodular bandits and their application to diversified retrieval. In *Advances in Neural Information Processing Systems*, pp. 2483–2491.

Yue, Y., Marla, L., & Krishnan, R. (2012). An efficient simulation-based approach to ambulance fleet allocation and dynamic redeployment. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 398–405.