Information Fusion for (Re)Configuring Bike Station Networks With Crowdsourcing

Suining He and Kang G. Shin

Abstract—Bike sharing service (BSS) networks have been proliferating all over the globe thanks to their success as the first/last-mile connectivity inside a smart city. Their (re)configuration — i.e., station (re)placement and dock resizing — has thus become increasingly important for BSS providers and smart city planners. Instead of using conventional labor-intensive manual surveys, we propose a novel information fusion framework called CBikes that (re)configures the BSS network by jointly fusing crowdsourced station suggestions from online websites and the usage history of bike stations. Using comprehensive real data analyses, we identify and exploit important global trip patterns to (re)configure the BSS network while mitigating the local biases of individual feedbacks. Specifically, crowdsourced feedbacks, station usage, cost and other constraints are fused into a joint optimization of BSS network configuration. We also model the spatial distributions of station usage to account for and estimate the unexplored regions without historical usage information. We further design a semidefinite programming transformation to solve the bike station (re)placement problem efficiently and effectively. Our extensive data analytics and evaluation have shown CBikes’ effectiveness and accuracy in (re)placing stations and resizing docks based on three large BSS systems (with > 900 stations) in Chicago, Twin Cities (Minneapolis–Saint Paul), and Los Angeles.

Index Terms—Bike sharing, urban planning, crowdsourcing, information fusion, semidefinite programming, urban computing

1 INTRODUCTION

With the advent of smart cities/communities and Internet of Things (IoTs), the urban sharing economy has been evolving very rapidly. In particular, bike sharing service (BSS) has emerged as one of the most popular and revolutionary powers that change the people’s urban life/health. Bike sharing enables the first/last-mile urban travel to be more economic, greener and healthier than traditional gasoline-engine-powered vehicle riding. City transportation also benefits from an additional network of bike stations connected by the trips with less hassle of traffic planning.

Experiencing deployment successes and receiving positive feedbacks, many BSS providers have begun expanding their BSS networks. Owing to such an expansion, the global bike sharing market is expected to grow at a compound annual growth rate of 21 percent during 2018–2022. For example, Divvy bicycle sharing program in Chicago, IL is adding 10,500 new bikes and 175 additional stations over the next three years from 2019. Meanwhile, Citi Bike in New York City will embrace another 4,000 bikes, 13 stations in the busiest areas and 2,500 docks since 2019. On the other hand, there exist BSS network shrinkages (at a micro or macro scale) for financial, event, seasonal or meteorological reasons.

With dynamic bike usage and complexity of urban environments, how to expand and shrink, or (re)configure the existing network of BSS stations becomes increasingly important for the BSS providers. As stations and bicycles are dynamically added/deleted/resized during the BSS (re)configuration, the station relocation, or station (re)placement (i.e., add, move or remove a station), as well as their dock resizing becomes challenging, involving more thorough site investigation and labor-intensive user surveys.

To better leverage the collective knowledge from the BSS users [5], many service providers, like aforementioned Divvy in Chicago, have attempted to crowdsource various station placement comments via their own websites, as illustrated in Fig. 1. Interested users can easily pinpoint, comment and vote for various potential station locations on an interactive map. This way, the BSS systems can easily and timely obtain many online feedbacks for their next stage expansion or shrinkage, while reducing their traditional survey and investigation costs significantly.

Despite its importance, however, how to (re)configure the BSS network based on the aforementioned crowdsourced comments is still very challenging and remains an open problem due to the following concerns:

- From the data perspective, the first challenge lies in the heterogeneity of information inputs. Crowdsourced feedbacks usually provide local, fragmented suggestions due to each individual’s limited geographic scope or personal interest/preference (say, close to home residence), while BSS network (re)configuration needs global knowledge of user mobility
and station-to-station dynamics. How to incorporate the local suggestions/comments together is important and should thus be considered carefully.

- From the user’s perspective, as all stations are “linked” by users’ trips, the second challenge stems from their trip tendency. Overcrowded or inadequate BSS network placement and ignorance of popular station-station pairs for users’ commute may discourage cyclists, thus lowering bike usage and platform profit.
- From the platform’s perspective, since the web crowds are enabled with large freedom to label locations they want, addressing such naturally-noisy/biased crowdsourced inputs becomes the third challenge, which should be considered by a joint fusion formulation.

To address above challenges, we propose CBIkes, a novel joint information fusion framework for Crowdsourced Bike sharing Station network (re)configuration. Specifically, CBIkes integrates local crowdsourced suggestions with global historical bike usage data upon a geographical map which is discretized into regions/grids. The information fusion in CBIkes not only takes into account the usage at deployed/explored city regions, but also estimates the usage at the unexplored/expansion ones. Given above, CBIkes converts BSS network (re)configuration into a graph matching problem. Each vertex (station) of the graph (network) is matched against this spatially and temporally-varying map of fused knowledge, subject to edges (links) or trips from others. We then formulate a novel joint optimization problem to balance among crowd satisfaction, platform utility, and (re)configuration cost.

CBIkes makes the following major contributions:

* Comprehensive (re)configuration data analysis: We analyze extensive real data of several BSS (re)configuration cases, and identify the important properties of their bike usage distribution evolution, BSS network density alternation, trip correlations between bike stations and crowdsourced feedbacks for the BSS systems.
* Novel data-driven and computational model designs: We derive important and practical data-driven model designs for bike sharing station network, including a novel metric for user trip tendencies, predicted usage at unexplored city grids and inter-station distance constraints, and integrate them in CBIkes.
* Crowdsourced information fusion & joint optimization: We propose a novel optimization framework which jointly considers multi-modal data from crowdsourcing and platform-usage statistics for BSS (re)configuration. We first formulate a grid-based candidate selection and graph matching problem, then transform it into a novel semidefinite programming (SDP) form, and finally solve it efficiently and effectively.

* Extensive experimental evaluation: CBIkes has been evaluated with significant amounts of real data (of more than 900 stations) from 3 premium BSS systems in Chicago, IL, Twin Cities (Minneapolis–Saint Paul),2 MN and Los Angeles, CA. These comprehensive studies validate the effectiveness and accuracy of CBIkes in optimizing bike sharing station (re)configuration given crowdsourced inputs.

Despite its focus on BSS systems, CBIkes can be extended to other sharing/connected vehicle network (re)configuration, including parking lot decisions for car-sharing [37], gas station redeployment [32] and charging station expansion for electric vehicles [14].

A preliminary version of this work was presented at a conference [10]. Besides motivating and elaborating more upon the core formulation (Sections 1, 4 & 7), this version makes significant improvements over the conference version as follows. 

1) Estimated Usage at Unexplored Grids/Regions: The conference version [10] did not model the bike usage at those reconfigured or expansion grids, yielding less accurate grid matching. While many researchers studied demand distribution based on known historical trip data [18], [29], the problem of estimating the demands at unexplored grids/locations has not been investigated. In this version, we have also investigated the latter problem, and developed an efficient multi-layer neural network to estimate the usage distributions at those unexplored grids, which further enhances the (re)configuration (relocation) performance (Section 3).

2) Additional Experimental & Ablation Studies: We have also conducted more experimental evaluations of the proposed framework as well as several important system parameters (including search scope and grid size), validating the comprehensive model designs (Sections 5.2 & 5.3).

3) Performance Improvement: Our new designs have been shown to outperform those in the previous version in terms of (re)placement/resizing accuracy improvement and reconfiguration cost reduction (Section 5.3).

4) Deployment Discussions: We have also added more discussion upon the deployment of CBIkes (Section 6).

The rest of this paper is organized as follows. We first overview the system framework and important concepts for our problem in Section 2. Then, Section 3 presents (re)configuration analysis and data-driven designs, followed by the core problem formulation and novel optimization framework in Section 4. Section 5 provides experimental evaluations, while Section 6 discusses some deployment considerations. After reviewing related work in Section 7, the paper finally concludes with Section 8.
Fig. 2. The system framework flow of CBikes.

2 SYSTEM & CONCEPTS

We present the basic CBikes framework (Section 2.1) and introduce important definitions of CBikes (Section 2.2), followed by the datasets evaluated (Section 2.3).

2.1 System Framework

Fig. 2 shows the components and layers of CBikes. Specifically, CBikes consists of 4 consecutive layers for computing bike station (re)configuration: input, design, core and action layers. At the input layer (Section 2.3), historical and estimated station-usages, crowdsourced feedback of station expansion/shrinkage suggestions, as well as predefined costs are collected and delivered to a central server, preprocessed and then stored into databases. Note that other practical geographic design concerns or constraints, including the number of service bikes and accessible station deployment areas, are also inputted by the service provider, processed and stored into its database. Our focus here is to develop a generic optimization framework, given the above primary and secondary information.

At the design layer (Section 3), we form the joint objective functions, and integrate map information and station geographic distances into constraints. Finally, we formulate a joint optimization framework, transform and solve it at the core layer (Section 4), optimizing station sites with respect to predefined map grids. Guided by the results of the action layer, the service provider may (re)place stations and resize their docks. In case results are not satisfactory, the parameters can be tuned interactively for another optimization trial.

2.2 Key Concepts

We elaborate on the important terms, concepts or definitions for our mathematical formulation. Formally, we have

Definition 1 Bike station network (BSN). Each station $i$ is represented by $S_i = (\text{lat}_i, \text{lon}_i, \kappa_i)$, $i \in \{1, \ldots, M\}$, where tuple $[\text{lat}_i, \text{lon}_i]$ denotes its geographic coordinates and $\kappa_i \geq 0$ is its capacity. Denote the location of each $S_i$ as a $2 \times 1$ vector $\mathbf{l}_i = [\text{lat}_i, \text{lon}_i]^T$, and let $\mathbf{L} = [\mathbf{l}_1, \mathbf{l}_2, \ldots, \mathbf{l}_M]^T$ be the $M \times 2$ coordinate matrix of all stations on the map. Given a set of $M$ geographical nodes $\mathbf{L}$ and their links $\mathbf{E} \subseteq \mathbf{L} \times \mathbf{L}$ connecting them, a network of BSS stations is represented by a graph $\mathcal{G} = (\mathbf{L}, \mathbf{E})$.

Given an already-deployed BSN, after a certain period we obtain

Definition 2 Historical bike trip data. Each trip corresponds to a user’s bike ride which happens at a certain time from a station to another. Specifically, a set of bike trips from a start station $S_i$ to an end $S_j$ can be represented as $\mathbf{r}(i, j) = \{t_i, (i, j)\}$, where $(i, j)$’s are the set of pick-up/drop-off timestamps of each trip in $\mathbf{r}(i, j)$. Note that $\mathbf{r}(i, j)$ is symmetric if and only if riders return their bikes at the same station as they were rented, i.e., $\mathbf{r}(i, j) = \mathbf{r}(j, i)$ iff $i = j$.

Based on the deployment results the service provider may initiate:

Definition 3 Bike station network (re)configuration (BSNR). A phase of BSNR basically consists of station (re)placement and dock resizing. At each BSNR, the service provider can place new stations, remove or move existing ones, or just keep them, and resize the docks. We consider two consecutive stages of a BSNR, i.e., two sets of station status before and after a (re)configuration. For ease of description, denote the $M$ stations before BSNR as $S_i$’s, and let the old (prior to the (re)configuration) network be $\mathcal{G} = (\mathbf{L}, \mathbf{E})$. Each $S_i$’s location before BSNR is denoted as $\mathbf{I}_i = [\text{lat}_i, \text{lon}_i]$, with the pre-(re)configured capacity $\kappa_i$. At each BSNR, we consider (re)placing $M$ stations and resizing the dock capacity to accommodate a total of $K$ bikes.

We note that the number of stations to be (re)placed, $M$, can be determined by the BSS platform as a known input factor (can be represented as a budgetary constraint). BSNR decisions should also involve public engagement and cater to users’ demand. Before each BSNR, via certain media or platform (like a website) interested users may easily suggest station sites, i.e.,

Definition 4 Crowdsourced station feedbacks. Each feedback indexed by $n$ on the interactive map is represented as $f_n = (\text{lat}_n, \text{lon}_n, t_n, \text{text}_n)$, where the pair $(\text{lat}_n, \text{lon}_n)$ is the location/site coordinate, $t_n$ is its timestamp, and $\text{text}_n$ is the related posted comment, if any.

We briefly introduce the actions of BSNR. Station (re)placement is to find their appropriate locations. As searching in continuous geo-space may lead to a computation complexity problem, we discretize the entire map into multiple grids. This way, we have finite candidate sets for efficient computation, whose granularity can be determined via task customization [4], [18]. Formally, we have

Definition 5 Station (re)placement grid. The entire city map is discretized into a set of $R$ regular grids (rectangle grid in our case), i.e., $\mathbf{G} = [\mathbf{g}_1, \ldots, \mathbf{g}_R]^T$, an $R \times 2$ matrix where each grid is given by a coordinate $(2 \times 1$ vector) of its center, $\mathbf{g}_r = [\text{lat}_r, \text{lon}_r]^T$, $r \in \{1, \ldots, R\}$.

Note that $R$, the number of grids, is determined by the trade-off of accuracy, granularity and computational
efficiency (evaluated in Section 5). After station (re)placement, CBikes further resizes their docks.

**Definition 6 Dock resizing.** The total dock capacity equals (or at least) the total number of bikes, i.e., $\sum_{i=1}^{M} k_i \geq K$. CBikes resizes the dock $k_i$ (enlarge, decrease or maintain) at each station $i$ to satisfy both incoming crowdsourced needs and historical/potential demands.

In our prototype studies, we consider the total dock capacity as a pre-determined input by the BSS provider, i.e., $\sum_{i=1}^{M} k_i = K$. Note that the cost of dock resizing only considers those stations staying at the same locations as in $G$. Dock-related costs of other newly-added/removed stations are included in their subtotals of creation and removal.

Profit, cost and station usage are critical from the platform perspective, while matching request and convenience may matter to the users. To accommodate both, we study in this paper:

**Definition 7 Crowdsourcing-based BSNR (CBSNR).**

Given historical bike trip data, crowdsourced feedbacks, cost of actions, and other practical BSS design constraints, CBSNR problem is to (re)configure the existing network to jointly match crowds’ feedbacks and station usage statistics at minimum cost.

### 2.3 Overview of Datasets Studied

We consider the following BSS data (including map information) for our CBSNR analysis here and evaluation in Section 5:

- **Divvy at Chicago, IL**, which consists of total 582 stations by 2017 (2nd quarter). 3 major expansions with total 282 new stations were recorded since 2013. Overall, 11,544,750 trips are studied.
- **Nice Ride at Twin Cities, MN**, which includes a total of 202 stations in Minneapolis-St. Paul Metropolitan area until 2016. 5 major expansions with 134 new stations are recorded since 2013. Overall, 2,857,027 trips are analyzed.
- **Metro Bike at Los Angeles County, CA**, which consists of total 119 stations in Los Angeles (LA) County by 2017 (3rd quarter). 2 major network expansions with total 56 new stations were recorded since 2016. Overall, 277,195 trips are evaluated.

This massive trip data includes start/destination stations, related pick-up/drop-off timestamps (or trip durations), user type (say, day-pass holders or annual subscribers) or even age/gender/birthday information. We further scrape the crowdsourced feedbacks from “Suggest a Station” website of each BSS provider (Divvy,4 Nice Ride5 and Metro Bike6). For each CBSNR, we use the 1,100 latest feedbacks $f_{tn}$’s with $[lat_{tn}, lon_{tn}]$’s (with $tn$ before the BSNR). In our studies, we have filtered out the crowdsourced feedbacks in inaccessible regions (Section 3.5).

Besides aforementioned datasets, we also collect the point-of-interest (POI) data for each BSS system (Chicago: 4,329; Twin Cities: 3,100; LA: 5,948) from the OpenStreetMap (OSM)6 website. As most observations are qualitatively similar, we focus on Divvy and Nice Ride in the following data analytics in Section 3.

### 3 (RE)CONFIGURATION ANALYSIS & DESIGN

The inherent complexity of CBSNR calls for careful and practical designs based on usage data and users’ feedback. We design key components of CBikes and their integration via comprehensive analysis of real data: historical and estimated station usages (Sections 3.1 and 3.2), inter-station trip tendency (Section 3.3), geographic distance constraint (Section 3.4), and finally crowdsourced feedbacks (Section 3.5). For each component, we make important observations from the data (before (re)configuration), and quantitatively formulate the design problem.

#### 3.1 Historical Usage at Each Station

**Observation.** Intuitively, the more often a station was used at a certain location, the more likely it will be kept there. We first summarize and show the spatial station usage w.r.t. BSNR. Figs. 3 and 4 visualize the spatial distribution of usage as a heat map. The warmer the color, the more pick-ups/drop-offs are recorded ($\log_{10}(usage)$). Due to BSNR, clear configuration changes can be seen between 2013 and 2015. More city areas are covered, and higher usages can be observed among the points of interests (including the Skyline and Lake Coast) in Chicago as the network expands. Similar patterns can be observed from Twin Cities and LA County.

**Design.** To better differentiate historical usages of different stations, we design a usage-related measure for each $S_i$ w.r.t. each $g_r$.

Let

$$T_r = \{t(i, j) | (S_i \text{ is at } g_r) \cup (S_j \text{ is at } g_r)\}$$

be the aggregated set of trips starting or ending at grid $r$. We define the **historical usage importance** of $g_r$ for a station location candidate $l_i$ as

$$U_r^i = \frac{\exp(\lambda_r^i |T_r|)}{1 + \exp(\lambda_r^i |T_r|)}.$$  

where

$$\lambda_r^i = \frac{l_i \cdot g_r}{\|l_i\| \cdot \|g_r\|}.$$  

#### 3.2 Inter-station Usage Relationship

**Definition 8 Inter-station Usage Relationship.**

Given each $S_i$ and $S_j$ at $g_r$, we define the inter-station usage relationship as

$$U_{ij}^r = \frac{\exp(\lambda_{ij}^r |T_{ij}|)}{1 + \exp(\lambda_{ij}^r |T_{ij}|)}.$$  

where

$$\lambda_{ij}^r = \frac{l_i \cdot l_j \cdot g_r}{\|l_i\| \cdot \|l_j\| \cdot \|g_r\|}.$$  

#### 3.3 Geographic Distance Constraints

**Definition 9 Geographic Distance Constraints.**

Given each $S_i$ and $S_j$, we define the geographic distance constraints as

$$D_{ij}^r = \frac{\exp(\lambda_{D}_{ij}^r |D_{ij}|)}{1 + \exp(\lambda_{D}_{ij}^r |D_{ij}|)}.$$  

where

$$\lambda_{D}_{ij}^r = \frac{l_i \cdot l_j \cdot d_{ij}}{\|l_i\| \cdot \|l_j\| \cdot d_{ij}}.$$  

#### 3.4 Configuration Optimization

**Objective.** The objective of the configuration optimization is to minimize the total cost subject to the constraints as follows:

$$\min \left\{ \sum_{r \in G} \left( C^{dock} + C^{creation} + C^{removal} \right) \right\}$$  

subject to

$$\sum_{r \in G} \left( U^r_i + U^r_j + D^r_{ij} \right) \geq \theta_r.$$  

where

$$\theta_r = \frac{\exp(\lambda_r^i |T_r|)}{1 + \exp(\lambda_r^i |T_r|)}.$$  

#### 3.5 Crowdsourced Feedbacks

**Observation.** Intuitively, the more often a station was suggested at a certain location, the more likely it will be placed there. We first summarize and show the spatial crowdsourced feedbacks w.r.t. BSNR. Figs. 3 and 4 visualize the spatial distribution of feedback as a heat map. The warmer the color, the more feedbacks are recorded. Due to BSNR, clear configuration changes can be seen between 2013 and 2015. More city areas are covered, and higher feedbacks can be observed among the points of interests (including the Skyline and Lake Coast) in Chicago as the network expands. Similar patterns can be observed from Twin Cities and LA County.

**Design.** To better differentiate crowdsourced feedbacks of different stations, we design a feedback-related measure for each $S_i$ w.r.t. each $g_r$.

Let

$$F_r = \{f(i, j) | (S_i \text{ is at } g_r) \cup (S_j \text{ is at } g_r)\}$$

be the aggregated set of feedbacks starting or ending at grid $r$. We define the **crowdsourced feedback importance** of $g_r$ for a station location candidate $l_i$ as

$$F_r^i = \frac{\exp(\lambda_r^i |F_r|)}{1 + \exp(\lambda_r^i |F_r|)}.$$  

where

$$\lambda_r^i = \frac{l_i \cdot g_r}{\|l_i\| \cdot \|g_r\|}.$$  

#### 3.6 Cost of Dock Resizing

**Definition 10 Cost of Dock Resizing.**

Given each $S_i$ and $S_j$, we define the cost of dock resizing as

$$C^{dock}_{ij} = a_i b_j,$$

where $a_i$ and $b_j$ are the cost of dock resizing at stations $i$ and $j$, respectively.

#### 3.7 Cost of Creation/Removal

**Definition 11 Cost of Creation/Removal.**

Given each $S_i$, we define the cost of creation/removal as

$$C^{creation} = \begin{cases} a_i, & \text{if created} \\ -b_i, & \text{if removed} \end{cases},$$

where $a_i$ and $b_i$ are the cost of creation/removal at station $i$, respectively.

#### 3.8 Cost of Station Expansion

**Definition 12 Cost of Station Expansion.**

Given each $S_i$, we define the cost of station expansion as

$$C^{expansion} = \begin{cases} a_i, & \text{if expanded} \\ -b_i, & \text{if reduced} \end{cases},$$

where $a_i$ and $b_i$ are the cost of expansion/reduction at station $i$, respectively.

#### 3.9 Cost of Distance Constraint Violation

**Definition 13 Cost of Distance Constraint Violation.**

Given each $S_i$ and $S_j$, we define the cost of distance constraint violation as

$$C^{constraint} = \begin{cases} a_{ij}, & \text{if violated} \\ 0, & \text{otherwise} \end{cases},$$

where $a_{ij}$ is the cost of constraint violation at station pair $(i,j)$.
2) geographic distances from the \(P_1\) nearest stations before CBSNR (\(P_1\)-D): Based on the street centerline map, we find the geographic distances between the target grid and each of its \(P_1\) nearest peer stations (before CBSNR). Then, we have \(P_1\) distance measures \([\text{dist}_1, \ldots, \text{dist}_{P_1}]\).

3) number of POIs for each type and total numbers of POIs ((\(P_2+1\))-D): For each grid \(g_i\), we find regarding each type of POI \(i\) (say, business or mall; \(i \in \{1, \ldots, P_2\}\)) the number of venues, \(z_i\), within it. We also find the total number of POIs, \(\sum_{i=1}^{P_2} z_i\), within all the grids of the city. We visualize the distributions of POIs (including restaurants, cafes, banks, supermarkets and many others based on key: amenity in OpenStreetMap (OSM)) in the city of Chicago and Minneapolis-Saint Paul in Figs. 5 and 6.

4) Pol entropy (1-D): Since the functionality of a grid can also be specified by a few certain types of POIs, we further introduce the POI entropy to characterize it as:

\[
\text{z}_{\text{entropy}} = - \sum_{i} z_i \log \left( \frac{z_i}{\sum_{i=1}^{P_2} z_i} \right).
\]

Specifically, we form a \(P\)-D (\(P=P_1+P_2+4\)) feature vector \(z\) consisting of the above factors as input. Given the aggregated historical usage of those explored grids before CBSNR, we train a multi-layer fully-connected (FC) dense neural network to estimate aggregated usage of the unexplored ones, denoted as \(|\hat{T}_r|\). Its layer-to-layer propagation can be given by

\[
z_i = \sigma(W_i \cdot z_{i-1} + b_i),
\]

where \(W_i\) is the neuron weight matrix, \(b_i\) is the bias vector for layer \(i\), and \(\sigma(\cdot)\) is the activation function (we use RELU in our prototype). The output after multiple stacked layers is the estimated \(|\hat{T}_r|\) at each grid, given input of the feature vector \(z\).

After predicting the potential usage \(\hat{T}_r\)’s for each \(g_i\) without historical records, we feed them to Eq. (2) and calculate the estimated usage importance \(\hat{U}_i\) of a grid \(r\) for each station candidate \(i\). This way, CBikes accommodates both historical and estimated usage within the information fusion.

Note that the estimation model presented in Eq. (6) is general enough to accommodate many other factors if available
for better performance. We will further evaluate the beneficial effect of usage estimation upon the reconfiguration of CBikes in Section 5.

3.3 Inter-station Trip Tendency

Observation. Despite its importance, considering total usage only may not be sufficient. For example, a BSS user may frequently commute between a pair of stations (say, her/his home and office or school). Individually considering each station without inter-station trip tendency may overlook such frequently commuting users (which yields a stable platform income) and remove those stations having strong links \( E \subseteq L \times L \) with others.

To further illustrate this, Fig. 7 shows an example of trip tendency among 5 stations in Chicago in 2014. We summarize their pick-up/drop-off flows w.r.t. each outgoing/incoming direction (i.e., a vector between start and destination). Dark blue sectors indicate the volume of outgoing bike flows while light yellow represents incoming bikes. Volumes in all directions are normalized to \([0, 1]\) for each \( S_i \). The larger radius of a sector, the more proportion of its bike flows start or end in that direction. We can observe that a strong north-south trip pattern w.r.t. stations along Lake Michigan beaches mainly because the tourists’ recreational rides create a large trip tendency at stations along the lake shore.

Similarly, Fig. 8 shows the trip tendency to/from several stations in Minneapolis, MN. We can see strong bike flows between west downtown and university area, indicating bike commutes by students, staff and faculty. In particular, we can observe significant south–west and south-east flows at the station of 6th Ave. SE & University Ave. (circled), which likely bridges the downtown and campus. Despite its less total usage (lower \( U_r \) in Eq. (2)) than others, CBSNR should also value importance of this station.

In summary, inter-station trip tendency is highly correlated with purposes of users’ trip choice (start, end), including commutes between home and school or recreational sightseeing. Further, its strength characterizes the volume/tendency of urban flows. Therefore, we incorporate the tendency in our optimization model.

Design. Recall that \( r(i, j) \) represents the set of bike trips from \( S_i \) to \( S_j \) (\( i \neq j \)). To focus on the connectivity and trip-tendency, we adapt the link probability in theories of network embedding [21], and define a new tendency metric \( p(i, j) \) between \( S_i \) and \( S_j \) as

\[
p(i, j) = \frac{1}{1 + \exp(-\bar{a}_i \cdot \bar{a}_j)},
\]

where the vector \( \bar{a}_i \) represents the proportion of trips from \( i \) to \( j \), i.e., \( |r(i, j)| \), as well as that of the remaining trips, i.e.,

\[
\bar{a}_i = \frac{|r(i, j)|}{\sum_{k=1,k \neq i}^M |r(i, k)|}, \quad 1 - \frac{|r(i, j)|}{\sum_{k=1}^M |r(i, k)|},
\]

and similarly for \( \bar{a}_j \). Note that \( p(\cdot, \cdot) \) is symmetric, i.e., \( p(i, j) = p(j, i) \). \( \bar{a}_i \cdot \bar{a}_j \) returns the dot product of the two vectors.

In other words, the larger proportion of bikes are commuting between stations \( i \) and \( j \), the larger \( p(i, j) \) is \((0 < p(i, j) < 1)\), implying more important connectivity of these two stations. Then, we find \( \sum_{j=1,j \neq i}^M p(i, j) \) for each \( S_i \), further indicating its overall connectivity with other stations. This way, we may characterize the complex network structure efficiently [21], highlighting the connectivity and trip-tendency between stations. Considering the frequent usage and travel patterns of bike users, BSNR should preserve interactive connectivities between these stations.

We further visualize in Figs. 9 and 10 the distributions of \( p(i, j) \)’s (normalized) of two stations (red stars) in Chicago and Minneapolis, which correspond to the trip patterns discussed in Figs. 7 and 8. The color of each grid represents the value of the tendency metric of a station there with the
target one. The warmer the color, the larger the metric value, meaning more trips happen between the target station and those neighbors. Thanks to the modeling of $p(i, j)$ we can fuse the user preferences in CBikes’ formulation.

From the data management’s point of view, the total usage and the trip tendency of stations are inherently correlated, as the former is the result of aggregating the latter. To highlight station connectivity and mitigate inherent redundancy, as shown in Eq. (8) we normalize the usage in the model. Besides, our evaluation (Section 5) shows that inclusion of tendency beyond usage improves the performance, which has not yet been considered in previous siting studies [4], [16], [18].

### 3.4 Geographic Inter-Station Distance

The BSS is designed to provide first-/last-mile commute, and a user is allowed to return the bike at any station near her/his destination. Thus, the density of deployed stations is a critical design consideration, i.e., the network should be neither too dense nor too sparse.

**Observation 1.** We first overview the histograms of outgoing trip distances, which characterize the tendency of a user when deciding on a trip. We do not show round trips as they are included in single station usage (Section 3.1). Figs. 11 and 12 show the outgoing trip distance distribution w.r.t. years for each BSS system. We can observe that a clear “last-mile” traffic flow, i.e., more than 65 percent outgoing users tend to drop off bikes within 2km (around 1.5miles).

Interestingly, as BSS expands, increasingly more percentage (88% in 2013 → 90% in 2016) of users take short-distance (<4 km) trips in Chicago, while in Twin Cities this part is decreasing (97.34% in 2010 → 93.3% in 2013 → 89.81% in 2016). It is likely due to the difference in network density. With markedly more nearby stations and available bikes, it is more convenient for Chicagoans to ride between near stations. For Nice Ride, as average distance to nearest station is larger (0.47 km in Divvy versus 0.58 km), under such nearby stations of a sparser network may take less usage percentage.

**Observation 2.** We also show the bike usage of each station versus the distance to its nearest neighbor. This way, we can characterize the impact between stations due to service coverage overlap. Specifically, we conduct negative binomial regression (NBR) [12] on single station usage $|T|$ (the number of trips) against different distances $D$ (m) to the nearest peers. Considering the probability

$$P(|T| = a|D) = \frac{e^{-z}z^a}{a!}$$

and mean of $|T|$ is $z$ [12], NBR finds the set of $b$’s which maximize the log-likelihood for

$$\ln z = b_0 + bD.$$

Fig. 13 shows the regression parameter $b$ versus $D$. $b$ characterizes sensitivity of station usage towards network density. Overall, we observe in both systems a positive effect ($b > 0$) of the distance to the nearest neighbor over the station usage, implying that usage generally increases with distance from the nearest neighbor. A strong counter-effect upon a station can be inferred within a close distance from others (say, less than 400 or 500m) which may lower its usage. It is mainly because of a competitive effect [25] that close-by stations may serve the same group of users and prevent each other from being fully utilized. As a short-range effect, it saturates quickly after a certain range (say, 600m in Divvy and 700m in Nice Ride), due to discouraged usage of distant sites.

**Design.** To reflect the above observations, over $E \subseteq L \times L$ we set the lower/upper bounds $[d_{ij}, \overline{d}_{ij}]$ for the distance between two neighboring stations $S_i$ and $S_j$ (in a neighborhood set $\mathcal{N}$), i.e.,
We apply a heuristic local search [2] around all \(S_i\)'s in \(G\) based on historical usage statistics, crowd feedbacks or their fused map (Section 4.2) to determine a rough neighborhood set of \(N\). As CBikes is a general framework, geographic distances other than the euclidean metric (like the Manhattan distance for metropolitan cities like New York City [33]) can be easily applied. Note that we consider locally constraining neighboring station candidates in close grids (say, within 2 to 3 grids), making differences of metrics rather small in practice. Similar to many state-of-the-art studies [18], [34], for prototype and illustration purposes we consider the euclidean distance here.

For convenience and utility, the upper bound caters to the majority of travel distance preferences, while the lower bound mitigates conflicts between neighboring stations. We consider distance at the 65-percentile of cumulative usage distributions from Figs. 11 and 12 for \(d_{ij}\), and distance at the “knee point” (where the plotted curve “turns”, or formally where a curve is best approximated by a pair of lines) in Fig. 13 for \(d_{ij}^{l}\). Note that all derived parameters for each test are only based on periods before (re)configuration takes place. Despite the global bound setting here, one may easily customize \([d_{ij}, d_{ij}^{l}]\) further w.r.t. each station pair.

In summary, including links of stations (including inter-station trip tendency and distance) is important as simple scalar quantification and local feedbacks of crowds who have limited scopes may ignore the actual trip tendency. Their introduction helps assist the global optimization, and we will further validate their importance and effectiveness via evaluation of real data (Section 5).

### 3.5 Crowdsourced Feedbacks

**Observation.** Crowds are essential to CBSNR, and Fig. 14 visualizes the spatial distribution (“heat-map”) of aggregated crowd feedbacks before BSNR. The warmer color means more feedbacks. We also plot the initial station locations in 2013 (before expansions). From the spatial distribution of crowdsourced feedbacks, we may observe strong sociodemographic factors [17], [18], [29]. For example, many suggestions are made to the central business district and skyline (say, Magnificent Mile) of Chicago, matching intensive commuting needs there. Besides, anticipation also comes from south and west, probably due to student commuter demands around the university campus and introduction of metro stations. We also observe similar patterns in feedbacks of the other two systems. The crowdsourced feedbacks have potential and power in identifying latent factors (qualitatively and quantitatively) for network (re) configuration, and serve as an important supplement to many other GIS databases [38].

Note that the local and dispersed crowds’ feedbacks could not always directly reveal the overall trip tendency connecting the start and the destination, mainly because each individual usually recommends new stations closest to either her/his own work place or residence. Besides, one may not reveal both the start and end of each trip due to his privacy and identity concerns. The global inter-station trip tendency has been modeled in our optimization to account for the above biases or insufficiency.

Pre-processing the crowdsourced data, including filtering those in inaccessible regions, is essential. For example, we have noticed and filtered out some hilarious input locations in Lake Michigan for Divvy. Via comprehensive map boundary and building constraints, we can easily identify those unreasonable feedbacks. As users may vote for more reasonable labels for themselves, and CBikes jointly considers historical usage and geographic constraints, these noisy inputs can be suppressed further.

**Design.** Given Definitions 4 and 5, we consider crowds’ feedbacks in a discretized manner, i.e., we aggregate the number of feedbacks \(f_i\)'s falling into each rectangle grid. Intuitively, the more crowdsourced pin-points go into a grid, the more likely it would be selected. This way, we consider the aggregated feedbacks \(V_i\) for each \(g_i\), and define a measure of vote intensity as a penalty function \(\phi(V_i)\) for our optimization input. A larger \(\phi(V_i)\) due to more votes implies a heavier “penalty” to be minimized by the solver. Specifically, given input \(|V_i|\) votes at \(g_i\), we have

**Definition 8 Deadzone-linear penalty (DLP).** The DLP function with a deadzone width \(\beta \geq 0\) is given by

\[
\phi(V_i) = \begin{cases} 
0 & \text{if } |V_i| \leq \beta; \\
|V_i| - \beta & \text{if } |V_i| > \beta.
\end{cases}
\]

In other words, our DLP de-emphasizes the grids with crowds’ votes less than \(\beta\), mitigating outlier effect, and focuses on others with more support, which is also reasonable in traditional user surveys for BSS expansion [19], [25]. Using a linear \(|V_i| - \beta\), CBikes also mitigates sensitivity towards large but noisy votes than other higher-order penalty functions [3]. After calculating for all \(g_i\)'s, we normalize each \(\phi(V_i)\) \((r \in \{1, \ldots, R\})\) into the range \([0, 1]\).

In summary, as a joint optimization framework, CBikes fuses heterogeneous sources of information and data-driven designs, instead of single-point knowledge input, for final joint decisions, thus mitigating the noisiness of crowd feedbacks. The effectiveness of our proposed information fusion will be validated in Section 5.

### 4 Core Formulation & Methodology

We present the problem formulation to integrate the above designs. We first present the grid matching basics (Section 4.1), and provide the objective functions (Section 4.2). We then discuss the formulation (Section 4.3), followed by semidefinite
programming transformation (Section 4.4). We finally provide a complexity analysis (Section 4.5).

4.1 Station (Re)Placement & Grid Matching

Station (re)placement is more challenging than dock resizing. We convert the BSS (re)placement problem to the problem of estimating affinity (closeness) of each station with predefined geographic grids. Each $S_i$’s location is considered as the weighted average of grid coordinates (Definition 5). Consider $M$ stations are to be (re)placed. Let $h^i_r$ be the weight of grid $r$ in determining $S_i$’s location $l_i$, i.e.,

$$l_i = \sum_{r=1}^{R} h^i_r g_r, \quad \forall i \in \{1, \ldots, M\},$$

where each $h^i_r$ follows normalization and nonnegative constraints,

$$\sum_{r=1}^{R} h^i_r = 1, \quad h^i_r \geq 0, \quad \forall r \in \{1, \ldots, R\}.$$ (13)

For ease of presentation, we define $H$, an $M \times R$ matrix consisting of all $h^i_r$'s. The set of location coordinates of all stations is then

$$L_{M \times 2} = H_{M \times R} G_{R \times 2}.$$ (15)

In our problem formulation, we want to determine the grid weights, as the variables, for station (re)placement.

4.2 Objective Function Design

To incorporate heterogeneous sources of data, we present a novel information-fusion technique in our joint optimization. Specifically, we present the joint difference functions fusing crowds and historical usage, and the cost measures for (re)configuration actions. Combining these leads to our final objective function.

Metric of Joint Difference. To quantify the matching of knowledge fusion, we further design a generic metric, i.e., joint difference of grid matching, denoted as $\Delta^i_r$, for each candidate station $i$ at a grid $r$. Specifically, given $V$ feature metrics $F_v(i, r)$ $\geq 0$ showing the fitness of matching, we may define

$$\Delta^i_r \triangleq \frac{1}{\prod_{v=1}^{V}(1 + F_v(i, r))}.$$ (16)

$F_v(i, r)$’s are derived from available historical usage (Sections 3.1 & 3.3) and crowd feedbacks (Section 3.5), i.e.,

$$\Delta^i_r \triangleq \frac{1}{(1 + U^i_r) \left(1 + \sum_{j=1, j \neq i}^{M} p(i, j) \right) (1 + \phi(V_r))}.$$ (17)

The inverse function in Eq. (17) means that the more historical usage $U^i_r$, total trip tendency $\sum_{j=1, j \neq i}^{M} p(i, j)$ and votes $\phi(V_r)$, the smaller $\Delta^i_r$, and the more favored $g_r$ for $S_i$. It guarantees $0 < \Delta^i_r \leq 1$, and adapts to cases of either with little historical usage or few crowds’ votes (say, any $F_v(i, r) \to 0$).

We also illustrate and visualize the spatial distribution of joint difference $\Delta^i_r$’s in Eq. (18), i.e., “heat map” of fused knowledge. Fig. 15 shows $\Delta^i_r$’s of two station candidates in Divvy (dashed circle: $id = 1$; solid circle: $id = 464$). The warmer the color, the smaller the $\Delta^i_r$, indicating a higher matching potential there for that station.

Note that for further grid differentiation, the joint difference modeling in Eq. (17) is general to be integrated with other external information (other feature metrics $F_c(i, r)$’s) if available, including distance to the central business district, closeness to rail stations and other interesting sociodemographic factors (estate price, income or point of interest number) [25], [38] affecting the station functionality.

Given the joint difference for each station, we further look at the entire network. Let $\Delta$ be an $M \times R$ matrix consisting of all $S_i$’s joint differences. We define an operator $\psi(H, \Delta)$ returning sum of entry-wise products of elements in matrices $H$ and $\Delta$, or formally, the trace (denoted as $\text{Tr}(\cdot)$) of product $HA^T$. Then, the total joint difference of CBSNR estimates and the map of fused knowledge is

$$\psi(H_{M \times R}, \Delta_{M \times R}) \triangleq \text{Tr}(HA^T) \triangleq \sum_{i=1}^{M} \sum_{r=1}^{R} h^i_r \Delta^i_r.$$ (18)

Specifically, the smaller the $\Delta^i_r$, the higher $h^i_r$ assigned to $g_r$, and the more likely $S_i$ is (re)placed there (Eq. (13)), i.e.,

$$h^i_r \geq h^i_q, \quad \text{if} \quad \Delta^i_r \leq \Delta^i_q, \quad \forall r \neq q \in \{1, \ldots, R\}, \quad \forall i.$$ (19)

Cost of Station (re)placement. Considering the feasibility of CBSNR, we integrate the estimates of potential (re)placement cost. Let $c_o \geq 0$ and $c_v \geq 0$ be the costs of adding and removing a station, respectively (customizable w.r.t. each $g_r$ and each $S_i$). The move action is considered as a removal followed by an add. Then, we define the costs of all actions for each $S_i$ at $g_r$, as:

$$\theta^i_r = \begin{cases} 0 & : \text{if no action is imposed;} \\ c_o & : \text{if a new station is added;} \\ c_x & : \text{if an existing station is removed;} \\ c_x + c_o & : \text{if a station is moved to other place}. \end{cases}$$ (20)

Recall that we consider $l_i = \sum_{r=1}^{R} h^i_r g_r$, the weighted average of closely-matched grids. For existing stations, let $\tilde{h}^i_r = 1$ if $S_i$ was at $g_r$ and $\tilde{h}^i_r = 0$ vice versa. For newly-added ones, $\tilde{h}^i_r = 0$, for $\forall r$. Increasing or decreasing $h^i_r$ at grid $r$ implies a higher potential of adding or removing $S_i$. To fit these in our formulation, we characterize these two changes for each cost $\theta^i_r$ as

$$(h^i_r)_o = \max\{h^i_r - \tilde{h}^i_r, 0\},$$

$$(h^i_r)_x = \max\{\tilde{h}^i_r - h^i_r, 0\}.$$ (21)
Then, we set the total cost of (re)placing all \( M \) stations in \( R \) grids as

\[
C^\ast = \sum_{i=1}^{M} \sum_{r=1}^{R} \theta^i_r = \sum_{i=1}^{M} \sum_{r=1}^{R} \left( (h^i_r)_{c_0} + (h^i_r)_{c_0} \right).
\]

(22)

Cost of Dock Resizing. Let \( M' \leq M \) be the number of stations staying at their same locations without (re)placement (moved/removed). Recall in Definition 6, dock resizing considers only the cost of these \( M' \) stations, where each resizing action for an \( S_i \) costs

\[
\eta_i = \begin{cases} 
0 & \text{if dock size is unchanged; } \\
\kappa_i & \text{if dock size is increased by 1; } \\
\kappa_i & \text{if dock size is decreased by 1. } 
\end{cases}
\]

(23)

If a dock needs to be enlarged, we have \( \kappa_i \geq \kappa_i \), and vice versa. Similar to Eq. (21), we define the changes at each station as

\[
(k_i)_j = \max\{k_i - \kappa_i, 0\} \quad (k_i)_j = \max\{\kappa_i - k_i, 0\}.
\]

(24)

We design the cost function to capture the change w.r.t. each station’s location weight assignment in (re)configuration. Similarly, we may set the total cost of dock resizing as

\[
C^d = \sum_{i=1}^{M'} \eta_i = \sum_{i=1}^{M'} \left( (k_i)_j \cdot c_i + (k_i)_j \cdot c_i \right).
\]

(25)

Summary. Fig. 16 summarizes the idea of joint difference “heat map” in CBikes formulation, as formulated in Eq. (17), fusing multiple heterogeneous information sources of usage, trip tendency and votes. Distances derived in Section 3.4 serve as constraints for the grid matching process against the heat map. Given the objective designs (including joint difference and cost) and distance constraints, the core formulation of CBikes determines the final actions, altering the weights \( \{h^i_r\}'s \) which are detailed as follows.

4.3 Problem Formulation

Station (replacement) problem in CBSNR is formulated as: given the crowds’ site suggestions and the historical usage, the objective is to (re)place stations such that total joint difference (in crowdsourced feedbacks and historical usage), as well as the total cost of station (replacement) are jointly minimized.

To accommodate both grid matching and (replacement) cost, we form the final objective as \( \psi(H, \Delta) + \alpha C^\ast \), where \( \alpha > 0 \) is a tunable parameter (we empirically set \( \alpha = 0.5 \)). Formally, we have

\[
\begin{aligned}
\min_{H} & \quad \psi(H, \Delta) + \alpha C^\ast \\
\text{s.t.} & \quad \text{Constraints in Eqs. (11), (14), (15) & (21).}
\end{aligned}
\]

(26)

We further present the formulation of dock resizing. Intuitively, more capacity should be assigned to stations with lower \( \Delta^i \triangleq \sum_{l=1}^{M'} h^i_l \Delta^i_l (i \in \{1, \ldots, M'\}) \), i.e., more crowd supports and historical usage. In other words, \( \kappa_i \geq \kappa_j \) if \( \Delta^i \leq \Delta^j \). In practice, the dock size may not be too large due to space constraint in some city areas. The dock sizing also makes a trade-off between cost and service, where a larger dock size will reduce the time period when a station is out of stock or overstock at the cost of deployment. We may pose an upper limit \( \kappa_{\text{max}} \) for each dock, which may vary with local street environment due to space availability or customization. Specifically, the dock resizing is to minimize the dock resizing cost \( C^d \) and match the frequently-used and popular stations, i.e.,

\[
\begin{aligned}
\min_{\{\kappa_i\}} & \quad C^d \\
\text{s.t.} & \quad \kappa_i \geq \kappa_j, \quad \forall i \neq j, \quad 0 \leq \kappa_i \leq \kappa_{\text{max}},
\end{aligned}
\]

\[
\Delta^i = \sum_{r=1}^{R} h^i_r \Delta^i_r = \sum_{i=1}^{M'} \kappa_i + \sum_{i=M'+1}^{M} \kappa_i = K.
\]

(27)

Total capacity \( K \) can be slightly larger than actual bike number in order to be more resilient to bike flow dynamics.

4.4 SDP Transformation

Note that the lower distance bound, \( \delta_{ij}^2 \leq \|l_i - l_j\|^2 \) in Formulation (26), is a non-convex constraint [3], making its solving rather difficult. To address this difficulty, we introduce a novel semidefinite programming (SDP) technique [3], [9], [20] in order to solve the station (replacement) problem efficiently. Our basic idea is to introduce interim variables representing the station candidate locations, which turn out to be positive semidefinite, and then relax the lower bound constraints via matrix transformation of SDP [20], making it easier to be solved in polynomial time by interior-point algorithms [3], [20].

Mathematically, we first define an indicator vector \( (o_{ij})_{M \times 1} \) with \( M \) elements, among which the \( i'th \) element is 1, the \( j'th \) is -1 and all others are 0. Let \( \delta_{ij}^2 = (l_i - l_j)^T (l_i - l_j) \) be the resultant distance (squared) from predictions of \( S_i \) and \( S_j \), and we may further have

\[
\delta_{ij}^2 = o_{ij}^T L L^T o_{ij}, \quad \forall i \neq j, (i, j) \in N.
\]

(28)

We then introduce a transition matrix \( Z \in \mathbb{R}^{M \times M} \) as \( Z = L L^T \), or

\[
Z - L L^T = 0.
\]

(29)

Then, we rewrite the aforementioned bound constraint into

\[
\delta_{ij}^2 \leq o_{ij}^T Z o_{ij} \leq \delta_{ij}^2.
\]

(30)
Next we relax Eq. (29) into a semidefinite form [3], i.e.,

\[
Z - LL^T \succeq 0.
\]

(31)

We aim at transforming Eq. (29) into one with linear matrix inequality (LMI) [3], [20] which turns out to be convex and solvable. Therefore, we introduce a block matrix form called Schur complement [3] for transformation, which is formally defined as follows.

**Definition 9 Schur Complement.** Let \( A \) be a matrix which is partitioned into four matrix blocks \( B, C, D \) and \( E \), i.e.,

\[
A = \begin{bmatrix} B & C \\ D & E \end{bmatrix}.
\]

(32)

where \( B \) and \( E \) are symmetric and nonsingular matrices. Then, Schur complement of block \( E \) in matrix \( A \), denoted as \( A/E \), is given by

\[
A/E = B - CE^{-1}D.
\]

(33)

According to related theory of matrices [3], we have \( A \succeq 0 \) if \( A/E \succeq 0 \). Recall that \( Z - LI_{2\times2}L^T = I_{2\times2}/Z \succeq 0 \) (Eq. (31)), where \( I_{2\times2} \) is a \( 2 \times 2 \) diagonal unit matrix. We then have its \((M+2)\times(M+2)\) LMI form:

\[
\begin{bmatrix}
Z_{M \times M} & L_{M \times 2} \\
(L^T)_{2 \times M} & I_{2 \times 2}
\end{bmatrix} \succeq 0.
\]

(34)

This way, a semidefinite programming solver [3], [20] can be applied upon the LMI, and the non-convex problem can be solved efficiently and effectively. In summary, the final formulation is given by

\[
\arg \min_H \psi(H, \Delta) + aC^*, \\
\text{s.t.} \text{ Constraints in Eqs. (14), (15), (21), (30), & (34).}
\]

(35)

Then, CBikes rounds each station estimation \( l_i \) to its nearest grid. Service providers may customize and enforce extra constraints (some inaccessible area, e.g., \( h_{i}^l = 0 \), or region boundary, e.g., \( A \cdot lon_i + B \cdot lat_i + C \geq 0 \)) given geographical areas where a dock is not supposed to be deployed (say, a building or a river).

In practice, SDP relaxation renders Eq. (31) a slightly flexible design instead of an over-rigid one, helping adapt to more sophisticated network structures underneath. Other refinements, if needed, can be applied to fine-tune those relaxed distance bounds. One may also check on over-relaxed pairs and adjust using the gradient descent approach [3] to re-satisfy their constraints. We observed only a very small proportion (say, usually less than 1.85 percent) out of all station pairs need a cosmetic refinement, making our SDP design applicable in most cases.

### 4.5 Complexity Analysis

We briefly analyze the computational complexity of CBikes. Given \( M \) stations and total \( N_f \) feedbacks, finding \( \Delta_i^s \) of all \( R \) grids takes \( O(N_f + MR) \). With \( M \) stations and \( R \) grids, the complexity of SDP is \( O(M^2R^3) \) [3], [20], and the total sums to \( O(N_f + M^3R^4) \) for CBikes.

Further computation reductions can be made in several ways. For example, for each \( S_i \), out of all grids we may only consider the top several location candidates, which have lower joint differences \( \Delta_i^s \), and locally search its potentially-nearby neighbors [2], [18] for fewer mutual distance constraints in the optimization. Specifically, for each BSS station \( i \in \{1, \ldots, R\} \), we find the top \( R' \) \((R' < R)\) grids as the pruned search scope.

Using the above methods, \( R \) and constraints (say, Eqs. (14), (19), and (30)) can be reduced significantly, thus achieving better computational efficiency.

### 5 Experimental Evaluation

We first present the evaluation setups in Section 5.1, and then illustrate the effects of different system settings in Section 5.2, followed by the experimental results in Section 5.3.

#### 5.1 Evaluation Setups & Schemes Compared

We compare CBikes with the following schemes in BSNR design:

- **BSNR-w/o-Cost**: which greedily considers crowds and historical usage, without considering the cost for CBSNR.
- **BSNR-w/o-Crow**: which focuses on only historical usage [4], [18], without crowd feedbacks, to (re)place or resize the BSS stations.
- **BSNR-w/o-Hist**: which greedily considers only crowdsourced feedbacks without historical usage, to (re)configure the stations.
- **BSNR-w/o-Tend**: which considers no inter-station trip tendency, and independently (re)configures each station [16], [34].
- **BSNR-w/o-Dist**: which does not consider any distance bound constraint [14].
- **HEU**: a heuristic scheme, instead of joint optimization, adopted by some BSS providers (e.g., Capital Bikeshare) Site candidates are first filtered by some heuristic criteria \(^7\) (like utility). Top-ranked candidates are selected and further fine-grained.
- **RAND**: which randomly (re)places the BSS stations into grids and resizes them without using any design metrics in Section 3.
- **Previous CBikes** [10] (denoted as CBikes-1.0): which is the previously published conference version without considering the usage estimation.

We evaluate the above algorithms based on the datasets (i.e., Divvy, Nice Ride and Metro Bike) described in Section 2.3. For BSNR-w/o-Cost, BSNR-w/o-Crow, BSNR-w/o-Hist, BSNR-w/o-Tend, BSNR-w/o-Dist, HEU and RAND, we adopt the estimated usage at unexplored grids/regions in order to evaluate performance of other setups. We compare the station networks before and after each CBSNR phase, i.e., \( G \) and \( G_i \), including each station’s status, i.e., \( S_i = (lat_i, lon_i, \kappa_i) \) against \( S_i = (lat_i, lon_i, \kappa_i) \). We analyze (re)placement of stations and their capacity change. With the timestamps \( t_m \) in Definition 4), crowdsourced feedbacks before this CBSNR (or

---

between two consecutive expansions, if any) are used as optimization inputs.

At each CBSNR phase, we use the following evaluation metrics:

- **Accuracy, precision, f-measure & recall:** We compare the difference with the ground-truth station distribution. Specifically, we determine accuracy by checking whether each station is matched with its ground-truth grid. We measure the latter three well-known metrics of binary prediction w.r.t. the grids, i.e., a value 1 (0) represents that a station is (not) placed inside a grid.

- **(Re)configuration cost:** We compare the costs of all schemes, i.e., station (re)placement ($C'$) and dock resizing ($C''$). For the purpose of reference, we also show the ground-truth (GT) costs derived from the actual (re)configuration done by service providers.

- **Mean absolute error (MAE) & mean squared error (MSE):** differences between predicted dock size ($\tilde{k}_i$) and ground-truth ($k_i$).

All computation is done on a desktop of Windows 10, Intel Core i7-6700, 32 GB RAM and Nvidia GTX 1050Ti. Unless otherwise stated, the default parameter values are set as follows. For each CBSNR phase, by analyzing trips and stations before it happens, we empirically set the $[d_{ij}, \eta_{ij}]$ as described in Section 3.4, $\alpha = 0.5$ and $\beta = 10$. We have empirically observed that a large $\beta$ results in fewer crowdsourced feedbacks included in the problem formulation, and a small one introduces more noisy feedbacks (detailed results are left due to space limit). Taking into account the above trade-offs, in our studies, we empirically set above $\beta$, and the setting leads to reasonable performance of CBikes upon the crowdsourced inputs.

To estimate bike usage at unexplored grids, we apply dropout between fully-connected (FC) layers and batch normalization on the data to mitigate overfitting and enhance convergence; the Adam optimizer is used and the learning rate is set to 0.01; for each CBSNR, we leave 10 percent of the grids for the validation of results, and train the neural network model using the rest (90 percent) of the data (feature vectors and the bike usage of the grids with stations); the neural network structure implemented with Tensorflow and Python is: input layer $\rightarrow$ FC(16) $\rightarrow$ FC(128) $\rightarrow$ FC(16) $\rightarrow$ output layer with tanh activation, where $\text{dim}_{\text{in}}$ in FC(\text{dim}) (with relu activation) represents the number of dimensions inside the fully-connected dense layer. The number of epochs is set to 2,000. The input feature vectors and the output usage data are min-max normalized. For each city, we find $P_1 = 5$ nearest neighbors and $P_2 = 20$ types of POIs for prediction of usage potential (the parameter selection is based on the validation process upon the dataset different from the test one).

In our experimental studies, as a summary, we observe that the mean absolute percentage errors (MAPEs) for Divvy (446 unexplored grids), Nice Ride (112 unexplored grids) and Metro Ride (105 unexplored grids) are 0.1585, 0.2199, and 0.1942, respectively. Note that we are leveraging the estimated usage of these unexplored grids/regions to differentiate them for CBSNR, and the estimation performance suffices to support station (re)placement decisions.

To balance computation efficiency and (re)placement granularity, we set a $90 \times 90$ grid mesh (each grid is $0.23 \times 0.40$ km$^2$) for Divvy (Chicago), with a bounding box $[-87.80^\circ W, -87.55^\circ W; 41.74^\circ N, 42.06^\circ N]$. For Nice Ride (Twin Cities), we use a $60 \times 60$ grid mesh (each is $0.32 \times 0.26$ km$^2$), within a box $[-93.32^\circ W, -93.08^\circ W; 44.89^\circ N, 45.03^\circ N]$. As LA county is much larger, a $120 \times 120$ mesh (each is $0.29 \times 0.42$ km$^2$) comes with a box $[-118.49^\circ W, -118.12^\circ W; 33.71^\circ N, 34.17^\circ N]$ for Metro Bike. Based on the existing public market analysis$^7$, we consider $c_s = 80$, $c_d = 100$ (station (re)placement) and $c_l = c_d = 10$ (dock resizing).

5.2 Evaluation on System Settings
We first evaluate CBikes’ performance while varying its important components and settings. Note that we set the parameters based only on historical data of periods prior to each CBSNR for bias-free evaluation. Taking Divvy in Chicago as a representative example, we evaluate CBikes’ sensitivity to the following different important parameters.

**Local search scope & number of neighbors** (Section 4.5): Fig. 17 shows the effect of local search scope in reducing the computation complexity of CBikes. We conducted experimental studies on the reconfiguration of the Divvy system in 2015 when the number of stations increased from 300 to 474. As more grids are involved in the local grid search, the higher (re)placement granularity from CBikes is expected. However, the performance begins to converge after adding a few more neighbors and the computation overhead also increases. Therefore, we select 10 neighbors by default for reasonably efficient deployment.

**Density of grids** (Section 5.1): We show in Fig. 18 CBikes’ sensitivity to the density of grids in terms of accuracy, precision, f-measure and recall for the Divvy dataset. Clearly, the denser grids yield more fine-grained estimation results,

---

Fig. 17. (Re)Placement performance versus numbers of neighbors (Divvy).

Fig. 18. (Re)Placement performance versus grid sizes (Divvy).
may be easily affected by noisy logs.

HEU outperforms BSNR-w/o-Dist on each station. It achieves higher accuracy (often by 5–11.2% (heuristic) adjusts stations without joint optimization and case of extensive expansion, causing larger variations. Lacking crowdsourced feedbacks, and suffers much worse and varied performance. Without mutual constraints, BSNR-w/o-Dist may get similar grid coverage, but lower matching accuracy is based on station index, while others are for performance (accuracy, precision, f-measure and recall) in terms of station (re)placement (Tables 1, 2, and 3) and dock resizing (Tables 4, 5, and 6). Compared to our earlier results reported in [10], CBike achieves higher accuracy (often by 6.58–11.21 percent) and lower reconfiguration cost (often by > 40%) thanks to its more external and predicted knowledge. Considering the scale of BSS networks with hundreds of stations, CBike can help the city planner significantly reduce the planning cost. Overall, the estimated usage improves the station (re)placement more than the dock resizing, mainly due to more location-dependent designs in the (re)placement problem. Since the current model of CBike outperforms that in [10], unless otherwise stated, we will henceforth focus on evaluating the former.

Station (re)placement. We first show the (re)placement performance (accuracy, precision, f-measure and recall) in Tables 1, 2 and 3. Each metric is provided with the mean and 75th/25th percentiles of all CBSNR phases. Note that accuracy is based on station index, while others are for BSS reconfiguration. To balance accuracy, overhead and granularity, we set their granularity may not represent practical BSS reconfiguration. On the other hand, sparser grids are easier to predict while their granularity may not represent practical BSS reconfiguration. To balance accuracy, overhead and granularity, we set their granularity may not represent practical BSS reconfiguration.

5.3 Evaluation on Datasets

Overview. With additional knowledge of estimated usage at unexplored grids/regions, CBike outperforms CBike-1.0 in terms of station (re)placement (Tables 1, 2, and 3) and dock resizing (Tables 4, 5, and 6). Compared to our earlier results reported in [10], CBike achieves higher accuracy (often by 6.58–11.21 percent) and lower reconfiguration cost (often by > 40%) thanks to its more external and predicted knowledge. Considering the scale of BSS networks with hundreds of stations, CBike can help the city planner significantly reduce the planning cost. Overall, the estimated usage improves the station (re)placement more than the dock resizing, mainly due to more location-dependent designs in the (re)placement problem. Since the current model of CBike outperforms that in [10], unless otherwise stated, we will henceforth focus on evaluating the former.

Station (re)placement. We first show the (re)placement performance (accuracy, precision, f-measure and recall) in Tables 1, 2 and 3. Each metric is provided with the mean and 75th/25th percentiles of all CBSNR phases. Note that accuracy is based on station index, while others are for BSS reconfiguration. To balance accuracy, overhead and granularity, we set their granularity may not represent practical BSS reconfiguration. On the other hand, sparser grids are easier to predict while their granularity may not represent practical BSS reconfiguration.

5.3 Evaluation on Datasets

Overview. With additional knowledge of estimated usage at unexplored grids/regions, CBike outperforms CBike-1.0 in terms of station (re)placement (Tables 1, 2, and 3) and dock resizing (Tables 4, 5, and 6). Compared to our earlier results reported in [10], CBike achieves higher accuracy (often by 6.58–11.21 percent) and lower reconfiguration cost (often by > 40%) thanks to its more external and predicted knowledge. Considering the scale of BSS networks with hundreds of stations, CBike can help the city planner significantly reduce the planning cost. Overall, the estimated usage improves the station (re)placement more than the dock resizing, mainly due to more location-dependent designs in the (re)placement problem.

at the cost of longer computation time and lower accuracy, especially above a certain grid density (say, after 90 × 90). On the other hand, sparser grids are easier to predict while their granularity may not represent practical BSS reconfiguration.

5.3 Evaluation on Datasets

Overview. With additional knowledge of estimated usage at unexplored grids/regions, CBike outperforms CBike-1.0 in terms of station (re)placement (Tables 1, 2, and 3) and dock resizing (Tables 4, 5, and 6). Compared to our earlier results reported in [10], CBike achieves higher accuracy (often by 6.58–11.21 percent) and lower reconfiguration cost (often by > 40%) thanks to its more external and predicted knowledge. Considering the scale of BSS networks with hundreds of stations, CBike can help the city planner significantly reduce the planning cost. Overall, the estimated usage improves the station (re)placement more than the dock resizing, mainly due to more location-dependent designs in the (re)placement problem. Since the current model of CBike outperforms that in [10], unless otherwise stated, we will henceforth focus on evaluating the former.

Station (re)placement. We first show the (re)placement performance (accuracy, precision, f-measure and recall) in Tables 1, 2 and 3. Each metric is provided with the mean and 75th/25th percentiles of all CBSNR phases. Note that accuracy is based on station index, while others are for BSS reconfiguration. To balance accuracy, overhead and granularity, we set their granularity may not represent practical BSS reconfiguration. On the other hand, sparser grids are easier to predict while their granularity may not represent practical BSS reconfiguration.
global pictures, and thus more post-processing is required before better results can be achieved. We also note that due to additional estimated usage at unexplored grids, the performance metrics of the schemes improve from the ones without estimated usage [10]. In contrast, with joint information fusion and optimization CBikes outperforms others.

Due to a much larger volume of trip data and denser network with more stations, CBikes in Chicago is optimized better and slightly outperforms those in other two cities. Considering the coupling of users and stations (trip tendency and distance bounds) makes CBikes outperform BSNR-w/o-Tend and BSNR-w/o-Dist. Divvy may witness stronger effect of inter-station trip tendency (more commute and recreational trips) and there is a slightly larger gap between CBikes and BSNR-w/o-Tend. Besides, as more CBSNR phases (total 5) are involved in Twin Cities, all schemes experience more performance variations than in other cases.

Tables 1, 2 and 3 also summarize the total (re)placement costs. Clearly, one may expect a huge cost to be incurred by BSNR-w/o-Cost. With more information fused, CBikes achieves much lower costs and outperforms others. The ground-truth station (re)placement costs (mean, 75 percent) for Divvy, Nice Ride and Metro Ride are respectively (2.19, 2.52, 1.96), (0.62, 0.72, 0.53) and (1.38, 1.49, 0.0). We can also see that its differences with ground-truth (actual (re) placement costs) are also much smaller.

Dock Resizing. Due to space limit and similarity of results, we focus on dock resizing of Divvy and Nice Ride here. Tables 4 and 5 show the different schemes in terms of resizing MAEs and MSEs w.r.t. ground-truth $k$'s in Chicago and Twin Cities. Large resizing error may lead to underutilization or underprovisioning of docks, causing waste and imbalance of BSS resources. CBikes is shown to achieve much lower errors (usually more than 20 percent improvement) than other schemes. Overall, dock resizing may be easier in Chicago than in Twin Cities due to more trip data and better optimized (re)placement results.

Compared to Divvy, historical usage at Nice Ride is more important in dock resizing than crowd popularity. Due to a sparse network at Nice Ride, most crowds’ feedbacks focus on the issues of adapting coverage or density, without paying attention to the resizing of existing stations. Thus, without sufficient historical usage information, BSNR-w/o-Hist could not effectively determine the importance of each station’s capacity, and hence larger error occurs to it at Nice Ride than BSNR-w/o-Crowd and others.

Table 6 summarizes the dock resizing costs ($\log_{10}(C^i)$). The ground-truth costs (mean, 75 and 25 percent) for the Divvy and Nice Ride are respectively [3.29, 3.49, 3.06] and [2.94, 3.13, 2.78]. Note that similar costs may occur when wrong subsets of docks are resized at a similar scale. With better accuracy due to more comprehensive information fusion and lower adjustment cost (often by half an order of magnitude), CBikes helps effectively adapt to bike demands with better feasibility.

(Re)Configuration Visualization & Computation Overhead. We visualize (re)configuration prediction and ground-truth results in Figs. 19, 20, and 21 for Chicago (re)configuration in 2016), Twin Cities (re)configuration in 2015) and LA County (re)configuration in 2017). One can see that the predictions via crowdsourced information fusion and joint optimization markedly resemble the actual values. In particular, we show the downtown replacement results without and with usage estimation in Figs. 22 and 23. Thanks to the additional knowledge of the downtown neighborhoods, our new scheme achieves better matching results compared with the previous version.

In terms of computation, the optimization time w.r.t. datasets of Divvy, Nice Ride and Metro Bike are 93.71s (due to much more stations), 19.7s and 7.27s, which are suitable for periodic (monthly or annual) bike station network (re) configuration.

6 DISCUSSIONS

Network Shrinkage: As most existing BSS systems are growing in recent years, our evaluation data in hand mainly contains expansions, and does not include any (re)configuration cases with only shrinkage. However, the data we
studied includes removed/moved stations (say, around 21.25 percent of all stations). Our model is general enough to accommodate both expansion and shrinkage of BSN, and can achieve good accuracy.

Incorporating Other Information. Due to resource limit, a myriad of other factors, such as demographic distribution and city management regulation [17], [18], [29], may not be well considered in our current prototype. Their absence might also account for the discrepancy from actual results. However, as a generic information fusion framework, CBikes can easily integrate them if and when given. Also, note that we focused on urban-level BSNR, reducing the initial search scope and facilitating decision-making on management of BSNs. Given our results as a reference, secondary fine-grained adjustments of dock locations inside grids may be made subject to various constraints, including bike accessibility, user visibility and space compatibility, which are orthogonal to our focus.

Our studies focus on replacement and resizing cost for long-term decisions of the BSS providers, while operational cost including rebalancing actions and maintenance usually results from short-term and spatio-temporally dynamic factors (including weather and traffic conditions). Among the candidates with similar demands, a station with lower operational cost after (re)placement is often preferred. Despite the lack of dynamic operation data from the service providers, our formulation can be extended to such additional knowledge if available.

The demand from the bike transition is also affected by the inventory status, including the case of invisible demand loss due to a station’s out-of-stock condition, and consequent substitution effect of its neighbors. While our comprehensive information fusion takes into account the ride preferences, further investigation should be conducted upon the above issues for better predictability of demands.

Further Denoising. Large error in using crowds’ feedbacks only (BSNR-w/o-Hist in Section 5) indicates the severity of “noisy” crowdsourcing. CBikes can exploit many state-of-the-art approaches [13], [19], [26] to filter the comments or incentivize better suggestions from the crowds. Besides, service providers periodically conduct formal panels or seminars where citizen representatives could discuss BSNR. One may design weighting schemes to assess the quality of various feedbacks for better accuracy.

Dynamic Traffic Prediction. Recently, researchers have proposed highly accurate traffic prediction based on deep learning [30], [35]. Yao et al. [30] proposed the meta learning to learn the bike flows from multiple different cities. Wang et al. [23] studied an entropy-based prediction model for future bike usage. While CBikes can be easily integrated with theirs for more accurate and prompt bike traffic prediction [15], [23], our coarse-grained study focuses on qualitative

Fig. 19. Visualization in Chicago (765.6 km²).

Fig. 20. Matching visualization in Twin Cities (304.87 km²).

Fig. 21. Matching visualization in Los Angeles County (1,754.5 km²).

Fig. 19. Station (re)placement in Down Town, Minneapolis without usage estimation.

Fig. 22. Station (re)placement in Down Town, Minneapolis with usage estimation.

Fig. 23. Station (re)placement in Down Town, Minneapolis with usage estimation.
inference of the importance of new city grids in terms of potential bike usage over a long period. In future, we would like to study the prediction of dynamic traffic flows for smart transportation systems.

Acceleration. While CBikes leverages a centralized structure, it can be easily extended to the distributed designs. Furthermore, due to the last-mile nature of the bike sharing, we can cluster the bike stations into many clusters, which are more connected (say, in terms of trip volumes and time correlations) within each cluster than across any two of them, and conduct the cluster-wise computation to reduce centralized computation. Parallelization and GPU can be easily applied, which is outside the scope of this paper and will be part of our future work.

7 RELATED WORK

We briefly review the related work in the areas of urban computing, station placement and bike sharing systems.

Urban Computing & Information Fusion. Urban computing [37] aims to improve social life quality under the trend of speedy urbanization. With faster computing, smarter IoTs and more sensing data, many urban transportation problems have been redefined intelligently and efficiently. CBikes serves as a novel cross-domain knowledge fusion technique [36], unleashing the data-driven and crowdsourcing power to look at traditional site (re)configuration for emerging bike sharing [6], [7], [11], [28].

Site Placement & Expansion. Due to the recent boom of intelligent transportation, site placement, including gas stations [32], ambulance points [37], and electric vehicle charging docks [14] has been investigated to improve their social and business values.

Note that our work is different from the problems of placing stores [27], gas or electric charging stations [14], since we are given crowdsourced comments and usage statistics from already-deployed stations to (re)configure the BSS network, thus making their initial station placement not directly applicable to our problem. Our joint optimization and crowdsourced fusion are also complementary to emerging urban dynamics [31] and functional zone inference [18], and their studies can be integrated with ours for further refinement of results. Unlike others estimating geographical dependencies of real estate [8], CBikes considers users’ trip tendency (pick-up/drop-off) between the bike stations.

Bike Sharing Systems & Services. Recent popularity of BSS has triggered many interesting studies, such as mobility and demand prediction [18], [24], [29], [33], station re-balancing [17], lane planning [1], trip recommendation and station deployment [16], [18]. However, few of state-of-the-art studies considered optimizing the (re)configuration of existing BSS network with crowdsourced knowledge. Orthogonal to the important spatial-temporal modeling for real-time bike demand prediction (including dynamic geographical, meteorological or seasonal factors) [17], [18], [29], CBikes focuses on fusing long-term batched station usage [25], [38] with aggregated crowdsourced feedbacks, for periodic network (re)configurations. Note that our (re)configuration can be done monthly, seasonally or annually subject to the urbanization process, profit, cost and the service provider’s own customization.

Many external factors may influence the success of (re)configuration [33], [38], including human-built facilities (quality/availability), natural environments (like topography, season or weather [29]), socio-economic or psychological considerations (say, social norms or habits), and utility (cost and travel time). Though it is very challenging to design a complete model, incorporating historical spatial-temporal usages, large-scale crowdsourced preferences and refined cost metric would be a good way to accommodate these factors.

In contrast to recent approaches to BSS deployment [18], [34], we propose a generic optimization framework that accommodates both network expansion and reduction using data-driven designs and novel semidefinite programming [3]. CBikes adopts a flexible formulation fusing crowdsourced knowledge with historical usage statistics jointly, and accounts for interactions of users and stations, thus adapting much better to complex station correlations.

Our study is also orthogonal to emerging station-free BSS systems [1], [22]. CBikes can be used for station-free BSS if each parked bike is considered a “dock-less station”. However, as unregulated parking may still prevent its wide acceptance by social-norm, we focus on station-based bike sharing systems in this paper.

8 CONCLUSION

BSS network (re)configuration – i.e., station (re)placement and dock resizing – has become very important for many BSS providers. We have proposed a novel optimization framework, CBikes, to (re)configure bike station networks with crowdsourced station suggestions. A comprehensive data analysis first derives inter-station trip tendency and distance constraints. Crowds’ feedbacks, historical usage, costs and designs are then fused into a joint optimization formulation. We have also modeled the spatial distributions of station usage to account for, and estimate the unexplored regions without historical usage information. We further leverage SDP transformation to solve the nonconvex (re)placement problem efficiently and effectively. Extensive experiments with 3 premium BSS systems, supported by related crowds’ feedbacks, have validated the accuracy and effectiveness of CBikes.

ACKNOWLEDGMENTS

This work was supported in part by the DGIST Global Research Laboratory Program through NRF funded by MSIP of Korea.

REFERENCES


Suining He received the PhD degree from the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology (HKUST). He is currently working as an assistant professor with the University of Connecticut, Storrs CT. He worked as a postdoctoral research fellow with the Real-Time Computing Lab (RTCL), the Department of Electrical Engineering and Computer Science, the University of Michigan, Ann Arbor, MI. He is a Google PhD fellow, 2015. His research interest includes smart transportation, data mining, ubiquitous and mobile computing.

Kang G. Shin (Fellow, IEEE) is the Kevin & Nancy O’Conner professor of computer science in the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor. His current research focuses on QoS-sensitive computing and networking as well as on embedded real-time and cyber-physical systems, such as autonomous vehicles. He has supervised the completion of 85 PhDs, and authored/coauthored close to 1,000 technical articles, a textbook, and about 60 patents or invention disclosures, and received numerous awards including the 2019 IEEE Computer Society Award for Outstanding Research in Privacy Enhancing Technologies, and the best paper awards from the 2011 ACM International Conference on Mobile Computing and Networking (MobiCom 2011), the 2011 IEEE International Conference on Autonomic Computing, the 2010 and 2000 USENIX Annual Technical Conferences, as well as the 2003 IEEE Communications Society William R. Bennett Prize Paper Award and the 1987 Outstanding IEEE Transactions on Autonomic Computing Conference Achievement Award. He has been a member of about 20 institutional awards, including the Research Excellence Award in 1989, Outstanding Achievement Award in 1999, distinguished faculty Achievement Award in 2001, and Stephen Attwood Award in 2004 from The University of Michigan (the highest honor bestowed to Michigan Engineering faculty); a Distinguished Alumni Award of the College of Engineering, Seoul National University in 2002; 2003 IEEE RTC Technical Achievement Award; and 2006 Ho-Am Prize in Engineering (the highest honor bestowed to Korean-origin engineers). He has chaired Michigan Computer Science and Engineering Division for three years starting 1991, and also several major conferences, including 2009 ACM Mobicom, 2008 IEEE SECON, 2005 ACM/USENIX MobiSys, 2000 IEEE RTAS, and 1987 IEEE RTSS. He is the fellow of the ACM. He has also served or is serving on numerous government committees, such as the US NSF Cyber-Physical Systems Executive Committee and the Korean Government R&D Strategy Advisory Committee. He has also helped founding a couple of startups and is currently serving as an executive advisor for Samsung Research.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.