Submitted to *Interfaces*
manuscript (Please, provide the mansucript number!)

Analytics and Bikes: Riding Tandem with Motivate to Improve Mobility

Daniel Freund
Marketplace Labs, Lyft Inc., New York, New York, dfreund@lyft.com

Shane G. Henderson
School of Operations Research and Information Engineering, Cornell University, Ithaca, New York 14850, sgh9@cornell.edu

Eoin O'Mahony
Uber Technologies Inc., San Francisco California, eoin@uber.com

David B. Shmoys
School of Operations Research & Information Engineering and Department of Computer Science, Cornell University, Ithaca, New York 14850, dbs10@cornell.edu

Bike-sharing systems are now ubiquitous across the United States. We have worked with Motivate, the operator of the systems in, for example, New York, Chicago, and San Francisco, to innovate a data-driven approach to managing both their day-to-day operations and to provide insight on several central issues in the design of its systems. This work required the development of a number of new optimization models, characterizing their mathematical structure, and using this insight in designing algorithms to solve them. Here, we focus on two particularly high-impact projects, an initiative to improve the allocation of docks to stations, and the creation of an incentive scheme to crowdsource rebalancing. Both of these projects have been fully implemented to improve the performance of Motivate's systems across the country; for example, the Bike Angels program in New York City yields a system-wide improvement comparable to that obtained through Motivate's traditional rebalancing efforts, at far less financial and environmental cost.

Key words: Transportation, Optimization, Inventory, Sharing Economy

Introduction

Bike-sharing systems have transformed the urban landscape throughout the world. In the United States, the largest such systems are operated by our industry partner, Motivate. These include Citi Bike in New York, Divvy in Chicago, and Ford GoBike in the Bay Area. In 2017, users in these three bike-sharing systems accounted for more than 70% of the 35 million trips facilitated by bike-sharing systems across the country (National Association of City Transportation Officials 2018). In contrast, there were only 320,000 trips in bike-sharing systems nation-wide in 2010.

Our work is focused on station-based bike-sharing systems, in which there are a specified number of finite-capacity stations. The capacity of the station is given by the number of docks at it. Each dock can either be full or empty, depending on whether or not it contains a bike. Whereas the availability of full docks allows riders to rent a bike from a station, the availability of empty docks allows bikes to be returned. Because systems allow riders to rent a bike at any station (that has a bike available) within the system and return it at any other (that has an empty dock), the availability of empty and full docks are both necessary for the system to work successfully. The spatial flexibility of renting and returning bikes anywhere in the service area makes bike-sharing a sustainable transportation alternative for tourists and commuters alike. Yet, to successfully provide this alternative to commuters, bike-sharing systems need to handle the asymmetry of "tidal" commuter flows: in the morning, riders overwhelmingly rent bikes in residential areas (causing stockouts due to stations having no bikes) and return bikes in commercial areas (causing stockouts at stations at which all docks are full). Operators alleviate the effects of the tidal commuter flows through rebalancing; that is, they typically deploy box trucks or vans to move bikes from full to empty stations. This form of motorized rebalancing constitutes a large operational cost, and our work with Motivate has aimed at reducing the need for it.

Contribution

Below, we quote Jay Walder, the former CEO of Motivate.

> *"The work with Cornell was, in my mind, really about giving us an analytical foundation to be able to make some of the decisions that were most critical to what we are doing."* (Motivate 2018).

We describe here how our collaboration with Motivate helped the company become an industry leader in the use of analytics. Our collaboration started originally on the tactical level, aimed at the development of tools that improve the operational efficiency of existing rebalancing methods. Later on, we also introduced new ideas to Motivate's strategic vision for tackling imbalance. We focus here on two of these projects aimed at reducing the need for motorized rebalancing.

In the first project, we aimed at reallocating capacity among stations; in other words, we tried to identify opportunities in moving docks from stations where they were underutilized, to stations where additional capacity would improve service quality for riders. Our work set up an optimization problem and identified sufficient mathematical structure to develop computationally efficient algorithms for its solution. The second project set up an incentive program, prominently advertised as *Bike Angels*, to crowdsource rebalancing through the user base. Rather than trying to help with rebalancing, both projects helped Motivate change the system design in a way that reduces the need for rebalancing in the first place. They have been fully implemented, and Motivate's systems, and its users, benefit from these advances. Both of these strategic projects, as well as the earlier tactical ones, relied on the same data-driven approach. See Freund (2018) for a full account of these innovations.

On a scientific level, our work has combined a wide array of different OR tools; it involved innovations in stochastic modeling, in data analysis, in developing discrete optimization formulations (built on the stochastic models), and in designing algorithms capable of efficiently solving them. This paper summarizes these contributions and details their impact.

Our work with Motivate builds on the idea of user dissatisfaction functions (UDFs), first defined by Raviv and Kolka (2013). Given a planning period, UDFs map the number of bikes and docks at a station, i.e., its capacity and its inventory level, to the expected number of stockouts during the planning period. Computing this expectation requires a stochastic model of demand that captures the arrival of users with (or without) bikes wanting to return (or rent) bikes at a station and it requires data analysis to fit the parameters of the model. When stations are empty or full, demand is censored, since the operator does not observe realized stockouts. In our work, we developed a decensoring method that estimates time-dependent demand for arrivals and returns at each location (O'Mahony and Shmoys 2015). Further, we developed a novel, numerically stable method (O'Mahony 2015) to compute UDFs, based on the so-called Poisson equation for continuous-time Markov chains.

For the first project, to (re)allocate dock capacity, we design a nonlinear integer program (IP) that treats the inventory level and the capacity at each station as variables and optimizes the allocation of capacity and inventory across the entire system. For this IP, we derived discrete convexity properties, especially multimodularity as we discuss below, using sample-path arguments, that then allowed us to develop an efficient algorithm to solve it. Our algorithm not only solves the IP to optimality, but also advanced our understanding of constrained optimization for discrete convex problems (Freund et al. 2018b). Our algorithm works as follows: We start from an initial assignment of docks to stations, and first identify the optimal allocation of bikes to stations through a simple integer linear program (for which the linear relaxation is guaranteed to have an optimal solution that is integer).

Then, in each iteration, we determine the reallocation of at most one bike and at most one dock that most improves the objective function. This continues until there is no move that improves the objective. The proof of correctness relies on two elements: first showing that for any realization of demand, the UDF at each station has a property known as multimodularity, and second, showing that this structure is sufficient to guarantee that the algorithm always finds an optimal solution to the system-wide allocation problem. Motivate completed a pilot in NYC to reallocate dock capacity. The flexibility inherent in the integer programming formulation allowed Motivate to specify a restricted set of moves compatible with the physical space constraints of the station locations, and subject to this refinement, the docks moved in the pilot were the ones specified by the optimal solution computed by our tools. Based on usage data collected after the pilot, we designed a UDF-based method to estimate the impact of reallocated dock capacity. The estimated impact of the pilot, along with our analytical tools, have led Motivate to adopt the principle of moving dock capacity, with over 200 docks moved so far in its systems nationwide. Though these moves were also informed by solutions to the IP, other considerations (e.g., agreement from the associated Department of Transportation) also played a role. Our methodology is now used by Motivate in analyses run twice a year to identify further potential moves and it also informs the allocation of new capacity added to the system (Press Office New York City Hall 2018).

For the second project, to design an incentive scheme, the goal is to reward customers for rides that drive the system towards desirable configurations, that is, incentivizing customers to align their rides with desirable outcomes. But which rides should be incentivized? Again, UDFs play a central role. We explain how UDFs are employed, and discuss the marginal benefits of making the incentives dynamic in time, in response to real-time demand, in contrast to static policies that set them in advance.

Related Work

Since bikesharing became popular in the early 2010s, the OR community has embraced it as a new and interesting application of study. We refer the reader to extensive literature reviews in Freund et al. (2018a), Freund (2018), and de Chardon et al. (2016). Most important for our work are the contributions of Raviv and Kolka (2013), Schuijbroek et al. (2017), and Parikh and Ukkusuri (2014). Preliminary versions of our own contributions described here have previously appeared as O'Mahony and Shmoys (2015), O'Mahony et al. (2016), Chung et al. (2018), and Freund et al. (2018b), where a brief announcement of the last is given as an extended abstract in Caro and Tang (2018); the extended abstract is similar to our presentation of the capacity reallocation project here.

Imbalance, Rebalancing, and User Dissatisfaction Functions

> *"You're better than this @DivvyBikes. Two stations completely full in West Town/ Fulton Market. First time I've had to seek out a third"* (Ross 2017).
>
> *"@CitiBikeNYC Why are your guys removing bikes from Madison square park at 2:30? People will need them to get home at 5"* (McCloskey 2015).

The efficient operation of bike-sharing systems involves subtleties with respect to both the modeling and the prediction of demand. This is exemplified by the two tweets above, complaining about opposite rebalancing actions in two different situations: the first complains about *not* picking up bikes at full stations, while the second complains about the opposite. Depending on the exact nature of demand at a station, it might or might not be optimal to leave a station full rather than remove bikes through rebalancing operations. In systems with hundreds of stations, the demand patterns of which change over time, distinguishing between these scenarios (and quantifying the difference) necessarily requires data-driven support systems.

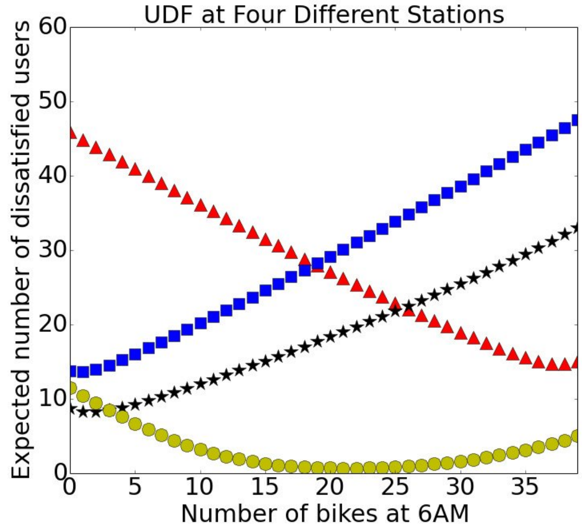
Estimating Unsatisfied Demand via User Dissatisfaction Functions

UDFs underlie almost all of our work with Motivate. These functions provide an inventory model that maps, for each station and any planning horizon, that station's capacity (i.e., the maximum number of bikes that can be present at that station) and an initial number of bikes at that station at the start of the planning horizon, to the expected number of stockouts over the course of the horizon. A planning horizon can be an arbitrary time interval; in our context it usually spans either a full day (e.g., from 6 am to midnight), when allocating capacity, or 9-15 hours from the present point in time (when planning rebalancing). Each UDF is based on a stochastic process from which we sample a sequence of customers arriving to rent or return bikes at that station over the given planning horizon. Then, for every user in the sampled sequence who arrives to rent a bike, the number of bikes (the inventory) decreases by one; for every user who arrives to return a bike, it increases by one. However, when no bikes are available and the next user in the sampled sequence is one who attempts to rent a bike, then we record a dissatisfied user; we similarly record a dissatisfied user when the number of bikes is equal to the station's capacity and the next user in the sequence attempts to return a bike. In both cases, we assume that the dissatisfied customer leaves without a rental/return, so the number of bikes remains at zero in the first case and at the station's capacity in the second case. The UDF is then defined as the expected number of dissatisfied users over the course of the planning horizon. The expectation is taken over the sampled sequences (for a fixed sequence, the number of dissatisfied users is deterministic); the appendix includes a more formal mathematical definition for how the UDF is computed for a single sampled sequence. Thus, to specify a UDF, we must also give a stochastic model of user behavior with respect to that station and planning horizon.

We model users wishing to rent bikes through independent nonhomogeneous Poisson processes, one for each station. A user's destination station is independently selected from an origin-dependent distribution, and biking times are independent (their distribution depends only on the origin/destination station pair). With the added assumptions that "upstream" stations never run out of bikes and that no trips begin and end at the same station, the splitting and superposition properties of Poisson processes then imply that the arrival process of users returning bikes to a fixed station is also a nonhomogeneous Poisson process that, in addition, is independent of the arrival process of renters at that same station. Moreover, it implies that these processes at different stations are independent. In practice, upstream stations certainly do run out of bikes, but we make this assumption because simulation results (Jian et al. 2016) demonstrate that the resulting prescriptions remain extremely effective even when we relax the assumption and because of the enormous mathematical and computational simplification it yields. In particular, it follows that a city-wide bike-sharing system decomposes by station (i.e., we can compute UDFs for each station in isolation). UDFs can thus be defined for each station and time interval in isolation; they map the initial number of bikes and empty docks at a station to the (expected) number of out-of-stock events over the course of the interval (O'Mahony 2015).

Our choice of objective function, the expected number of out-of-stock events, reflects the subscription-based nature of most large bike-sharing systems, wherein users pay once per year for a subscription and then ride for free throughout the year. Retention of subscribers is then the driving strategic principle behind a successful bike-sharing operation, which explains why we focus on this customer-service focused objective. This objective also aligns with short-term users who pay (e.g., a per-ride fee). In the bike-sharing systems on which we have worked, such day-users contribute a small fraction of rides.

Figure 1 **Sample UDFs at Four Different Stations Show the Optimal Level of Bikes at a Station Can Vary Dramatically.**

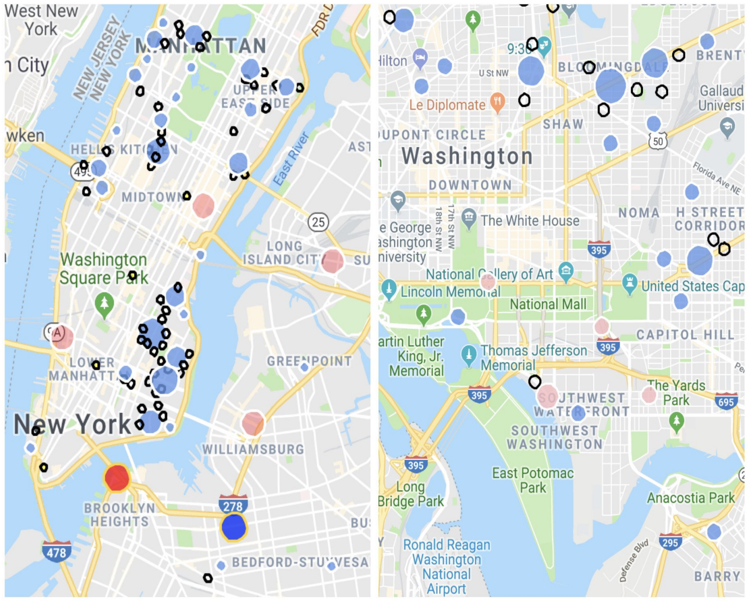


To estimate the various parameters of, for example, the underlying Poisson processes or the transition matrix (of origin-dependent destination distributions), we use a combination of maximum likelihood and specialized decensoring techniques. Some sample UDFs are depicted in Figure 1, where we only show the dependence on the number of bikes, and not the number of docks, for clarity.

Target Levels for Bikes

The plots of four different UDFs portrayed in Figure 1 show four very different functions; yet all of them are convex. This suggests that, in general, the expected number of out-of-stock events may be a convex function of the initial number of bikes at a station. It is intuitive that the expected number of renters not finding a bike should be convex (diminishing marginal returns) and decreasing in the number of bikes available at the outset. It is also intuitive that the expected number of riders unable to return a bike because of full docks is also convex and increasing in the number of bikes available at the outset. Sample-path arguments that employ induction on the sequence of events at a fixed station establish that this intuition is indeed correct. The convexity of the UDFs has

Figure 2 **The Screenshot Shows Older Versions of the Developed Map in NYC and Washington D.C.**



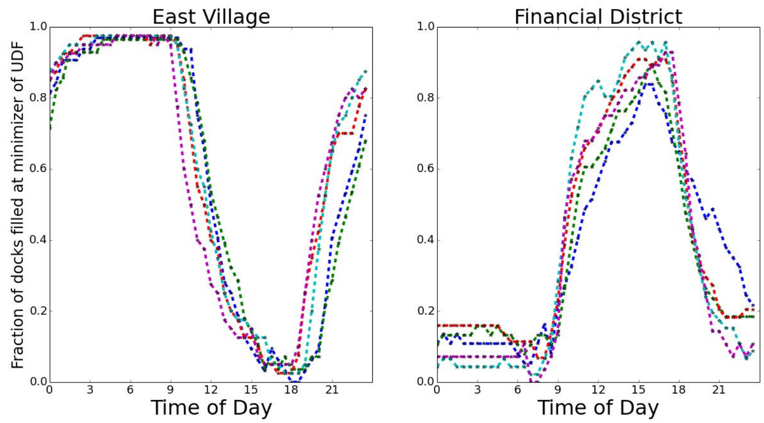
Note. The circles on the map indicate to dispatchers which stations should have bikes added (in blue) and which ones should have bikes removed (in red), with the area of each circle proportional to the recommended number. Map data: ©2018 Google.

significant implications for Motivate's operations. In particular, the unique minimum at each station provides a natural target for rebalancing at a given point in time. Motivate uses these target levels in a decision aid we developed to guide dispatchers over the course of the day (cf. Figure 2). Figure 3 shows how these minimizers vary over time in different neighborhoods; noticeably, stations in the same neighborhood show strong similarities, especially before and at the beginning of each rush hour when the minimizers are clustered either close to empty or close to full, making rebalancing targets conceptually simple. In contrast, towards the end of each rush hour, larger differences occur, and rebalancing relies more heavily on analytics.

System-wide Optimization for Bikes

Beyond using the UDFs for their minimizers at each station, we can also use them to identify the optimal allocation of all bikes system-wide at each point in time. Using the

Figure 3 **The Graphs Show the Optimal Number of Bikes to Position at Stations in Two NYC Regions as a Function of the Time of Day**



Note. Customers on the Upper West Side, a residential area, want bikes in the morning, but empty docks to receive returning bikes in the evening. The reverse is true in Midtown Manhattan.

number of bikes at each station as decision variables, we can formulate this as an optimization problem: constraints dictate that the number of bikes at each station is bounded above by the number of docks at that station, and the objective is the expected number of system-wide stockouts, that is, the sum of the values of the UDF over all stations. See the appendix for the mathematical formulation.

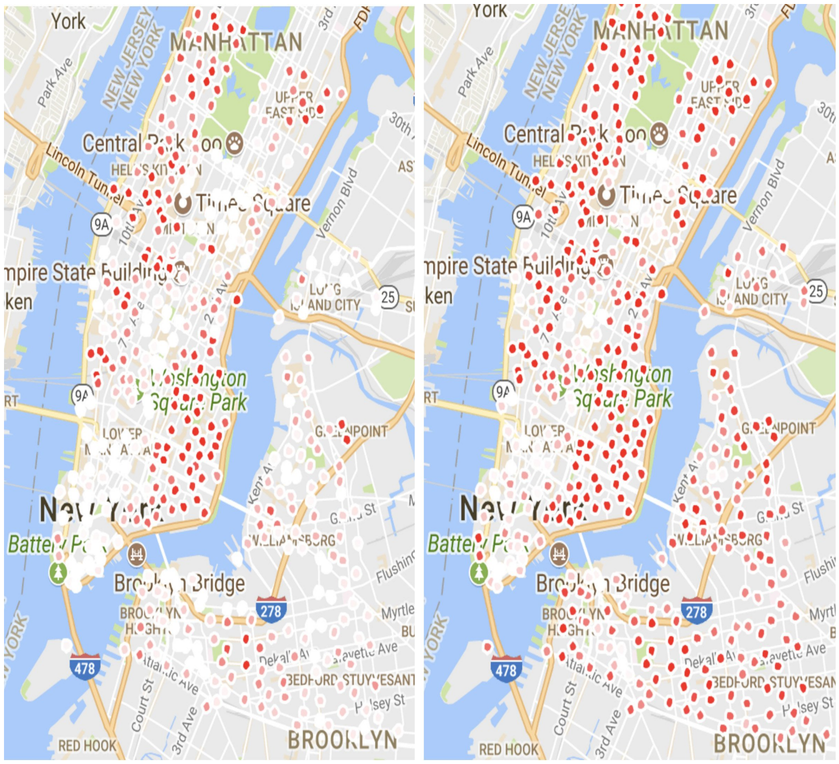
Because the UDFs are convex, we can efficiently solve the resulting IP (see Hochbaum 1994). The optimal solutions are visualized for the morning rush hour, for different values of B, the total number of bikes in the fleet, in Figure 4. The optimal objective function values for different settings of B quantify the cost impact of changing the fleet size (cf. Figure 5).

Allocating Capacity

Below, we quote Emily Gates, the former Director of Operational Strategy at Motivate.

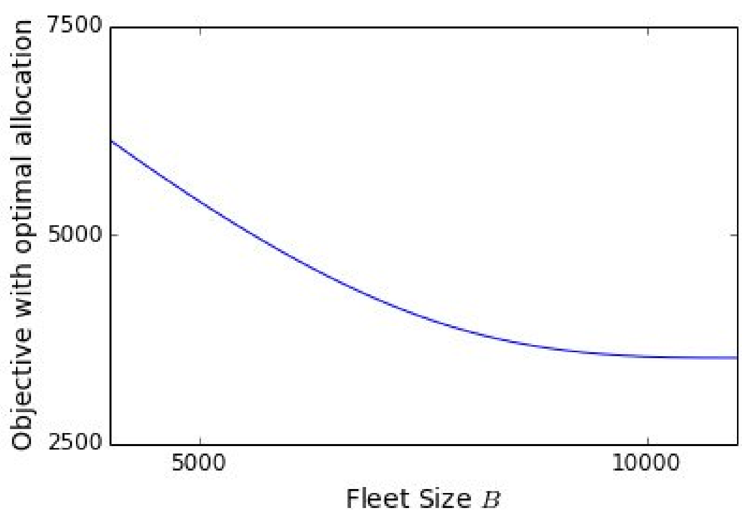
"Cornell came up with a way to very simply measure the impact of a dock's availability on customers which meant that we could move docks around and before we moved them

Figure 4 The Maps Show the Optimal Allocations of B Bikes for the Morning Rush for Different Values of B



Note. Stations shaded more deeply indicate a larger number of bikes allocated. The graph on the left shows $B = 5,000$; the graph on the right shows $B = 10,000$. Map data: ©2018 Google.

Figure 5 In the Graph, We Evaluate the Objective of the System-wide Optimal Allocation of Bikes for Different Fleet Sizes



Note. Although a value of $B = 11,046$ is required to set all stations to their target levels, the improvement beyond $B = 10,000$ is negligible.

have a pretty good understanding of the impact on individual customers every single day" (Motivate 2018).

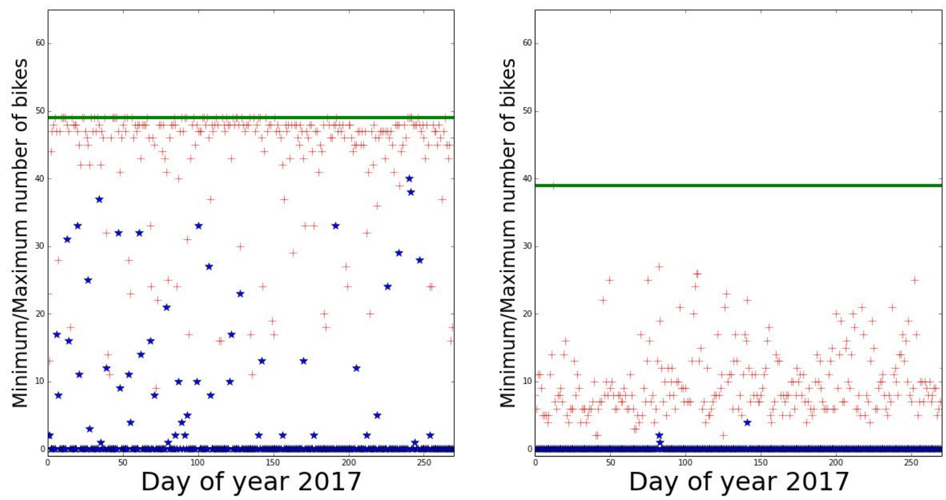
Through our multiyear effort to develop analytical tools to aid Motivate's operational decision-making with regard to rebalancing, we developed a deeper understanding of the demand patterns that underlie Motivate's systems. Through this we identified the potential of reallocating capacity (docks) across stations. Specifically, we found in our analysis that docks were much more consistently utilized at some stations than at others (cf. Figure 6). In a way, this was not a surprising discovery. The sizing of stations in Motivate's systems was largely decided upon before the systems themselves launched, that is, without prior observation of the actual demand patterns. In this section, we describe the optimization methods that ultimately led to Motivate reallocating hundreds of docks to reduce stockouts. As indicated earlier, these results were also described in the brief announcement Caro and Tang (2018).

Integer Programming Model

A natural approach to optimize over allocations of docks is to extend the integer program for the system-wide allocation of bikes. In contrast to the previous formulation, the capacities at each station, denoted by K_i, are no longer treated as an input; instead, they are additional decision variables. To avoid the optimization model simply adding capacity everywhere, we bound the number of docks used in a solution, that is, $\sum_i K_i$, by the existing system-wide capacity. Furthermore, our formulation also lends itself to exploring the impact of an increased system-wide capacity.

There are several questions, both technical and practical, that need to be addressed to use this formulation. First, from a technical perspective, it is not a priori obvious that the resulting nonlinear IP can be solved. In contrast to the optimization over bikes only,

Figure 6 **The Graphs Show the Minimum (Blue Star) and Maximum (Red Plus) Number of Docks Used at Two Stations in NYC During the First Nine Months in 2017**



Note. The horizontal line in each plot marks the capacity of each station. On most days, the station on the left uses almost all of its capacity; the station on the right rarely uses more than half of its capacity.

the integrality property of this IP's linear relaxation need not hold. Second, for practical purposes, the optimal solution may involve more reallocated docks than stakeholders would approve. This could be due to political constraints (e.g., from the Department of Transportation) or due to operational constraints (e.g., from Motivate). Our algorithmic solution addressed both problems simultaneously, as we explain next.

Gradient-descent Search: Local and Global Optima

Given the IP (cf. Appendix B), we can view the set of feasible solutions as the different allocations of bikes and docks to stations subject to both budget constraints (for both bikes and docks) and bounds on potential (dock) capacity of each station. We can then define an undirected graph on the set of feasible solutions by associating one node with each feasible solution. The adjacency of nodes is defined as follows: two nodes are adjacent if their respective allocations differ by at most one dock and one bike being reallocated. A "local optimum" in this graph can then be defined as a node with objective value no more than that of each node adjacent to it. We can optimize over this graph through an intuitive analogue, in a discrete sense, of gradient-descent algorithms: given a feasible solution and its corresponding node in the graph, we iteratively update to the best solution in the neighborhood of the solution currently obtained. We can prove that if we start at a node corresponding to the optimal bike allocation for the current dock allocation, and repeatedly update until we reach a local optimum, then, for any integer k, k such updates yield the optimal solution among those obtainable by moving at most k docks. This allows us to solve the integer program with the additional constraint bounding the number of docks moved.

The proof of this result relies on a more general property of the user dissatisfaction functions, called multimodularity, which was introduced by Hajek (1985); although the precise definition is somewhat involved, one can view this property as a kind of multidimensional diminishing-returns property. For example, consider a station for which the current allocation is 10 empty docks and 10 full docks, and the improvement gained by adding one full dock (i.e., a bike and a dock). That improvement is at least as much as is gained by adding a full dock to the same station if it is already allocated 11 empty docks and 10 full docks. An inductive proof over all sample paths, along the same lines as was used to establish convexity just in the number of bikes, can be used to show that the UDFs are multimodular.

The intuition underlying the result that in the kth iteration we find the best allocation obtainable by moving at most k docks is based on a weaker statement, that is, that without the constraint on the number of docks moved, the local optimum found is a global optimum. The proof of this statement is instructive, and illustrative of the way in which we exploit multimodularity. Suppose, for a contradiction, that a local optimum is not a global optimum. Then find another feasible solution in the graph with a better objective function value, and among all such solutions, choose the feasible solution closest to the local optimum in the graph (where "closest" is defined with respect to the number of edges in the shortest path between them); if there are multiple such nodes that are equally close, choose one arbitrarily. Suppose that u is the node corresponding to the local-but-not-global solution, and v is that closest node just selected. The node v need not be a global optimum either; it is simply a node that is better than u. Consider the shortest path between nodes u and v, and let w be the node adjacent to v that is traversed in this path just prior to reaching v. By the choice of v, the objective function value of w is worse than it is for v; in other words, the local move made in changing the solution at w to become v is an improving one. However, the multimodularity property allows us to argue that making the same change to node u must yield at least as much improvement, contradicting the local optimality of the feasible solution corresponding to node u. The same ideas can then be extended to guarantee that the solution we find in the k iterations is globally optimal when restricting ourselves to moving at most k docks, even though in that case it is not in general true that local optimality guarantees global optimality.

Best in k Iterations

The discrete gradient-descent algorithm we derived is significantly more efficient than relying on general-purpose integer optimization techniques. Moreover, we were able to provide Motivate with a complete package to run analyses from parameter estimation to visualizations of optimal solutions, partly because we do not rely on general-purpose IP solvers. But most importantly, the fact that k iterations of the algorithm yield *the best allocation that can be obtained by moving at most k docks* addresses practical concerns about a larger number of docks being moved than stakeholders will countenance. In particular, we can run the algorithm for only k iterations, and thereby obtain the solution that is optimal for the additional constraint on movement of docks. On a theoretical level, this result is of interest because it introduced a class of constrained optimization problems to the discrete convexity literature that previously had not been studied. In practice, we find that the potential of reallocated capacity faces strong diminishing returns. For example, in NYC most of the potential of reallocated capacity can be realized through strategic reallocations of a few hundred docks; yet to achieve the actual optimal allocation (without consideration of the constraint on docks moved) would require moving thousands of docks.

Algorithmic Efficiency

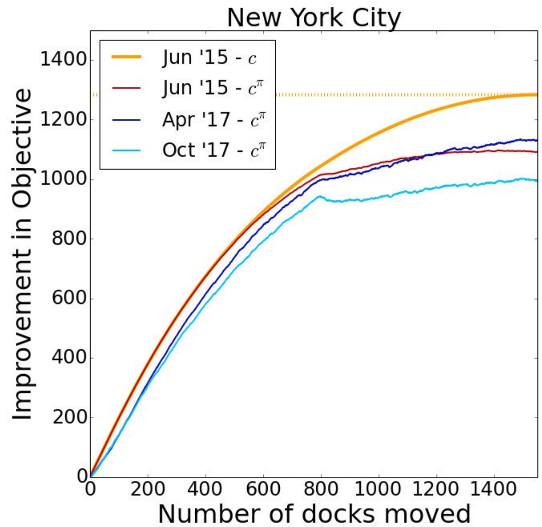
The simple search procedure of the gradient-descent algorithm, implemented with the appropriate data structures, solves real instances to optimality within minutes. A more fundamental change, in terms of efficiency, is based on generalizing the structure of the gradient-descent procedure using so-called scaling techniques; such techniques have long been investigated in the context of multimodularity and other forms of discrete convexity (Murota 2003). The gradient-descent algorithm makes a unit bike/dock change to the system in each iteration; instead, we can consider a series of phases in which the algorithm starts by making "big" gradient-descent steps (by changing bike/dock allocations, for some integer ℓ, in units of 2^ℓ bikes/docks instead of one at a time), repeatedly finding a local optimum with respect to the coarser step and using that result as an initial solution with respect to the next level of refinement (say, changing by $2^{\ell-1}$). These ideas (Freund et al. 2018b) lead to both empirical and (nontrivial) theoretical improvements to the running time achieved by our approach.

Robustness

In contrast to the daily rebalancing of bikes, the physical reallocation of docks is a much more complicated operational procedure, requiring specialized equipment. Dock reallocations should thus be thought of (at most) on an annual basis. Since demand is heavily affected by seasons, the reallocations should be robust with respect to these seasonal changes. Further, many systems continue to expand their footprint within a city, with such expansions having an impact (in complicated ways) on the demand at existing stations. The number of stations in NYC, for example, has increased from 330 to over 700 since 2015. Intuitively, it would thus not be surprising if demand changed so much that reallocations suggested based on 2015 demand data had a negative impact on the system performance with current demand. To gain confidence that this is not the case we evaluated the impact of reallocations (obtained by optimizing over the demand estimates in one month) based on the demand estimates from different months, years, and seasons. Despite the strong seasonal effects on total demand, the analysis (cf. Figure 7) showed that the improvement from reallocated capacity is extremely robust. In particular, reallocations suggested based on data from summer 2015 yield almost the same reduction in stockouts when evaluated based on data from Spring or Fall 2017.

In our collaboration, Motivate expressed concern about an additional issue of robustness that was not captured by our IP formulation: since the IP jointly optimizes over allocations of bikes and docks, it effectively identifies the optimal allocation of docks given perfectly allocated bikes. However, since we had set out to reduce the need for motorized rebalancing, assuming perfect rebalancing seems to contradict that original goal. To ensure that our recommendations would be robust under imperfect rebalancing, we defined an objective that assumes no rebalancing at all: in particular, we computed the average number of out-of-stock events over infinitely many days with absolutely no rebalancing (assuming our demand estimates). In particular, in this objective, defined as the long-run average of the UDF, the number of bikes at a station at the beginning of the day is not a decision-variable, but is instead induced by the demand at that station.

Figure 7 **The Figure Shows the Estimated Improvement in the Objective as a Function Docks Moved**



Note. We optimize with respect to the objective with perfect rebalancing and demand estimates from June 2015. In each iteration of the algorithm we compute the optimal reallocation of k docks. To show the robustness of the improvement, we also evaluate the objective of each allocation with respect to the long-run average objective (c^π) with demand estimates from (i) June 2015, (ii) April 2017, and (iii) October 2017.

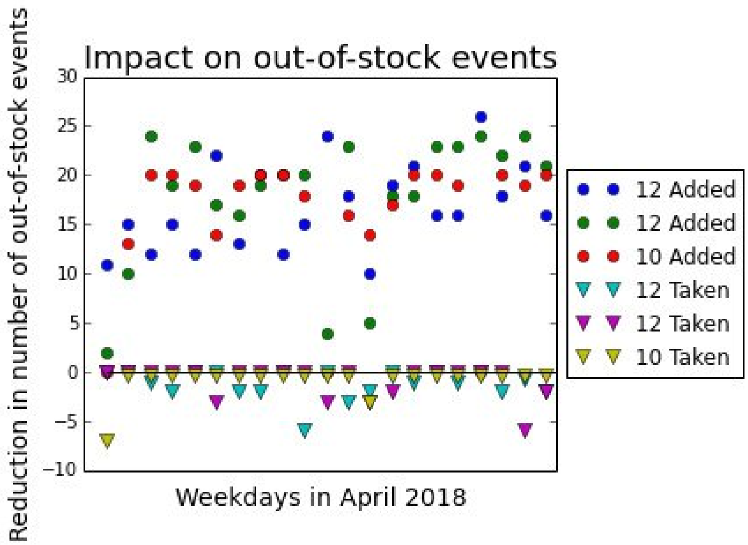
Since the long-run average of the UDF is a different objective, it would not be surprising if optimal solutions for our original IP (evaluated with the normal UDF) fared badly with regard to it. Indeed, it is easy to artificially construct distributions for which optimizing with respect to one of the objectives gives bad solutions when evaluating with the other. However, we do not find this behavior on real data. Indeed, when optimizing for one objective, we obtain a near-optimal solution for the other. Interestingly, this is not due to the optimal solutions for the different objectives being very similar. Rather, it seems to result from the objective functions being flat in the direction of the other optimal solution. In particular, the optimal solutions (in the two regimes) perform, in either regime, significantly better than the current allocation (cf. Figure 7), despite being about two-thirds as far away from each other (in L_1 distance) as they are from the current allocation. Intuitively, one might expect such behavior in a system with very strong antipodal demand patterns in the morning and evening. Consider, for example, a station with unlimited demand for rentals in the morning (and no demand for returns) and unlimited demand for returns (and no demand for rentals) in the evening. Each added dock (with a bike, in the model with rebalancing) at such a station reduces both objectives by two: it enables an additional rental in the morning and an additional return in the evening. Usually, this is the most one can hope for when adding capacity. Thus, stations where added capacity makes the largest improvement are also the ones where the two objectives converge.

Implementation and Evaluation

In November 2017, Motivate launched a pilot project in NYC that entailed the relocation of 34 docks. As part of the pilot, three stations had docks added and three stations had docks removed. Based on the realized demand observed after the reallocation of docks, we were able to do a counterfactual analysis: for the three stations at which the capacity was increased, we computed, for weekdays in April 2018, the number of stockouts that would have occurred with the same demand that was realized but without the additional capacity, that is, the reduction in stockouts that was due to reallocated capacity. This analysis depended only on data and not on distributional assumptions on the demand. To measure the cost (in terms of additional stockouts), we then did the same analysis, with our fitted stochastic models, for the stations at which docks were removed.

In our analysis (cf. Figure 8), we found that the average reduction in stockouts per reallocated dock came to about 1.42 per day. In particular, at stations with increased capacity, additional docks reduced stockouts on average by 1.5 per dock per day, and at stations with reduced capacity we estimated an increase in stockouts of about .08 per dock per day. With average service quality kept constant, Motivate would thus be able to rebalance 1.42 fewer bikes (per dock reallocated) every day, saving tens of thousands of dollars with only a tiny fraction of docks moved. Motivate has since deployed the same

Figure 8 **The Graph Illustrates Reductions (Circles) and Increases (Triangles) in Out-Of-Stock Events at Stations with Capacity Changes Evaluated on Each Weekday in One Month**



optimization approach in its other systems to guide the reallocation of hundreds of docks, while continuously monitoring the resulting impact on out-of-stock events. Comparing that impact (on stockouts) to that of rebalancing, and taking into account the cost of both rebalancing of bikes and reallocations of docks, the cost of moving docks generally pays off in as little as two to five weeks.

Bike Angels

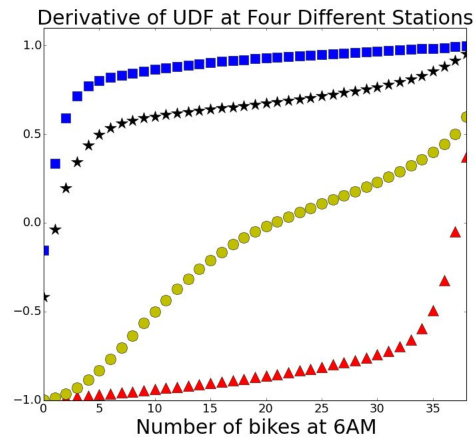
One natural approach to modulate demand for a bike-sharing system would be to adopt a framework of dynamic pricing and by doing so, provide a means to circumvent the need for rebalancing. This approach has been adopted (with both great effect and public derision) in the related realm of ride sharing. For Motivate, such an option is not feasible, because its systems are obliged (by its city partners) to offer customers annual subscription plans; these plans effectively offer customers *all you can ride* for a low price. For example, in NYC an annual plan costs only a little more than a one-month pass for the Metropolitan Transportation Authority. In 2015, we suggested to Motivate that despite the annual plans, systems could modulate demand by providing incentives to encourage rides that are beneficial for system balance.

The proposed incentive program was based on a map that labels each station as being in one of three different classes: `neutral`, `return`, or `rent`. Given that map, customers would receive points for trips undertaken from `rent` stations to either `neutral` or `return` stations and for trips from `neutral` stations to `return` stations (where trips from `rent` to `return` stations receive double points). As such, the setup involved two major design decisions: how do we label each station at a given point in time, and what rewards should customers receive for the collected points; we focus on the former.

In the first trial of an incentive program, we used fixed labels for each station in each rush period, that is, we decided up front on the `neutral`/`return`/`rent` label for each station and kept those labels fixed throughout the rush period; we used the same labeling for each weekday. In terms of user experience, there is an obvious advantage to such *static* labels: customers can plan on the same trip getting them the same number of points every day. In terms of efficiency, however, setting labels up front comes at a cost: given the randomness in daily usage, the program sometimes awarded points for trips that failed to improve the system balance.

To quantify potential failures in efficiency, we need a metric to evaluate the impact of each rental and each return. For ease of exposition, we restrict attention to returns. To derive such a metric we compute, for each incentivized return, the change it caused in the UDF for the return station, that is, the difference between the value of the UDF before and after the return. In essence, we compute the discrete derivative of the UDF at the station with respect to one additional bike (cf. Figure 9). One can show through a sample-path argument that this difference is always bounded between -1 (worst possible:

Figure 9 The Graph Shows Discrete Derivatives for Each of the Four UDFs in Figure 1

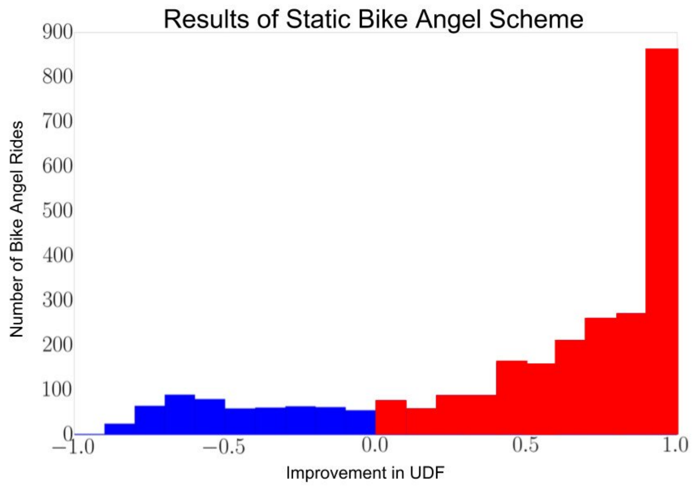


the return increased the number of future out-of-stock events by one) and 1 (best possible: the return decreased the number of future out-of-stock events by one). Figure 10 visualizes the computed changes for both rentals and returns incentivized by the static program. We find there that (1) most incentives, by far, were given for rentals/returns that have positive impact and (2) a nontrivial portion of incentives went to rentals/returns that have no positive or even a strictly negative impact.

Because the discrete derivatives can be computed in real time, we could also use them to guide decisions on where/when to incentivize; in particular, this would guarantee that incentives are given if and only if the rental/return reduces the expected number of future out-of-stock events. However, setting the labels in real time comes at a cost in terms of user experience: because the status of a ride is not determined even at the time of the rental, the user must make decisions without knowledge of whether any particular return will be rewarded. As such, it is natural to investigate the trade-off between efficiency on the one hand and predictability of incentives on the other.

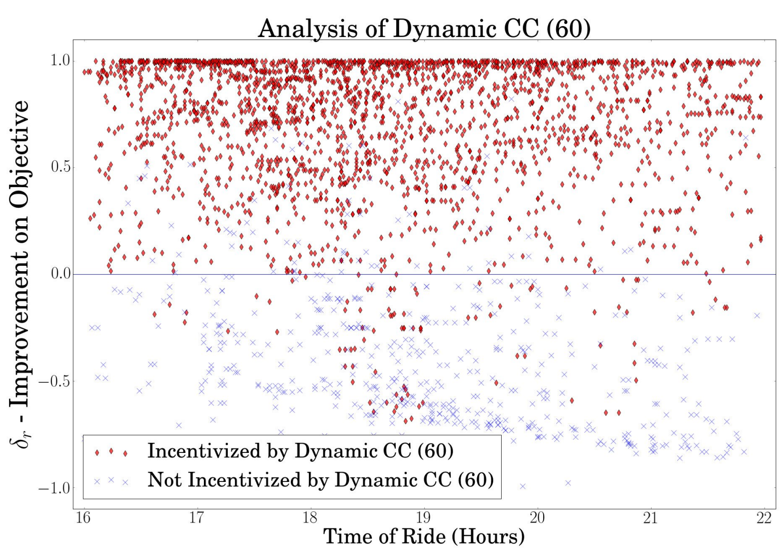
In our study (Chung et al. 2018), we used the data set from the static program to investigate the frequency of relabeling needed to maintain near-perfect efficiency. With that data, relabeling stations every 15 minutes suffices to ensure that exactly the right

Figure 10 The Histogram Shows Discrete Derivatives of the UDFs for Rewarded Rentals and Returns in the Static Incentive Program



Note. Most rewards are justified because they lie to the right of 0; however, a nontrivial portion of the rewards cause the system balance to degrade.

Figure 11 A Scatterplot Depicts the Set of Incentivized Rentals and Returns by the Static Program for a Limited Period During Afternoon Commutes



Note. Each point stands for one rental/return; the y-value provides the associated improvement for the UDF. A dynamic policy less frequently updated than the one now in place at Motivate would have incentivized only the ones indicated by red diamonds, and not those indicated by blue crosses.

rentals/returns are incentivized. As one further increases the length of the time intervals, the efficiency degrades smoothly. For example, relabeling every 60 minutes retains more than 97.5% efficiency (cf. Figure 11). This led Motivate to adopt an incentive scheme in which the discrete derivatives of the UDFs, updated on a quarter-hourly basis, dictate whether a station is incentivized.

Our work with Motivate originally proposed incentives, set up the analytics for the first pilots, identified inefficiencies with the static scheme, and informed the design of today's program. The impact of the Bike Angels program on service quality in NYC now matches that of motorized rebalancing, but at a much lower financial and environmental cost; furthermore, Motivate has also launched Bike Angels in its Ford GoBike program in the San Francisco Bay Area and in its Capital Bikeshare program in the Washington, DC metropolitan area.

Conclusion

Below, we quote Jay Walder, the former CEO of Motivate.

> *"So much of our decision-making was happening with intuition and the work with Cornell frankly was the foundation of this pivot for this company from an operating company working with intuition to an operating company working with data"* (Motivate 2018).

Our work with Motivate introduced UDFs throughout its enterprise. UDFs now inform the system design with respect to station sizes, they guide dispatchers in deciding how to route vans and box trucks to rebalance bikes, and they power the Bike Angel incentive programs in NYC and elsewhere.

But our work with Motivate extends well beyond these projects highlighted in this paper, because there are a multitude of opportunities to optimize operations. For example, a subtle extension of the sample-path arguments used to establish UDFs for single-station performance is at the core of a model that determines an optimal allocation of "trikes" that have been used for nonmotorized mid-rush-hour rebalancing (cf. Figure 12); as a consequence, the problem reduces to finding a minimum-cost matching of cardinality k in bipartite graphs (which can be easily solved). Furthermore, integer programming models were used to run pilot studies for optimized overnight truck routing for rebalancing given time-limited capacity to move bikes with a handful of trucks (Freund 2018, Chap. 6). Finally, optimization models were employed in the allocation of valets to staff corrals, where targeted stations had, in effect, expanded surge capacity during the day (cf. Figure 13).

Having a single measure across Motivate's operations in NYC allows the company to quantify the relative impact on service through apple-to-apple comparisons. Further, many of these methods have been exported from NYC to their other systems nationwide, putting Motivate's operations across the country on a sophisticated analytical footing.

Bike Angels and the dock reallocation efforts use UDFs in particularly creative ways to not only improve customer access to the system, but also to save costs for Motivate. In addition, they do so in an environmentally sustainable way. Conservative estimates on these costs show that using vehicular balancing to achieve the same effect of even just Bike Angels in NYC would cost over \$1,000,000 per year and create an additional 500 tons of CO_2 emissions per year. The extent to which Bike Angels has become part of popular culture in NYC is reflected in a short documentary, cleverly titled "The Point of a Ride" (Gerard 2018), which premiered at the 2018 Bicycle Film Festival in New York, and news stories (NPR 2018) about individual successful Bike Angels participants. Finally, they gave rise to intriguing mathematical challenges, which led to the development of new algorithmic results with potential applications beyond bike sharing.

Figure 12 **The Picture Shows a Trike that Citi Bike Uses to Transport up to Five Bikes in NYC**

Figure 13 **The Picture Illustrates Part of a Staffed Corral in New York City**

Note. By using all physical space available, Citi Bike can extend its the capacity far beyond the number of docks..

Acknowledgments

This work was supported in part by NSF grants CCF-1740822, CCF-1526067, CMMI-1537394, CCF- 1522054, CMMI-1200315, CCF-0832782, and CCF-1017688 as well as Army Research Office grant W911NF-17-1-0094. It would not have been possible without Motivate's commitment to becoming a data-driven company and its openness to trying new ideas.

Appendix A. User Dissatisfaction Functions

Denote a sequence of arriving customers as $X_1,\dots,X_T$ where $X_i \in \{-1,1\}$ denotes a customer returning a bike if $X_i = 1$ and a customer renting a bike if $X_i = -1$. We model a station with capacity K and b_0 bikes at the beginning of the planning horizon (before $X_1,\dots,X_T$ arrive) as having b_t bikes after the arrival of customer t where

$$b_t = \min\{\max\{0, b_{t-1} + X_t\}, K\}.$$

The minimum and maximum in the formulation ensure that the number of bikes is always between 0 and K. For example, if customer t wants to rent a bike (i.e., $X_t = -1$) but the station is empty (i.e., $b_{t-1} = 0$) then customer t cannot rent a bike and the number of bikes remains at zeor (i.e., $b_t = 0$). Given that we define dissatisfied customers as ones who leave without a rental/return we can then write the number of dissatisfied customers among $X_1,\dots,X_T$ as $c(b_0,K)$, as the number of indices t such that $b_t = b_{t-1}$ (i.e., the number of customers who unsuccessfully rent/return a bike and thus leave the number of bikes unaltered). The number of dissatisfied customers is then always a function of b_0 and K as well as the sequence $X_1,\dots,X_T$. We assume that a distribution of sequences is estimated through decensoring techniques (O'Mahony and Shmoys 2015); with arrivals given by Poisson arrival rates we can then compute, for given b_0 and K, analytically the expected number of dissatisfied customers over all sequences of arrivals (O'Mahony 2015).

Appendix B. Optimization Formulations

Denote the number of docks at Station i by K_i, the number of bikes by (a decision variable) b_i, the UDF by $c_i(b_i,K_i)$, and the total number of bikes available by (an input parameter) B, the nonlinear integer program discussed in the paper is

$$\begin{array}{rll} \text{minimize}_{\vec{b}} & \sum_i c_i(b_i,K_i) & \\ s.t. & \sum_i b_i & \leq B, \\ \forall i & 0 \leq b_i & \leq K_i. \end{array}$$

The first constraint ensures that the total number of bikes used is no larger than the number of bikes available; the second constraint ensures that the number of bikes placed at a station does not exceed the number of docks at that station.

When we discuss the system-wide optimization for bikes, the value K_i giving the number of docks at Station i is viewed as a fixed input parameter. Later, when dock moves are considered, K_i becomes a decision variable in addition to b_i, and the formulation is augmented with a constraint on the number of available docks. In that case, we can denote the current number of docks at each station i as $\bar{K}_i$, which allows us to also write the constraint

$$\sum_i |K_i - \bar{K}_i| \leq 2k$$

to ensure that at most k docks are moved. Here, the coefficient of 2 on the right side is needed because a dock moved from i to j not only decreases the number of docks at i by 1 but also increases the number of docks at j by 1, (i.e., the 1-norm distance, sometimes called Manhattan metric, between two solutions with one dock moved evaluates to 2).

References

Caro F, Tang CS (2018) The 3rd poms applied research challenge 2018 awards. *Production and Operations Management* 27(12):2339–2349.

Chung H, Freund D, Shmoys DB (2018) Bike angels: An analysis of citi bike's incentive program. *Proc. 1st ACM SIGCAS Conf. Comput. Sustainable Soc.*, 5:1–5:9 (New York: Association for Computing Machinery), URL `http://dx.doi.org/10.1145/3209811.3209866`.

de Chardon CM, Caruso G, Thomas I (2016) Bike-share rebalancing strategies, patterns, and purpose. *Journal of Transport Geography* 55(July):22–39.

Freund D (2018) *Models and Algorithms for Transportation in the Sharing Economy*. Ph.D. thesis, Cornell University.

Freund D, Henderson SG, Shmoys DB (2018a) Bikesharing. M H, ed., *Sharing Economy: Making Supply Meet Demand*, 435–459 ((Springer Nature Switzerland AG, Cham CH)).

Freund D, Henderson SG, Shmoys DB (2018b) Minimizing multimodular functions and allocating capacity in bike-sharing systems. *arXiv preprint* URL `https://arxiv.org/abs/1611.09304v3`.

Gerard P (2018) The point of a ride. `https://www.youtube.com/watch?v=HWZyfvN7Vg8`.

Hajek B (1985) Extremal splittings of point processes. *Mathematics of operations research* 10(4):543–556.

Hochbaum DS (1994) Lower and upper bounds for the allocation problem and other nonlinear optimization problems. *Mathematics of Operations Research* 19(2):390–409.

Jian N, Freund D, Wiberg HM, Henderson SG (2016) Simulation optimization for a large-scale bike-sharing system. *Proceedings of the 2016 Winter Simulation Conference*, 602–613 (IEEE Press).

McCloskey JA (2015) Tweet about citi bike. URL `https://twitter.com/JohnAMcCloskey/status/641680160811524097`.

Motivate (2018) Motivate commentary on work with Cornell. `https://www.youtube.com/watch?v=_iXjZvXHQuI`.

Murota K (2003) *Discrete convex analysis* (SIAM).

National Association of City Transportation Officials (2018) Bike share in the u.s.: 2017. URL `https://nacto.org/bike-share-statistics-2017/`.

NPR (2018) Citi Bike angel keeps wheels turning for other bike-share users. NPR Morning Edition, September 11 2018. `https://www.npr.org/2018/09/11/646567194/citi-bike-angel-keeps-wheels-turning-for-other-bike-share-users`.

O'Mahony E (2015) *Smarter Tools For (Citi) Bike Sharing*. Ph.D. thesis, Cornell University.

O'Mahony E, Henderson SG, Shmoys DB (2016) (Citi)Bike Sharing, working paper.

O'Mahony E, Shmoys DB (2015) Data analysis and optimization for (citi) bike sharing. *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 687–694.

Parikh P, Ukkusuri S (2014) Estimation of optimal inventory levels at stations of a bicycle sharing system. *TRB 94th Annual Meeting* .

Press Office New York City Hall (2018) URL `{}https://www1.nyc.gov/office-of-the-mayor/news/332-18/mayor-de-blasio-that-advance-l-train-disruption-citi-bike-will-increase-coverage`.

Raviv T, Kolka O (2013) Optimal inventory management of a bike-sharing station. *IIE Transactions* 45(10):1077–1093.

Ross JA (2017) Tweet about divvy. URL `https://twitter.com/JeremyAdamRoss/status/918323581477679105`.

Schuijbroek J, Hampshire R, van Hoeve WJ (2017) Inventory rebalancing and vehicle routing in bike sharing systems. *European Journal of Operational Research* 257(3):992 – 1004, ISSN 0377-2217, URL `http://dx.doi.org/https://doi.org/10.1016/j.ejor.2016.08.029`.