TEL AVIV UNIVERSITY

The Iby and Aladar Fleischman Faculty of Engineering

The Zandman-Slaner School of Graduate Studies

Setting Inventory Levels in a Bike Sharing Network

A thesis submitted toward a degree of

Master of Science in Industrial Engineering

by

Sharon Datner

January 2016

TEL AVIV UNIVERSITY

The Iby and Aladar Fleischman Faculty of Engineering

The Zandman-Slaner School of Graduate Studies

Setting Inventory Levels in a Bike Sharing Network

A thesis submitted toward a degree of

Master of Science in Industrial Engineering

by

Sharon Datner

This research was carried out in The

Department of Industrial Engineering

This work was carried out under the supervision of

Dr. Tal Raviv and Prof. Michal Tzur

January 2016

Abstract

Bike Sharing Systems (BSSs) allow customers to rent bicycles at automatic rental stations distributed throughout a city, use them for a short period of time, and return them to any station. One of the major issues that BSS operators must address is non-homogeneous asymmetric demand processes. These demand processes create an inherent imbalance, thus leading to shortages of bicycles, when users are attempting to rent them, and of vacant lockers when users are attempting to return them.

The predominant approach taken by operators to cope with this difficulty is to reposition bicycles to rebalance the inventory levels at the different stations. Most repositioning studies assume that a target inventory level or range of inventory levels is known for each station. In this study, we focus on determining the correct target level for repositioning according to a well-defined objective. This is a challenging task because of the nature of the user behavior that creates the interactions among the inventory levels at different stations. For example, if bicycles are not available at the desired origin of a user's journey, the user may either abandon the system, use other means of transportation, or look for available bicycles at a neighboring station. If in another case, a locker is not available at a user's destination, then that user is obliged to find a station with available space to return the bicycle to the system. Thus, an empty/full station can create a spill-over of demand to nearby stations. In addition, stations are related by origin-destination pairing.

In this study, we take this effect into consideration for the first time when setting target inventory levels and develop a robust guided local search algorithm for that purpose. We show that neglecting the interactions among stations leads to inferior decision-making.

Table of Contents

Acknowledgments **iv**

List of Figures **v**

List of Tables **vi**

1: Introduction **1**

2: Problem Definition **6**

3: Methodology **11**

4: Numerical Study **17**

4.1 Implementation and Experimental Settings 17

4.2 Input Data 18

4.3 Main Results 20

4.4 Robustness of the Algorithm 23

5: Conclusions **25**

References **27**

Appendix A: MILP Based Solution **31**

A.1 Model Formulation 31

A.2 Numerical Study 35

A.3 Discount Factor 38

Appendix B: Markov-Chain Based Solution **39**

B.1 Model Formulation 39

Acknowledgments

I would like to express my gratitude to the following people who helped and encouraged me throughout this work.
I am greatly thankful to my advisors Dr. Tal Raviv and Prof. Michal Tzur for their guidance, open door, and the drive to keep explore and innovate. It was a pleasure working with them, and a great learning experience.
I would also like to thank Mor Kaspi who was always willing to assist and share his knowledge and experience.
I wish to thank my parents for the support and encouragement. Finally, A special thanks goes to my beloved partner Or, for his constant support and advice.

List of Figures

2.1 User Behavior Model 7

2.2 Excess-Time User Behavior Model, adopted from [12] 9

3.1 Inventory Trajectory at a Single Station 12

3.2 Search Algorithm 15

4.1 Renting Attempt Event 18

4.2 Returning Attempt Event 18

B.1 Two Stations Inventory Levels Markov Chain Model 41

List of Tables

4.1 Problem Instances 19
4.2 Main Results 21
4.3 Time-driven Results Minus Occurrence-driven Results 23
4.4 Robustness to the Starting Point 23
4.5 Robustness to the Training Set 24

A.1 Problem Instances 36
A.2 Main Results 36
A.3 Central Planner Effect 37

1 Introduction

Bike Sharing Systems (BSSs) allow customers to rent bicycles at automatic rental stations distributed throughout a city, use them for a short period of time, and then return them to any station. This is an environmentally sustainable mode of transportation and one that can also be integrated with traditional means of public transportation. A significant increase in the number of BSSs and their popularity has recently been seen worldwide ([25]). For a review of the history of BSSs and prospects for their future, see [7] and [24].

One of the major aspects affecting the service quality of BSSs is the availability of bicycles and lockers at the different stations. Developing an inventory model for a BSS involves unique challenges because of the special features of these systems. A BSS experiences two types of demand: a demand for bicycles, by customers who wish to enter the system (renters), and a demand for lockers, by users who have finished their rides and wish to leave the system (returners). Therefore, basic inventory logic that dictates that a higher inventory level can satisfy more customers is not suitable for addressing this problem. Because each station has a constant capacity, a larger quantity of bicycles at a given station implies a smaller quantity of available lockers. Because of the non-homogeneous asymmetric demand processes that typically characterize BSSs, an inherent imbalance is created, leading to shortages both of bicycles when users are attempting to rent them and of vacant lockers when users are attempting to return them.

To prevent such shortage events, several studies have suggested regulation schemes and policies that influence customer demand to the benefit of the system. For instance, several authors have presented pricing mechanisms that give customers incentives to change the origins and destinations of their rides, e.g., [5], [19] and [28]. [12] and [13] proposed a parking reservation policy in which a user reserves a locker at the intended destination station before beginning a ride, thereby diminishing uncertainty and redirecting that user's demand to an available station. A different kind of policy was presented by [10]. Their policy encourages users to choose two destination stations instead of one, and the system then directs them to the station with the greater number of vacant lockers.

In practice, the most common approach taken by operators to cope with the difficulties posed by shortages of bicycles or lockers is to reposition bicycles to rebalance the inventory levels at the different stations. This repositioning is typically performed using a fleet of trucks, each of which carries several bicycles. Two types of repositioning can be distinguished: dynamic repositioning and static repositioning. Dynamic repositioning is performed when the system is active to react to the current system state and unexpected events; see [6], [14] and [18]. Static repositioning occurs during the night, when traffic is low and the BSS is idle. The various models and solution methods proposed to address the static repositioning problem include [16], [3], [4], [21], [8], [1] and [9].

Most static repositioning studies assume that a target inventory level or range of inventory levels is known for each station. Only a few studies have addressed the issue of how to determine these target levels for static repositioning. [16] formulated the problem as a stochastic MIP with the objective of minimizing the cost of the redistribution operation for a required service level. They defined a shortage as a net difference between the total demand over the planning horizon and the total inventory, ignoring the sequence of events occurring in the system. [20] and [23] presented a Markov-chain-based model in which the inventory level is tracked continuously throughout the day. Renters who arrive at an empty station and returners who arrive at a full station are assumed to abandon the system and are considered to be lost sales. [20] suggested a user dissatisfaction function that measures the performance of a station in terms of the expected penalty due to abandonment by returners and renters as a function of the initial inventory at a single station. [23] used dual-bounded service level constraints presented by [16]. Another study that addressed the issue of target inventory levels was conducted by [15], who modeled bike sharing stations as a dual Markovian waiting system and assumed that unsatisfied customers would wait at a station rather than abandoning the system. All of these studies considered models based on a single station, meaning that each station's inventory target level was calculated independently of the others and the interactions among stations were neglected. In [27], the inventory levels of all stations were set simultaneously, but these authors also ignored the influence of interaction on the system because they treated shortages of bicycles and lockers as lost sales. They determined the stations' inventory levels so as to minimize the total expected operation costs of the system due to relocation while satisfying a given level of service.

The interactions among the inventory levels at different stations are an inherent attribute of a BSS. When a customer arrives at an empty station (or when she observes this status online), she can choose between searching for an available bicycle at a neighboring station or abandoning the system to use other means of transportation. Thus, an empty station can create a spill-over of demand to nearby stations. In addition, if the customer decides not to use the system, a future demand for a locker at the destination station is eliminated. Such interactions occur between stations that are not located close to one another. Moreover, when a customer finishes a ride and wishes to return her bicycle, she may arrive at a full station and then be *obliged* to find an available space at another station nearby, meaning that a full station will always create locker demand at neighboring stations. In accordance with this concept, [22] presented different count demand models for BSSs and demonstrated that full/empty stations have an influence on neighboring stations' demand.

The concept of roaming between stations, creates a situation very similar to lateral transshipments. Lateral transshipments within an inventory system are stock movements between locations of the same echelon. These transshipments can be conducted periodically at predetermined points in time to proactively redistribute stock, or they can be used reactively as a method of meeting demand which cannot be satisfied from stock on hand ([17]). The second type of transshipment is the one relevant to our problem.

[17] review the literature about transshipments. They categorize it using several characteristics. Using their classification we can better define which kind of transshipment is best suited to describe our problem. First we note that we have one product (bicycle) and N retailers (stations). The order timing is periodic since the static repositioning occurs usually every night. The type of transshipment is

reactive, where a transshipment occurs when one of the stations faces a stock-out while another has available bicycles. The pooling is partial, since users can choose between roaming to a station with an available bicycle and abandoning the system. Shortages are considered lost sale, due to the fact that if a customer did not roam to a another station (and by that made a transshipment) she abandons the system. Finally, while the decisions regarding transshipments are decentralized, the inventory replenishment policy is centralized, since the operator decides on the inventory levels at all stations. Literature regarding these properties includes: [11], [2] and [26].

Despite these similarities, transshipments deal with traditional inventory problems, rather than the unique dual inventory problem that arises in vehicle sharing systems, as described earlier. Another major difference is related to the length of a period. While in inventory models transshipments occur only once within a period (i.e., once in between two replenishment events), in BSS two replenishment periods are typically an entire day, so that many transshipment events may occur within it, including transshipments in opposite directions. In this study we combine the network's transshipment decisions made by its customers with the special aspects of the inventory problem in vehicle sharing systems.

This study is the first to consider the interactions among stations in BSS decision-making. Our contributions are as follows: First, we present a formal definition of the BSS inventory problem with station interactions (BSIP-SI). Second, we develop a guided local search algorithm to set the initial inventory level at each station (the target level). This search uses a simulation model in which a user behavior model is implemented that includes the roaming between stations that occurs upon a shortage of bicycles or lockers. Third, we use real data to test our algorithm and compare our results with the common practice of operators and with the results of the model presented in [20], which ignores these interactions. We show that our algorithm results in a better quality of service for all of the different instances tested. Our results indicate that the interactions among stations' inventory levels cannot be neglected. Specifically, they have an impact on the desired target inventory levels.

The remainder of the thesis is organized as follows: in Chapter 2, we define the problem, including the user behavior model and related assumptions. In Chapter

3, we characterize the influence of the initial inventory on the system performance and develop our guided local search algorithm accordingly. Chapter 4 presents the numerical study performed, the properties of the data used, the results and an analysis of the robustness of the search algorithm. Chapter 5 presents a discussion and summary of the results. In addition, there are two appendices describing different solution approaches we previously investigated. Appendix A describes a Mixed Integer Linear Programming model, where a central planner dictates the users' choices within the BBS, in addition to determining the initial inventory levels. Appendix B presents a two station model, based on a Markov Chain model, extending the solution method presented at [20].

2 Problem Definition

In this section, we provide a formal definition of the bike sharing system inventory problem with station interactions (BSIP-SI). We start with a broad and general definition of the problem. Then, we illustrate some of the more abstract ideas through a more specific formulation that will be used in our numerical experiment in Section 4.

An instance of the problem is defined by the following:

- A set of bike sharing stations - Each station is characterized by its capacity, i.e., the number of lockers/docking poles.

- A general stochastic demand process for desired rides for each origin-destination pair - The origins and destinations are assumed to coincide with the geographic locations of the stations. The process is defined for a finite planning horizon (typically a working day) and may be non-homogeneous in time and space.

- A journey dissatisfaction function (JDF) with respect to the user. This function maps any combination of a desired ride and a corresponding actual journey to a non-negative value. The ideal journey from station A to B is always the one that proceeds via a direct bicycle ride from A to B, and therefore, the JDF for this scenario is zero by definition. Otherwise, for example, if the user could not find a bicycle at the desired origin and decided to abandon the system or roam to a neighboring station, the JDF returns a larger value that represents the dissatisfaction or dis-utility of the user arising from this occurrence. In our numerical study, we address a special case of the JDF, namely, *excess time*, as will be described later.

- A user behavior model. This model characterizes the choices made by the users, particularly when there are no bicycles at the desired origin station (referred to as a shortage) or when there are no vacant lockers at the desired destination (referred to as a surplus). In general, the user behavior model can be viewed as a decision model that maps a user action to each origin-destination pair and state of the system. The decisions may include waiting for some amount of time at the origin or destination, roaming to a nearby station before renting a bicycle and/or to return it, or abandoning the system and using other modes of transportation. The state of the system at each moment is described by the number of bicycles and (equivalently) the number of available lockers at each station. It is safe to assume that users will strive to minimize their JDF. The general user behavior model is depicted in Figure 2.1.

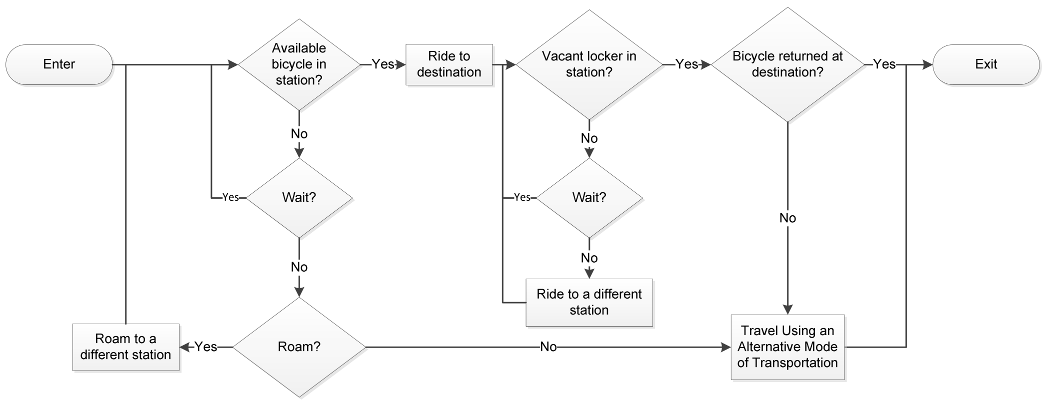


Figure 2.1: User Behavior Model

Given this input, the BSIP-SI is defined as follows: Set the initial inventory levels of the stations to minimize the total JDF of all journeys over a given planning horizon, typically one day. This problem definition is sufficiently general to capture many assumptions about the preferences and behavior of the users and operators. The use of a given planning horizon is motivated by the fact that in many systems, most of the repositioning work is performed during the night with the intent of preparing the system for the next day. Another underlying assumption of the above problem definition is that the total number of bicycles in the system is not a binding constraint. Although this assumption may not reflect the true situation in certain systems, we note that the cost of a bicycle is relatively low compared with other infrastructural and operational costs of the system. Thus, in a well-run BSS, an adequate number of spare bicycles should be available at the operators' disposal at any time.

One example of a JDF, which we consider in the numerical study presented in this thesis, is the JDF introduced by [12], i.e., the excess time. The excess time of a journey is defined as the difference between the actual time taken and the time of an ideal ride. The actual time of a ride may include waiting and roaming before and after riding, whereas the ideal time corresponds to a direct bicycle ride between the origin and destination stations. In other words, the excess time reflects any unnecessary time that the user was obliged to spend to complete her journey. This definition of the JDF clearly satisfies the requirement that a value of zero is assigned to ideal itineraries. In addition, it has the virtue of reflecting the extent of the negative implications of each failure in providing the desired service. Operators should take these implications into consideration when setting the inventory levels at stations.

We also adopt the corresponding user behavior model of [12], which is consistent with the excess-time JDF. This user behavior model assumes that each user is independently striving to minimize her own excess time. It also assumes that the users have full information about the state of the system but that they are myopic, that is, at decision points, they do not account for the implications of possible changes in the system state while roaming between neighboring stations in search of available bicycles or vacant docking poles. Moreover, upon renting, they optimistically assume that a vacant docking pole will be available for them at the time of their arrival at the destination.

The following notation is necessary to implement the user behavior model described above:
C_i - Number of lockers at station i, i.e., its capacity
T_{ij} - Travel time by bicycle from station i to station j
W_{ij} - Walking time from station i to station j
$B_i(t)$ - Number of bicycles at station i at time t
Note that $B_i(t)$ is a state variable, unlike the other quantities, which are data parameters.

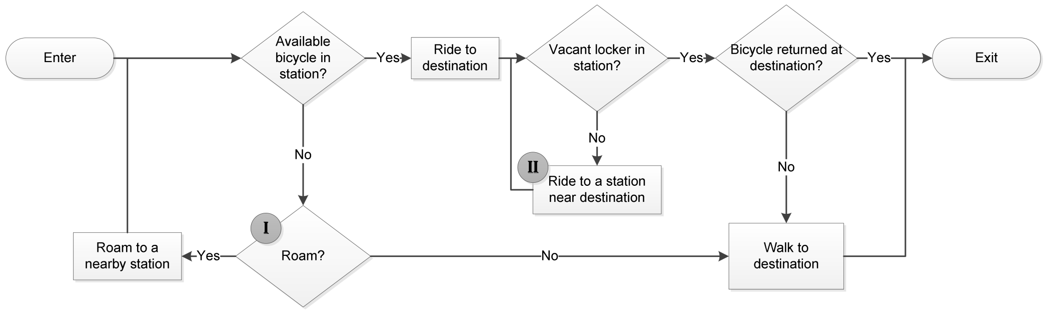


Figure 2.2: Excess-Time User Behavior Model, adopted from [12]

The model, as depicted in Figure 2.2, dictates that a user who does not find an available bicycle may choose to roam to a nearby station or walk directly to her destination (I). The user will prefer to rent a bicycle if the total time required for the journey when that option is chosen is shorter than the time required to walk to the destination. The total journey time includes the walking time to a non-empty nearby station (at time t) and the riding time from that station to the destination. Here, $k^* = \arg\min_{k:B_k(t)>0}(W_{ik} + T_{kj})$ is the non-empty station to which the user can roam that will result in the shortest total journey time. If $W_{ik^*} + T_{k^*j} < W_{ij}$, then roaming to station k^* is better than walking to the destination and the user will therefore choose to do so; otherwise, she will walk directly to her destination.

Once a bicycle is rented, the user rides to her destination. If, upon arrival at the destination, she finds an available locker, she returns the bicycle there and leaves the system. Otherwise, the user rides to a nearby station with an available locker (at time t), leaves the bicycle there and walks back to the original destination. The station is chosen in a similar manner: $k^* = \arg\min_{k:B_k(t)<C_k}(T_{jk} + W_{kj})$. If by the time the user arrives at station k^*, say at time t', it appears to be full, a new return station k^{**} is selected such that $k^{**} = \arg\min_{k:B_k(t')<C_k}(T_{k^*k^{**}} + W_{k^{**}j})$. This process is repeated until a vacant locker is found. However, because the availability of vacant lockers is confirmed before the user starts toward the alternative return station, it is most likely that a vacant locker will be found on the first attempt.

The JDF and user behavior model described above abstract out certain considerations of users and operators in BSSs. In particular, other sources of user dissatisfaction due to shortages may exist in addition to excess time. However, these models are sufficiently rich to capture the complex structure of the interactions among stations and thus are useful for setting stations inventory levels. We note that any other JDF that is monotonic and non-decreasing in its occurrences of shortage and surplus events, along with a user behavior model that is consistent with it, can be incorporated into the search algorithm introduced in the next section. For example, one may assume a user behavior model that allows for waiting at the destination station (until a locker becomes available) or using other modes of transportation in addition to walking and cycling. In such cases, the JDF should reflect the dis-utility associated with these actions. It may include considerations of the uncertainty regarding the total travel time associated with waiting or of the cost of using other modes of transportation.

3 Methodology

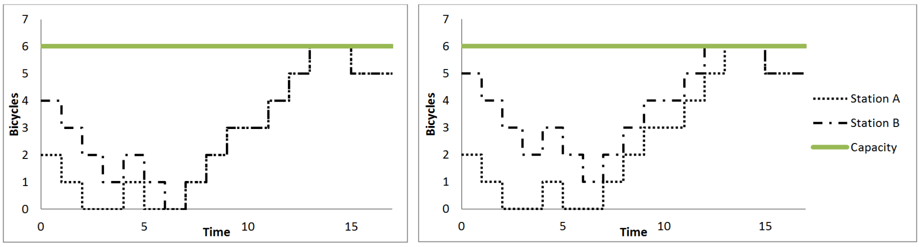
Before presenting our algorithm for setting initial inventory levels, we derive a useful property of the inventory dynamics in a single bike sharing station.

Proposition 1. *For a given demand realization at a station, consider the sequence of shortage and surplus events. Let $n \geq 0$ be the number of shortage events that occur before any surplus event. Then, increasing the initial inventory by l bicycles will result in the elimination of at most $\min(n,l)$ shortage events.*

Proof. Let $B_A(0)$ be the initial inventory at a given station A at the beginning of the planning horizon. Let B be an alternative station facing the same demand realization, with an initial inventory of $B_B(0) = B_A(0) + l$. Consider first the case in which $l \leq n$: After each of the first l shortage events at station A or surplus events at station B (or any combination of l such events), the difference $B_B(t) - B_A(t)$ is decreased by one. Therefore, after at most l shortage events at station A, $B_A(t) = B_B(t)$, and from this time onward, the inventory levels of the two stations coincide. In other words, there are at most l shortage events that can be eliminated by increasing the initial inventory of a station by l bicycles.

We illustrate the above argument using the example presented in Figure 3.1a. In this example, $B_A(0) = 2$ and $B_B(0) = 4$, that is, $l = B_B(0) - B_A(0) = 2$. At station A, there are two shortage events (at times 4 and 6) before the first surplus event. With each of these two shortage events, the difference between the two stations is decreased by one, until the station inventories coincide after the second shortage event.

Similarly, consider the case in which $l > n$: After each of the first n shortage events at station A or surplus events at station B (or any combination of n such



(a) $l \leq n$ **(b)** $l > n$

Figure 3.1: Inventory Trajectory at a Single Station

events), the difference $B_B(t) - B_A(t)$ is decreased by one. Therefore, after at most n shortage events at station A, $B_B(t) - B_A(t) \leq l - n$. Afterward, the inventory levels of the two stations coincide when station A becomes full, which occurs sometime before the first surplus event at station A. In other words, there are at most n shortage events that can be eliminated by increasing the initial inventory of a station by l bicycles; see Figure 3.1b. □

A similar property also applies for surplus events.

Proposition 2. *For a given demand realization at a station, consider the sequence of surplus and shortage events. Let* $n \geq 0$ *be the number of surplus events that occur before any shortage event. Then, decreasing the initial inventory by* l *bicycles will result in the elimination of at most* $\min(n, l)$ *surplus events.*

The proof of Proposition 2 is a mirror image of the proof of Proposition 1 and is thus omitted. An important conclusion that can be drawn from these propositions is that at a station that suffers both surplus and shortage events, only the type of event that occurs first can be mitigated by changing the initial inventory level; for example, if the first unmet demand is for a bicycle (i.e., a shortage), then by changing the initial inventory level, we can only prevent shortages and cannot affect any surpluses that occur afterward. We use this observation in the design of our guided local search algorithm by increasing or decreasing the initial inventory levels in accordance with the first type of event observed. We note that Proposition 1 and Proposition 2 are valid only under the assumption of a fixed demand realization at a station. In reality, any shortage or surplus event at a station affects the demand faced by other stations and may result in a chain reaction throughout the system. This complexity is addressed by our proposed algorithm as described at the end of this section.

We introduce a guided local search algorithm that strives to minimize the total JDF over the planning horizon by setting appropriate initial inventory levels. Our algorithm considers a fixed set of demand realizations, each representing a possible instance of the planning horizon. It searches for the initial inventory levels that minimize the average total JDF over these realizations as an approximation of the expected JDF.

The search is performed iteratively, starting from some initial solution to the problem, i.e., an initial inventory $B_i(0)$ for each station i. In each search iteration, the algorithm estimates the expected total JDF by simulating the system based on a user behavior model, given a set of demand realizations and a vector of initial inventory levels. Information on the occurrences of shortage and surplus events is collected during the simulation. Based on this information, the inventory levels are updated, typically at numerous stations simultaneously. The process is then repeated until some stopping criterion is met. The core of the search algorithm is the mechanism that updates the initial inventory levels of the stations at the end of each iteration.

Guided by Propositions 1 and 2, the information we collect focuses on the first shortage or surplus event at each station. We define the following categories of scenarios:

1. The first shortage event occurs before any surplus event.

2. The first surplus event occurs before any shortage event.

3. No shortage or surplus occurs, but $B_i(t) = 0$ for some t, i.e., the station is empty at some point.

4. No shortage or surplus occurs, but $B_i(t) = C_i$ for some t, i.e., the station is full at some point.

Note that Categories 3 and 4 are not disjoint. In each iteration, we count the number of demand realizations that belong to each category (M_1, M_2, M_3, M_4) for each station. We use these values to determine at which stations a change in the inventory level by one unit may be beneficial, i.e., we apply Propositions 1 and 2 with $l = 1$. An increase in the inventory level could be beneficial at a station where there are more realizations with shortages (Category 1) than surpluses (Category 2). In addition, we must consider the realizations in which there are no shortages or surpluses but the inventory level $B_i(t)$ reaches the station's capacity, that is, the station is full at some point (Category 4). Increasing the inventory level in Category 4 cases will result in a surplus, as in Category 2 cases. Accordingly, we add a bicycle to each station for which $M_1 > M_2 + M_4$. Using the same logic, we remove a bicycle from each station for which $M_2 > M_1 + M_3$. Note that each station can satisfy at most one of the conditions considered above. If a station does not satisfy any of these conditions, this means that its inventory level never reaches either of its boundaries, and therefore, its initial inventory level remains unchanged.

The process is repeated, using the same set of demand realizations, until a solution that was previously considered is encountered. We could stop the search at this point, considering that as a result of its deterministic nature, the algorithm would simply repeat its cycle from this point onward. However, as long as the algorithm's stopping criterion is not met, we instead continue by perturbing the best found inventory levels and continuing from that point. This perturbation also provides some protection against premature convergence to a local minimum. We apply the perturbation by adding a uniform discrete random variable $U[-2,2]$ to the initial inventory level at each station. If this modification results in a solution that exceeds the range $0,...,C_i$ for station i, then the corresponding value is truncated accordingly. Finally, the algorithm stops when a predetermined time budget or number of iterations is reached. A summary of the search stages is illustrated in Figure 3.2.

We refer to the search described thus far as an *occurrence-driven search*. The purpose of this occurrence-driven search is to reduce the number of shortage and surplus occurrences, which is typically consistent with the objective of minimizing any JDF. However, two arbitrary events will not necessarily have the same impact on a JDF. Therefore, it is desirable to devise a search algorithm that prioritizes the elimination of events that will result in a greater effect on the chosen

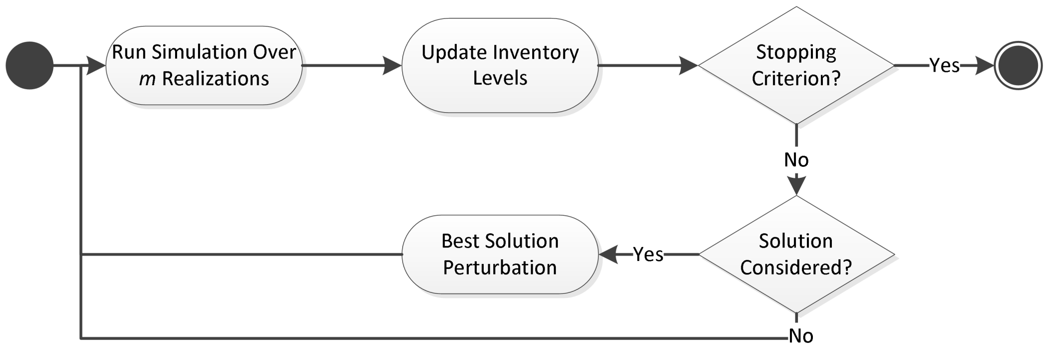


Figure 3.2: Search Algorithm

JDF.

Therefore, we introduce a *time-driven search* that is specially tailored for the excess-time JDF. We use an approach similar to that presented above but with an emphasis on the time that users must spend in the system as a result of each avoidable shortage or surplus. Using the same previously described scenario categories, instead of counting the number of realizations, we sum over the avoidable excess time. Let M'_1 be the sum of the excess times due to the first shortage event in all realizations of Category 1. This is calculated by, for each such realization, determining the station to which the user roams, $k^* = \arg\min_{k:B_k(t)>0}(W_{ik} + T_{kj})$, and then recording the difference between the resulting journey time with roaming and the ideal time, i.e., $\min(W_{ik^*} + T_{k^*j}, W_{ij}) - T_{ij}$. Similarly, M'_2 is the sum of the excess times due to the first surplus event in all realizations in which a surplus event occurs first. This is calculated as $\min_{k:B_k(t)<C_k}(T_{jk} + W_{kj})$.

M'_3 is calculated in the same way as M'_1 for realizations of Category 3. M'_3 is updated at the first time at which the station becomes empty. This represents an evaluation of the excess time that would have been added if the initial inventory level had been reduced by one. Similarly, M'_4 is calculated in the same way as M'_2 for realizations of Category 4. This represents an evaluation of the excess time that would have been added if the initial inventory level had been increased by one.

Based on the values of $M'_1, \ldots, M'_4$, we update the inventory levels of the stations in the same manner used in the occurrence-driven search: we add a bicycle to each station for which $M'_1 > M'_2 + M'_4$ and remove a bicycle from each station for which $M_2' > M_1' + M_3'$. The iterations of the search process and the stopping criterion remain the same.

The search algorithm uses a simulation model (described in Section 4.1) that implements the user behavior model using a discrete event simulation architecture. It simulates the system's inventory levels and customers' movement over the planning horizon, given certain initial inventory levels. Different inventory levels can lead to different user decisions, which then lead to different dynamics of the inventory levels, and so on. In this way, the simulation captures the interactions among stations.

4 Numerical Study

In this section, we present a numerical study conducted using the proposed algorithm and its results. Section 4.1 presents the search settings and implementation details. Section 4.2 describes the data used in the study. Section 4.3 reports our results, and Section 4.4 analyzes the robustness of the algorithm.

4.1 Implementation and Experimental Settings

The user behavior model was implemented in a simulation that reproduced two main types of events: renting attempt events (Figure 4.1) and returning attempt events (Figure 4.2). In a renting attempt event, a user arrives at a station. If a bicycle is available, a new returning attempt event is created and added to the event queue. Otherwise, based on the logic of the user behavior model, the user either leaves the system or roams to another station. In the latter case, a new renting attempt event is created. In a returning attempt event, a user arrives at a station with a bicycle. If a locker is available, the user leaves the system. Otherwise, the user roams to another station and a new returning attempt event is created.

The two search algorithms and the simulation were coded using MathWorks MATLAB R2011b. The experiments were run on an Intel Xeon X3450 @ 2.67 GHz with 16 GB of RAM. Each of the two search methods was run using three different starting points, i.e., sets of initial inventory levels: (i) Random - a random inventory level at each station; (ii) Half - a starting inventory level at each station equal to half of that station's capacity, a heuristic used both in the literature and in industry; and (iii) R&K - a starting inventory level at each station set using the single-station model suggested by [20]. The stopping criterion was set to 100 iterations. In each iteration, the value of the current solution was evaluated

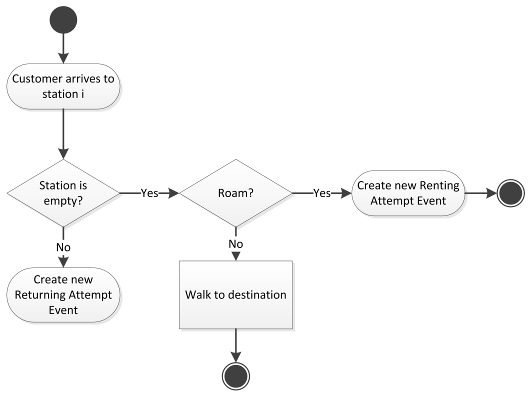


Figure 4.1: Renting Attempt Event

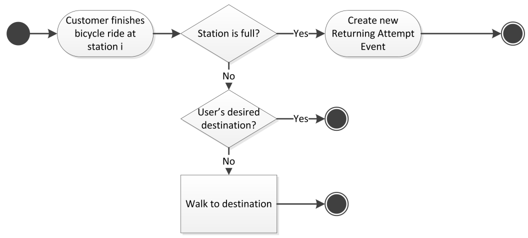


Figure 4.2: Returning Attempt Event

using 50 demand realizations generated based on demand processes obtained as described in Section 4.2. The quality of the solutions was evaluated using a *test set* of 500 additional realizations generated based on the same demand processes.

4.2 Input Data

We used data from three BSSs of different sizes, all of which are located in the United States: Hubway in Boston, Capital Bikeshare in Washington, D.C., and Divvy in Chicago. The network topologies of and detailed trip data for these systems are available on their websites. The problem was solved for a planning horizon of 9.5 hours starting at 7:00 am on a working day, assuming without loss of generality that dynamic repositioning would be performed by the end of this planning horizon. For each BSS, we used data from two different months, one working month and one during summer vacation. In this way, we could consider different demand patterns in the same BSS. Several properties of these problem instances are presented in Table A.1.

Table 4.1: Problem Instances

	Hubway		Capital		Divvy	
Number of Stations	131		232		300	
Period	May-13	Aug-13	Apr-13	Jun-13	Aug-13	Oct-13
Avg. Rides per Day	3364	4906	7311	8101	4953	5633
Avg. Rides per Planning Horizon	1823	2493	3536	3800	2505	2964

The demand estimation process was executed as described by [12]. All rent and return transactions were recorded by the operators. After eliminating holidays and weekend trips, we found that the daily demand patterns did not change significantly throughout each period. By aggregating these transactions over multiple days, we estimated the demand rate of renters for each origin-destination pair during each 30-minute period throughout the day. As may be expected, the demand process was not homogeneous over time. For example, the demand for bicycles at stations located near working areas was low at the beginning of the day and increased significantly toward the end of the working day.

We note that in their current state, the information systems cannot document user abandonments. This is primarily because when a user arrives at an empty station, she will not attempt to rent a bicycle, and therefore, no such attempt is registered by the system. To address this issue, we considered the proportion of time for which a station was empty or full and inflated the demand rates accordingly. However, using the transaction data we obtained, we could not distinguish between users who rented or returned bicycles at their desired origins or destinations and those who were obliged to roam to nearby stations. We note that statistical analysis of this phenomenon will be required to obtain more reliable estimations of demand; however, this is out of the scope of this study.

Based on the estimated origin-destination demand rates, we created a *training set* of 50 realizations, which was used as the input to the search algorithm, and a *test set* of 500 realizations, which was used to evaluate the solutions obtained by the algorithm. In this manner, we simulated the real-life situation in which operators set initial inventory levels based on their forecasts (the training set) and then observe the results on future days (the test set). In addition, the search results showed no effect of over-fitting to the realizations in the training set.

Riding and walking times were estimated using the Google Maps API. For regular trips, it is safe to assume that most users will ride directly from their origins to their destinations. This is not the case for round trips, i.e., trips that begin and end at the same station. The riding time for such trips was set to 30 minutes based on the observed average round-trip travel time.

In summary, our complete data set included riding-time and walking-time matrices, an O-D matrix for each 30-minute period of the day, the capacity and location of each station, and demand realizations (i.e., training and test sets) for all six problem instances. These data are available for download from our website at `http://www.eng.tau.ac.il/~talraviv/Publications/`.

4.3 Main Results

In this section, we present our main numerical results. Each problem instance was solved using two search methods (occurrence- and time-driven search) and three starting points. We first note that each of these six solutions consistently outperformed the two alternative solutions with which we compared our results, namely, Half and R&K (note that these solutions should not be confused with the three tested starting points of our search algorithm: Random, Half, and R&K). In Table 4.2, we report the results for the best solution of the six in each case, referred to as our solution, whereas in Section 4.4, we perform a detailed comparison of all solutions.

The first group of rows presents the total excess time per day (in hours) for the three tested solutions. The second group of rows in Table 4.2 shows the percentage reduction in excess time achieved by our solution compared with the other two solutions. The third and fourth groups of rows present the number and percentage (with respect to the total demand) of ideal rides, respectively. The average excess time spent in the system by a user who does not have an ideal ride is presented in the fifth group of rows. The sixth group of rows shows the total number of shortage and surplus occurrences. Note that this number is slightly different from the number of non-ideal rides because the same user may experience one or more shortage and/or surplus events. In the seventh group of rows, the average number of shortage and/or surplus occurrences per non-ideal ride user is presented. The

Table 4.2: Main Results

		Hubway-May	Hubway-Aug	Capital-Apr	Capital-June	Divvy-Aug	Divvy-Oct
Excess Time (h/day)	Half	85.57	140.43	274.45	318.03	36.08	75.50
	R&K	22.15	54.48	50.92	47.67	14.57	19.12
	Our Search	20.53	49.48	46.08	46.69	14.43	17.35
Excess Time Reduction	Vs. Half	76.01%	64.77%	83.21%	85.32%	60.00%	77.02%
	Vs. R&K	7.35%	9.20%	9.51%	2.08%	0.91%	9.23%
Number of Ideal Rides	Half	1321	1745	2140	2269	2201	2371
	R&K	1623	2119	3104	3207	2385	2754
	Our Search	1644	2133	3120	3377	2384	2767
Ideal Ride Ratio	Half	72.5%	70.0%	60.5%	59.7%	87.9%	80.0%
	R&K	89.0%	85.0%	87.8%	84.4%	95.2%	92.9%
	Our Search	90.2%	85.6%	88.2%	88.9%	95.2%	93.4%
Avg. Excess Time per Non-ideal Ride User (min)	Half	10.2	11.3	11.8	12.5	7.1	7.6
	R&K	6.6	8.7	7.1	4.8	7.3	5.5
	Our Search	6.9	8.2	6.6	6.6	7.2	5.3
Number of Shortage and Surplus Events	Half	630	998	2137	2476	334	748
	R&K	218	437	476	689	125	232
	Our Search	197	413	451	459	127	212
Avg. Number of Shortage and Surplus Events per Non-ideal Ride User	Half	1.3	1.3	1.5	1.6	1.1	1.3
	R&K	1.1	1.2	1.1	1.2	1.0	1.1
	Our Search	1.1	1.1	1.1	1.1	1.1	1.1
Avg. Running Time (h)		1.02	1.34	2.06	2.08	1.49	1.69

last row reports the average running time of the search algorithm per problem instance.

We observe that our solution consistently outperforms the other solutions in terms of total excess time. The excess time reduction is significant compared with the Half solution, whereas the reduction compared with the R&K solution varies. The mean excess time, when calculated based on the 500 realizations in the test set, is significantly smaller for our solution, with $p-value < 0.0006$, for all six problem instances. Interestingly, for the same BSS in different months, the percentage reductions in excess time achieved by our solution can be very different, as in the cases of Capital and Divvy. Recall that the R&K solution considers only a single station at a given time, neglecting the interactions among stations. Therefore, in instances in which such interactions are rare because of the balanced nature of the demand process, it is more difficult to affect the total excess time merely by adjusting the initial inventory levels.

Although we advocate the use of the excess-time JDF, we recognize that many other authors and operators use other performance measures, particularly the number of shortage and surplus occurrences. We observe in Table 4.2 that minimizing the total excess time results in reducing the number of such shortage or surplus events in five of the six cases. In one case (Divvy-Aug), the number of these events in our solution is slightly larger compared with that in the R&K solution. Similarly, the number and ratio of ideal rides are typically larger in our solution. Another interesting observation is related to the average excess time spent by users who do not have an ideal ride. In most cases, in addition to reducing the number of users who experience non-ideal rides, the average excess time they spend is also reduced. Furthermore, the average number of shortage and surplus occurrences experienced by a non-ideal ride user is no larger than in the other solutions. In short, our solution results in a higher number of satisfied users, and most unsatisfied users are less discomforted in terms of both the number of shortage and surplus events and their consequent excess time.

Next, let us consider the results of the occurrence- and time-driven search algorithms. In Table 4.3, we compare the solutions that represent the best results (among the three starting points) achieved by each of these algorithms in terms of excess time. The values presented in the table represent the difference between the two solutions, where positive values in the table correspond to higher measures for the time-driven search. The first column shows the names of the problem instances. In the second column, we present the percentage reduction in excess time achieved by the time-driven search minus the corresponding value for the occurrence-driven search. The third column shows the average difference in the number of shortage and surplus events per user between these two solutions. Similarly, in the last column, we present the difference in the ideal ride ratio. Recall that unlike the two previous measures, the ideal ride ratio is a measure that should be maximized; thus, negative values here reflect better results in the time-driven search.

As expected, the time-driven search algorithm is better suited to minimizing the total excess time. Interestingly, the two algorithms yield very similar results in terms of the number of shortage and surplus occurrences and the ideal ride ratio, although the time-driven algorithm demonstrates a slight advantage. We conclude

Table 4.3: Time-driven Results Minus Occurrence-driven Results

	Excess Time Reduction	No. Shortage and Surplus per User	Ideal Ride Ratio
Hubway-May	0.81%	-0.0006	-0.0011
Hubway-Aug	1.09%	0.0040	-0.0004
Capital-Apr	0.44%	-0.0002	-0.0003
Capital-June	0.33%	0.0025	-0.0008
Divvy-Aug	-0.05%	0.0096	-0.0004
Divvy-Oct	1.72%	0.0086	-0.0007

Table 4.4: Robustness to the Starting Point

	Random	Half	R&K
Hubway-May	7.00%	**7.35%**	6.91%
Hubway-Aug	**9.20%**	8.85%	8.80%
Capital-Apr	9.27%	9.21%	**9.51%**
Capital-June	1.67%	1.97%	**2.08%**
Divvy-Aug	0.48%	**0.91%**	0.80%
Divvy-Oct	8.96%	9.11%	**9.23%**

that the excess time may be a good surrogate objective function for various service quality measures. To further investigate the properties of the search method, in the following section, we focus only on the time-driven algorithm.

4.4 Robustness of the Algorithm

In this section, we consider the effects of different starting points and different training sets on the performance of the time-driven algorithm. In Table 4.4, we show how the search is affected by the different starting points (i.e., Random, Half and R&K). For each problem instance and starting point, the table presents the excess time improvement compared with the R&K solution (as in the fifth row of Table 4.2).

The most important observation to be drawn from Table 4.4 is that our search algorithm is not highly sensitive to its starting point, which is advantageous. Recall that the three starting points that we used were a randomly generated vector, a vector representing half of the capacity at each station and the solution obtained using the R&K method. Each of these starting points could itself represent a solution to the problem; among them, Random is typically the worst and R&K is the best in terms of excess time. Interestingly, the table shows that a better starting point does not necessarily lead to a better solution. In fact, the R&K starting point led to the best final result in only half of the problem instances. Clearly, if sufficient computational resources are available, some improvement may be gained by running the algorithm with multiple starting points, including various random vectors.

Next, let us examine the sensitivity of the algorithm to the specific training set of 50 realizations that was used as the input to the search algorithm. We created three more such sets based on the same demand processes and ran the search again using the R&K starting point. The solutions were evaluated using the same test set of 500 realizations as was the solution of the original search. The results are displayed in Table 4.5. The first column provides the names of the problem instances. The second column gives the excess time improvement (over R&K) of the original training set's solution for the R&K starting point. The remainder of the columns show the improvement rates achieved using the three other training sets with the same starting point. It is evident that the search algorithm is fairly robust. In nine of the eighteen runs, the improvement achieved using the newly generated training sets was equal to or larger than the original. Therefore, there is no reason to suspect that the search was over-fitted to the original training set.

Table 4.5: Robustness to the Training Set

	Original	1	2	3
Hubway-May	6.91%	6.81%	6.91%	6.81%
Hubway-Aug	8.80%	8.70%	8.65%	8.75%
Capital-Apr	9.51%	7.58%	9.01%	9.30%
Capital-June	2.08%	1.59%	2.42%	2.10%
Divvy-Aug	0.80%	0.80%	1.12%	1.20%
Divvy-Oct	9.23%	9.66%	10.26%	10.68%

5 Conclusions

In this thesis, we introduced the problem of setting the initial inventory levels in a BSS with station interactions and developed a simulation-based guided local search algorithm that optimizes the quality of service. Our algorithm is novel in the sense that it extracts information from the dynamics observed in the simulation. We proved that only the first shortage or surplus event at each station in each demand realization can be eliminated by changing the initial inventory level at that station by one unit. We used this property to guide our search procedure. The algorithm is capable of capturing and considering complex interactions in the system that originate from the behavior of the users. Such complexities could not be addressed without the use of simulation. The effectiveness of our algorithm was demonstrated using actual demand data from three real BSSs.

In our model, it is assumed that the goal of the operator when setting the initial inventory levels is to minimize the JDF, which is equivalent to maximizing the quality of service. A legitimate criticism of this modeling assumption is that the operator may have other objectives, such as minimizing his operational cost and, in particular, the cost of repositioning bicycles between stations. Moreover, it is not always possible to satisfy the inventory levels prescribed by our model. This can be the case, for example, because of the capacity and time constraints of the repositioning operation. Therefore, it is important to also explore values of solutions in the neighborhood of the solution obtained by our algorithm. Such an investigation is out of the scope of the current study and will be an important topic for future research.

Our numerical study shows that the interactions among stations should not be neglected when planning the inventory levels of BSS stations, as done by previous authors, e.g., [20]. We note that in any transportation system, and particularly in a BSS, each user is selfishly attempting to minimize her own dissatisfaction by selecting the best possible itinerary. If a central planner could assign an itinerary to each user, the total JDF could be reduced much further, although certain users might be worse off. In a preliminary stage of this study, we formulated the central planning problem of determining the initial inventory levels and the itineraries of all users as a variant of a network flow problem on a graph induced by a space-time diagram with multiple scenarios. We could solve moderately sized instances of this problem using a commercial solver. However, when we used the inventory levels prescribed by this model in combination with the simulation and user behavior model described in this study, we found that the resulting excess time and number of shortage and surplus events were not competitive with our results or even with the R&K solution. This finding can be attributed to the gap between the itineraries that would be selected by a central planner and those selected by the users themselves.

The discussion above is relevant to various decisions regarding the design and operation of BSSs, e.g., repositioning operations and the locations and capacities of stations. Future research should consider the behavior of users and interactions among stations when devising models for these problems. For example, when operators are considering the trade-off between setting up many small stations or fewer stations with greater capacity, the corresponding problem cannot be correctly solved without considering that users can roam between stations.

References

[1] Panagiotis Angeloudis, Jun Hu, and Michael GH Bell. A strategic repositioning algorithm for bicycle-sharing schemes. *Transportmetrica A: Transport Science*, 10(8):759–774, 2014.

[2] TW Archibald. Modelling replenishment and transshipment decisions in periodic review multilocation inventory systems. *Journal of the Operational Research Society*, 58(7):948–956, 2007.

[3] Mike Benchimol, Pascal Benchimol, Benoît Chappert, Arnaud De La Taille, Fabien Laroche, Frédéric Meunier, and Ludovic Robinet. Balancing the stations of a self service bike hire system. *RAIRO-Operations Research*, 45 (01):37–61, 2011.

[4] Daniel Chemla, Frédéric Meunier, and Roberto Wolfler Calvo. Bike sharing systems: Solving the static rebalancing problem. *Discrete Optimization*, 10 (2):120–146, 2013.

[5] Daniel Chemla, Frédéric Meunier, Thomas Pradeau, Roberto Wolfler Calvo, and Houssame Yahiaoui. Self-service bike sharing systems: simulation, repositioning, pricing. 2013.

[6] Claudio Contardo, Catherine Morency, and Louis-Martin Rousseau. *Balancing a dynamic public bike-sharing system*, volume 4. CIRRELT, 2012.

[7] Paul DeMaio. Bike-sharing: History, impacts, models of provision, and future. *Journal of Public Transportation*, 12(4):3, 2009.

[8] Güneş Erdoğan, Gilbert Laporte, and Roberto Wolfler Calvo. The static bicycle relocation problem with demand intervals. *European Journal of Operational Research*, 238(2):451–457, 2014.

[9] Iris A Forma, Tal Raviv, and Michal Tzur. A 3-step math heuristic for the static repositioning problem in bike-sharing systems. *Transportation research part B: methodological*, 71:230–247, 2015.

[10] Christine Fricker and Nicolas Gast. Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity. *EURO Journal on Transportation and Logistics*, pages 1–31, 2014.

[11] Yale T Herer and Michal Tzur. Optimal and heuristic algorithms for the multi-location dynamic transshipment problem with fixed transshipment costs. *IIE Transactions*, 35(5):419–432, 2003.

[12] Mor Kaspi, Tal Raviv, and Michal Tzur. Parking reservation policies in one-way vehicle sharing systems. *Transportation Research Part B: Methodological*, 62:35–50, 2014.

[13] Mor Kaspi, Tal Raviv, Michal Tzur, and Hila Galili. Regulating vehicle sharing systems through parking reservation policies: Analysis and performance bounds. *European Journal of Operational Research*, page forthcoming, 2016.

[14] Christian Kloimüllner, Petrina Papazek, Bin Hu, and Günther R Raidl. Balancing bicycle sharing systems: An approach for the dynamic case. In *Evolutionary Computation in Combinatorial Optimisation*, pages 73–84. Springer, 2014.

[15] Fabien Leurent. Modelling a vehicle-sharing station as a dual waiting system: stochastic framework and stationary analysis. 2012.

[16] Rahul Nair and Elise Miller-Hooks. Fleet management for vehicle sharing operations. *Transportation Science*, 45(4):524–540, 2011.

[17] Colin Paterson, Gudrun Kiesmüller, Ruud Teunter, and Kevin Glazebrook. Inventory models with lateral transshipments: A review. *European Journal of Operational Research*, 210(2):125–136, 2011.

[18] Dana Pessach, Tal Raviv, and Michal Tzur. Dynamic repositioning in a bike sharing system. Technical report, Working Paper, Tel-Aviv University, 2014.

[19] Julius Pfrommer, Joseph Warrington, Georg Schildbach, and Manfred Morari. Dynamic vehicle redistribution and online price incentives in shared mobility systems. *Intelligent Transportation Systems, IEEE Transactions on*, 15(4):1567–1578, 2014.

[20] Tal Raviv and Ofer Kolka. Optimal inventory management of a bike-sharing station. *IIE Transactions*, 45(10):1077–1093, 2013.

[21] Tal Raviv, Michal Tzur, and Iris A Forma. Static repositioning in a bike-sharing system: models and solution approaches. *EURO Journal on Transportation and Logistics*, 2(3):187–229, 2013.

[22] Christian Rudloff and Bettina Lackner. Modeling demand for bicycle sharing systems–neighboring stations as a source for demand and a reason for structural breaks. In *Transportation Research Board Annual Meeting*, 2013.

[23] Jasper Schuijbroek, Robert Hampshire, and Willem-Jan van Hoeve. Inventory rebalancing and vehicle routing in bike sharing systems. 2013.

[24] Susan Shaheen and Stacey Guzman. Worldwide bikesharing. *Access Magazine*, 1(39), 2011.

[25] Susan Shaheen, Stacey Guzman, and Hua Zhang. Bikesharing in europe, the americas, and asia: past, present, and future. *Transportation Research Record: Journal of the Transportation Research Board*, (2143):159–167, 2010.

[26] Greys Sošic. Transshipment of inventories among retailers: Myopic vs. farsighted stability. *Management science*, 52(10):1493–1508, 2006.

[27] Patrick Vogel, Bruno A Neumann Saavedra, and Dirk C Mattfeld. A hybrid metaheuristic to solve the resource allocation problem in bike sharing systems. In *Hybrid Metaheuristics*, pages 16–29. Springer, 2014.

[28] Ariel Waserhole and Vincent Jost. Pricing in vehicle sharing systems: Optimization in queuing networks with product forms. *EURO Journal on Transportation and Logistics*, pages 1–28, 2014.

A MILP Based Solution

At the beginning of the research, we tried a different solution approach, namely formulating the problem as a Mixed Integer Linear Programming (MILP), and solving it with a commercial solver. The decision variables, objective function and evaluation method were the same as in the rest of the study, namely: setting the initial inventory levels so as to minimize the total excess time, and evaluate the solution using the simulation model described in the body of the thesis. The major relaxation made to the problem with this approach is that the users no longer independently choose their own actions in the system. Instead there is a central planner who sets both the initial inventory level at each station and the users *actual* route.

A.1 Model Formulation

Consider a bike sharing system over a set of demand realizations of a finite horizon $[0,T]$. We use time discretization into small periods, where at each period in each station there can be at most one user who wishes to rent or return a bicycle. For convenience we refer to the beginning of period t as time t. Each user has a desired origin and destination. However, a user can roam between different stations in the system, therefore she has many different possible routes to fulfill her journey, including abandoning the system and traveling by foot. Since we assume users wish to minimize the time they spend in the system, we include in the model only a predetermined number of routes with the shortest travel time between the desired origin and destination, in addition to walking to the destination without renting a bicycle at all. The objective of the model is to minimize the total excess time of users in the system, by setting the initial inventory levels. Using 50 realizations (as a "training set") the model returns the best initial inventory levels vector, i.e., the initial inventory levels of all realizations are constrained to be the same.

Input

S Set of stations, indexed by $i = 1, \ldots, |S|$

C_i Number of lockers installed at station $i \in S$, referred to as the station's capacity

R Set of realizations, indexed by $r = 1, \ldots, |R|$

Q Set of users, indexed by $q = 1, \ldots, |Q|$, each user is represented by a tuple $< i, j, t, r >$ where the user wants to rent a bicycle from station i at time t and ride to station j in realization r.

W Number of possible routes for each user, including a single possibility of making the journey by foot.

TR_{qw} Travel time of user q in her w^{th} route .

U_{qw} A tuple representing the riding part of the wth route of user q. The tuple includes $< k, m, s, e >$, where the user takes bicycle from station k at time s to station m at time e.

Decision Variables

R_{qw} Equals 1 if user q takes her w^{th}.route.

B_{itr} Number of bicycles that arrive to station i at time t in realization r.

L_{itr} Number of bicycles taken from station i at time t in realization r.

I_{itr} Number of bicycles at station i at time t in realization r.

I_i^0 Number of bicycles at station i at time 0 in all realizations.

A_{qit} Set of route indices of user q that end their bicycle ride at station i at time t.

G_{qit} Set of route indices of user q that begin their bicycle ride at station i at time t.

$$\min \quad \sum_{q \in Q} \sum_{w=1}^{W} TR_{qw} R_{qw} \tag{A.1}$$

s.t.

$$I_{itr-1} + B_{itr} - L_{itr} = I_{itr} \qquad \forall i \in S, \forall t \in \{1,\dots,T\}, \forall r \in R \tag{A.2}$$

$$I_{itr} \le C_i \qquad \forall i \in S, \forall t \in \{1,\dots,T\}, \forall r \in R \tag{A.3}$$

$$I_i^0 = I_{i0r} \qquad \forall i \in S, \forall r \in R \tag{A.4}$$

$$\sum_{w=1}^{W} R_{qw} = 1 \qquad \forall q \in Q \tag{A.5}$$

$$\sum_{q \in Q} \sum_{w \in A_{qit}} R_{qw} = B_{itr} \qquad \forall i \in S, \forall t \in \{1,\dots,T\}, \forall r \in R \tag{A.6}$$

$$\sum_{q \in Q} \sum_{w \in G_{qit}} R_{qw} = L_{itr} \qquad \forall i \in S, \forall t \in \{1,\dots,T\}, \forall r \in R \tag{A.7}$$

$$I_{itr}, B_{itr}, L_{itr} \ge 0 \text{ integer} \qquad \forall i \in S, \forall t \in \{1,\dots,T\}, \forall r \in R \tag{A.8}$$

$$R_{qw} \in \{0,1\} \qquad \forall q \in Q, \forall w \in 1,\dots,W \tag{A.9}$$

The objective function A.1 minimizes the total time users spend in the system. It is calculated as the sum of traveling times of the chosen routes. The total excess time is the difference between the total time and the ideal time of the desired journeys. Since the ideal time for each journey is a constant, minimizing the total time is the same as minimizing the total excess time, and we can drop the constant from the objective function. Constraints A.2 are inventory-balance constraints, keeping the inventory levels correct for each station, in each realization, at each time period. Constraints A.3 insure that the inventory at each station is bounded by its capacity, in each realization, at each time period. Constraints A.4 state that the initial inventory level at each station is equal at all realizations. Constraints A.5 state that each user is assigned to exactly one route. Constraints A.6 and A.7 stipulate that for every actual journey made, there will be bicycle arriving or taken at the relevant stations at the relevant time.

To further clarify constraints A.6 and A.7, we defined the sets A_{qit} and G_{qit}. In order to track the number of bicycles at each station, we need to map between each route to the inventory change it inflicts. Each route defines the station where the bicycle is taken from and the station where the bicycle is returned to. The set A_{qit} includes the route indices of user q that *end* their bicycle ride at station i at time t. Meaning if $w \in A_{qit}$ for a given i,t, then this route create a bicycle return at station i at time t. We define it in the following manner: $A_{qit} = \{w \in 1,\ldots,W | U_{qw}.m = i \text{ and } U_{qw}.e = t\}$. In a similar manner we define the set G_{qit} to be the route indices of user q that *begin* their bicycle ride at station i at time t. Summing the binary variable R_{qw} over these sets, will include only the rides that were chosen for each user. By minimizing the total time users spend in the system in all the different realizations, while setting the same initial inventory levels for all realizations, that is, the value of the decision variables I_i^0, we retrieve a solution to our problem.

A.2 Numerical Study

In this section, we present results that were obtained when solving instances of practical size with the MILP formulations introduced in the previous section (Section A.1. The goal of the numerical study is to check whether the model succeeded in finding good solutions. In order to measure that, we compare our results with the results given by the method of R&K ([20]).

We took as a case study the bike sharing system in Washington DC, USA, Capital Bikeshare. As of June 2013 the system consisted of 232 stations. Weekday rent transactions were collected over few months. The demand estimation was done as described previously in the thesis. Based on this data set we created a training set of 50 benchmark problems that were used as input to the model presented in Section A.1, and a test set of 50 benchmark problems that were used to test the results it provided. The latter test was preformed in the same manner done in the rest of the thesis, i.e. via simulation model. In order to test the model on different instances of different size, we used the geographical conditions of the Capital Bikeshare system, and created several subset instances. Each instance was created from a distinct geographical region, where most of the rides started and ended inside it. The details of each instance are displayed in Table A.1.

All the experiments were ran on an Intel Xeon X3450 @ 2.67 GHz with 16 GB of RAM. The MILP model was implemented for the above instances using IBM-

Table A.1: Problem Instances

	Crystal	Arlington	Capital
Number of Stations	15	30	232
Avg. Rides per Day	130	255	7820

Ilog CPLEX 12.6, with CPLEX's default settings. In order to solve the model, the integer constraints were relaxed, allowing a user to rent a fraction of a bicycle. The results are displayed in Table A.2.

Table A.2: Main Results

		Crystal	Arlington	Capital
Excess Time (h/day)	R&K	0.59	**6.13**	13.82
	MILP Model	**0.57**	6.37	**13.45**
Excess Time Reduction		4.04%	-3.92%	2.68%
Number of Ideal Rides	R&K	**122**	**238**	**7138**
	MILP Model	121	237	6369
Ideal Rides Rate	R&K	94.0%	93.4%	91.3%
	MILP Model	93.8%	93.0%	81.5%

Each row in the table presents a different performance measure that was collected in the experiment for the three problem instances. The values presented in the table are estimated means that were obtained by running the 50 realizations of our test data set. The first group of rows presents the total excess time per day (in hours) for the two tested solutions: R&K (described in the body of this thesis) and the one obtained by our MILP model. The second group of rows shows the percentage of excess time reduction obtained by our MILP model compared with the R&K starting inventory levels. The third group of rows presents the number of ideal rides for each of the two solutions. The percentage of such ideal rides out of the total demand, including users who abandoned the system, is presented last.

It can be observed that unfortunately, the results are ambiguous. For the instances of Crystal and Capital the MILP model achieves better results, and managed to reduce the total and the excess time users spend in the system. But for the Arlington instances the MILP model did not achieve better results. In order to understand the reason for the results, we checked the impact of the major relaxation of the model, the central planner. A central planner can create cases in which even though the user is able to make her ideal ride, the model chooses for her a different route. That happens when it helps reducing the total time that all users spend in the system, even though it adds some time to that specific user. The problem with these cases is that in real life (as implemented in our evaluation simulation model), the user would choose the best ride possible for her. Thus the inventory levels that were set based on the central planner decisions would not fit, and could result in a larger excess time throughout the system.

Table A.3: Central Planner Effect

	Crystal	Arlington
Number of Ideal Rides	121.28	242.22
Number of Non-Ideal Rides	7.9	13.5
Non Reasonable Rides	6.78	12.16
Non Reasonable Rides Ratio	86.0%	89.9%

In Table A.3 we can see the results of the decisions made by the central planner in the MILP model. All the results are averages over 50 days. Each column includes the results of a different data instance. The first row depicts the amount of ideal rides chosen by the model. The second row presents the number of non-ideal rides. The third row includes the number of non-reasonable rides, meaning a non-ideal ride that was assigned to a user when she could have chosen a shorter route. Therefore at the evaluation stage (as in real life), the user would take her preferred route and would create a different inventory level than the one considered in the MILP model. The fourth row presents the non-reasonable rides ratio out of the non-ideal rides.

We can see that though most of the rides the model assigns are ideal, the routes that are assigned differently are mainly not what the user would choose in real life. Therefore those few users, who decide differently from the model, encounter different inventory levels, that can then create different decision making for future users. This way a chain of events can happen that may prevent from other users to achieve their ideal ride, thus hurting the quality of the solution.

A.3 Discount Factor

The model developed did not achieve conclusive results due to the relaxation of the problem. In order to try to overcome this problem, a revised model was tested. The new model included a discount factor over the route travel time. In this way, more weight was given to early rides over later rides. It was done in order to better simulate real life situations, where a user who got to the system earlier will make her ideal ride if possible, regardless of the effect it has on future users. The revised model managed to better simulate the users' choices, but the overall outcome was not better than the original model.

We suspect the reason for that is due to the fact that the discount factor affected the objective function as well. That caused events in the future to have less of an influence on the objective function, though there could be cases where the lion share of the excess time happened towards the end of the planning horizon. For instance, there could be a station where for most days there are surpluses in the evening, but only a couple of days with shortages in the morning. The revised model will give those couple of surpluses in the morning more weight, and will decide to probably reduce bicycles from the initial inventory level. This decision would result in an inferior solution.

Both the original and the revised model were unable to encompass the complexity of the bike sharing system, and did not produce conclusive improvement. For that reason we presented in this study a simulation based approach, which can better include the vast complexities of the system, and produce better results. We present this method as a service for other researchers who may consider a similar approach to this problem.

B Markov-Chain Based Solution

Another solution approach we investigated for our problem was using a Markov-Chain model. The decision variables, objective function and evaluation method were the same as in the rest of the study, namely: setting the initial inventory levels so as to minimize the total excess time, and evaluate the solution using the simulation model described in the body of the thesis. We developed a Markov-Chain model for the inventory levels of two neighboring stations, extending the R&K model presented in [20] for one station. In this solution approach, the interactions between stations were implemented by assuming that when one of the stations is empty/full, the customer will roam to the neighboring station, if it is possible. This means modeling the interactions between two stations as if they were relatively isolated from the rest of the BSS. We used the model by dividing the stations in the BSS to pairs and single stations, and applying the relevant model (R&K for single stations and the current method for pairs of stations) for each of them.

B.1 Model Formulation

We consider two neighboring bike-sharing stations over a finite horizon $[0,T]$ with the following settings: at time 0 the inventory level (number of bicycles) in each station is set. During the planning horizon, users that wish to rent or return bicycles arrive at the stations according to a non-homogeneous Poisson demand process with rate $\mu_k(t)$ and $\lambda_k(t)$, respectively, at station k , $k = 1,2$. The state of the system is described by two factors: i_1 the inventory level at station 1 and i_2 the inventory level at station 2. The notation (i_1,i_2) will be used to denote the state of the system. The Markov-Chain includes four events, the arrival of a bicycle to station 1 or 2 (returning attempt), and the departure of a bicycle from station 1 or 2 (renting attempt).

When both stations are not full nor empty, the transition between the different states happens in the following manner: Giving the state (i_1, i_2), the arrival of a bicycle to station 1 changes the system state to (i_1+1, i_2), and the arrival to station 2 leads to state (i_1, i_2+1). In a similar manner, the departure of a bicycle from station 1 changes the system state to (i_1-1, i_2), and the departure from station 2 results in state (i_1, i_2-1).

When one of the stations is empty/full, we assume the customer will roam to the neighboring station, if it is possible. Therefore when one of the stations is empty, the inventory level at the neighboring station is depleted at a higher rate, meeting both stations demands. That is, when the system state is $(0, i_2), \forall i_2 > 0$ the transition to state $(0, i_2-1)$ is at $\mu_1(t) + \mu_2(t)$ rate. The same happens for the transition from state $(i_1, 0), \forall i_1 > 0$ to $(i_1-1, 0)$. In the same way when one of the stations is full, the inventory level at the neighboring station is increasing at a $\lambda_1(t) + \lambda_2(t)$ rate. This model is depicted in Figure B.1.

Since repositioning is done periodically, we are interested in the transient dynamics of the process rather than in its steady state. Let $\pi_{ij}(t)$ denote the probability of the station being at state j at time t given that its initial state at time 0 was i .We use the notation $\pi(t)$ to refer to the whole transition probability matrix.

The objective function is to minimize the two stations' user dissatisfaction function (TS-UDF). TS-UDF is the expected penalty of the two stations due to shortage events of bicycles and lockers as a function of the initial inventory levels. In order to define the TS-UDF we first identify four sources of user dissatisfaction in the system according to this model, and the system is penalized for each of them:

p_1 Penalty charged for each potential user who abandons due to a shortage of bicycles (referred to as shortage) at both stations.

p_2 Penalty charged for each user who cannot return her rented bicycle at any of the two stations, due to a shortage of lockers (referred to as surplus), and has to roam to other further stations.

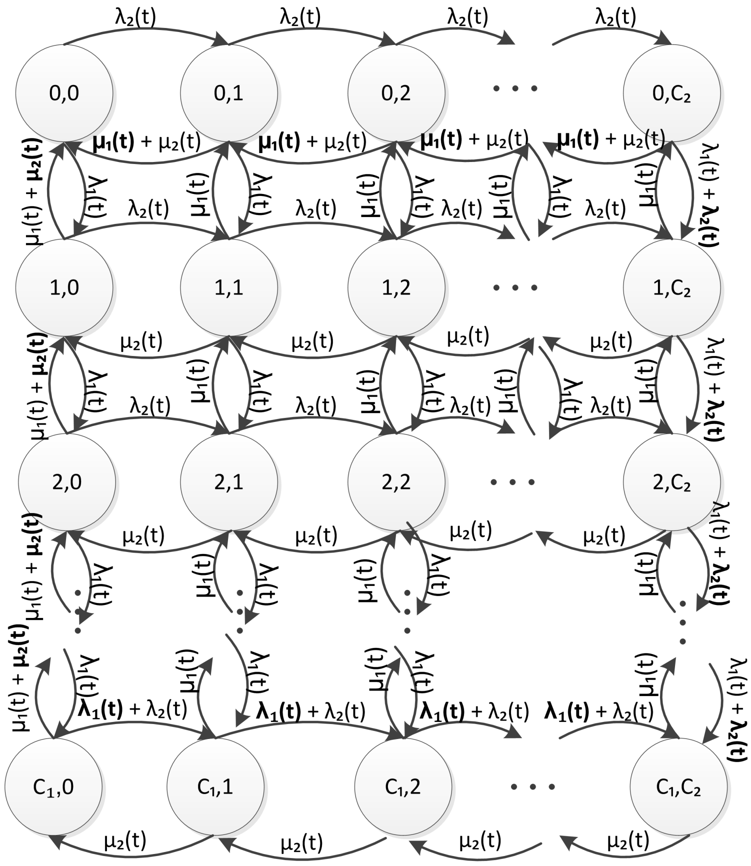


Figure B.1: Two Stations Inventory Levels Markov Chain Model

p_3 Penalty charged for each user who cannot rent a bicycle at her origin station and roams to the other station.

p_4 Penalty charged for each user who cannot return her rented bicycle at her destination station and roams to the other station.

Furthermore we use C_i for the station's capacity, i.e. the number of lockers at the station and T the end of the planning horizon. The decision variables are I_0^1, I_0^2, they represent the initial inventory level of each station at the beginning of the planning horizon (at time 0). The TS-UDF is:

$$
\begin{aligned}
F(I_0^1, I_0^2) = \int_0^T \Big\{ & \pi_{(I_0^1,I_0^2),(0,0)}(t)(\mu_1(t) + \mu_2(t))p_1 \\
& + \pi_{(I_0^1,I_0^2),(C_1,C_2)}(t)(\lambda_1(t) + \lambda_2(t))p_2 \\
& + \big[\sum_{k=1}^{C_2} \pi_{(I_0^1,I_0^2),(0,k)}(t)\mu_1(t) + \sum_{k=1}^{C_1} \pi_{(I_0^1,I_0^2),(k,0)}(t)\mu_2(t)\big] p_3 \\
& + \big[\sum_{k=0}^{C_2-1} \pi_{(I_0^1,I_0^2),(C_1,k)}(t)\lambda_1(t) + \sum_{k=0}^{C_1-1} \pi_{(I_0^1,I_0^2),(k,C_2)}(t)\lambda_2(t)\big] p_4 \Big\} dt
\end{aligned}
$$

The first part of the TS-UDF includes the probability that both stations will be empty given the initial inventory levels, i.e. $\pi_{(I_0^1,I_0^2),(0,0)}(t)$. When both stations are empty the rate of unmet demand will be the sum of both stations demand rate $(\mu_1(t) + \mu_2(t))$, and therefore it incurs the penalty p_1. Similarly the second part deals with the case where both stations are full. The third part of the definition deals with the roaming costs in case one of the stations is empty, and there is at least one bicycle at the other station (its inventory level is between one and its capacity). The roaming occurs at the same demand rate for the original station, that is $\mu_1 or \mu_2$. In a similar way, at the last part we handle the case where one station is full and there is at least one locker available at the other station.

Next, we used the procedure that was shown in [20] to estimate these equations using a discretization of the Markov chains. Using real BSS data we applied the TS-UDF on a whole system. It was done by pairing stations in the system by a decision rule related to their proximity. The results were then evaluated using the simulation and user behavior model presented in this thesis. Unfortunately the results were not consistently better than the R&K solution, due to the many interactions neglected by focusing only on the interactions between paired neighboring stations.

תקציר

מערכות שיתוף אופניים מאפשרות ללקוחות לשכור אופניים בתחנות השכרה אוטומטיות המפוזרות ברחבי העיר, להשתמש בהם לפרק זמן קצר ולהחזיר אותם בכל תחנה. אחת הסוגיות המשמעותיות שאיתה נאלצים מפעילי מערכות שיתוף אופניים להתמודד היא תהליכי ביקוש אסימטריים ולא הומוגניים בזמן. תהליכים אלו מייצרים חוסר איזון אינהרנטי שמוביל לאירועי חוסר של אופניים, כאשר משתמש מנסה לשכור אופניים, ולאירועי חוסר של לוקרים כאשר משתמש מנסה להחזיר אופניים ששכר.

הגישה העיקרית בה נוקטים מפעילי מערכות שיתוף אופניים בכדי להתמודד עם הקושי הזה, היא שינוע אופניים בכדי לאזן מחדש את רמות המלאי בתחנות השונות. רוב המחקרים העוסקים בשינוע האופניים מניחים שקיימת רמת מלאי או טווח של רמות מלאי שאליהן יש לשאוף להגיע בעת שינוע האופניים. בעבודה זו אנו מתמקדים בקביעת אותה רמת מלאי, בהתאם לפונקציית מטרה מוגדרת היטב. זוהי משימה מאתגרת מאחר וההתנהגות הטבעית של המשתמשים במערכת מייצרת יחסי גומלין בין רמות המלאי של התחנות השונות. לדוגמה, כאשר אין אופניים זמינים בתחנת המקור המבוקשת על ידי המשתמש, הוא עשוי לנטוש את המערכת, אולי באמצעות שימוש באמצעי תחבורה אלטרנטיבים, או לפנות לתחנה שכנה ולחפש בה אופניים זמינים. לעומת זאת, במקרה בו אין לוקר פנוי בתחנת היעד, המשתמש מחויב למצוא תחנה בה ישנו לוקר פנוי על מנת להחזיר את האופניים למערכת. לכן, תחנה ריקה\מלאה יכולה לייצר שינוי בביקוש לתחנות שכנות, ובנוסף לכך גם לתחנות שקשורות אליה באמצעות קשר של מקור-יעד.

בעבודה זו לראשונה, אנחנו מתייחסים ליחסי הגומלין בין התחנות בעת קביעת רמות המלאי הדרושות, ומפתחים חיפוש מקומי מונחה רובסטי בהתחשבות ביחסים אלו. אנחנו מראים שהתעלמות מיחסי הגומלין בין התחנות בעת קביעת רמות המלאי מובילה למספר רב יותר של חוסרים באופניים ובלוקרים.

אוניברסיטת תל אביב

הפקולטה להנדסה ע"ש איבי ואלדר פליישמן

בית הספר לתארים מתקדמים ע"ש זנדמן סליינר

קביעת רמות מלאי התחלתיות

במערכות לשיתוף אופניים

חיבור זה הוגש כעבודת גמר לקראת התואר "מוסמך אוניברסיטה"

בהנדסת תעשייה

על ידי

שרון דטנר

העבודה נעשתה במחלקה להנדסת תעשייה

בהנחיית דר' טל רביב ופרופ' מיכל צור

טבת תשע"ו

דצמבר 2015

אוניברסיטת תל אביב

הפקולטה להנדסה ע"ש איבי ואלדר פליישמן

בית הספר לתארים מתקדמים ע"ש זנדמן סליינר

קביעת רמות מלאי התחלתיות

במערכות לשיתוף אופניים

חיבור זה הוגש כעבודת גמר לקראת התואר "מוסמך אוניברסיטה"

בהנדסת תעשייה

על ידי

שרון דטנר

טבת תשע"ו

דצמבר 2015