Polynomial-time Algorithm for Dock Re-allocation Problem in Bike Sharing System

Akiyoshi Shioura[1,a)]

Abstract: In this paper we consider a nonlinear integer programming problem for re-allocation of dock capacity in a bike sharing system discussed by Freund et al. (2017). Our main result is to show that the re-allocation problem can be solved in polynomial time. Our approach is to decompose the problem into two subproblems by using a new parameter. We first show that the two subproblems have the same structure as a relaxation of the dock re-allocation problem, and can be solved by a proximity-scaling algorithm that runs in polynomial time. We then prove that the two subproblems have convexity with respect to the parameter, which makes it possible to find an optimal value of the parameter by binary search.

1. Introduction

We consider a nonlinear integer programming problem for re-allocation of dock-capacity in a bike sharing system discussed by Freund, Henderson, and Shmoys [1]. In a bike sharing system, many bike stations are located around a city so that users can rent and return bikes there. Each bike station has several docks and bikes; some docks are equipped with bikes, and the other docks are open so that users can return bikes at the station. The numbers of docks with bike and of open docks change as time passes, and it is possible that some users cannot rent or return a bike at a station due to the shortage of bikes or open docks, and in such situation users feel dissatisfied. To reduce users' dissatisfaction, operators of a bike sharing system need to re-allocate docks (and bikes) among bike stations appropriately. Change to a new allocation, however, requires the movement of docks and bikes, which yields some amount of cost. Therefore, it is desirable that a new allocation is not so different from the current allocation. Hence, the task of operators in a bike sharing system is to minimize users' dissatisfaction by changing the allocation of docks, while bounding the number of docks to be moved in the re-allocation.

This problem, which we refer to as the *dock re-allocation problem*, is discussed by Freund et al. [1] and formulated as follows:

$$\text{(DR)} \quad \left| \begin{array}{ll} \text{Minimize} & c(d,b) \equiv \sum_{i=1}^{n} c_i(d(i),b(i)) \\ \text{subject to} & d(N) + b(N) = D + B, \\ & b(N) \le B, \\ & \|(d+b) - (\bar{d}+\bar{b})\|_1 \le 2\gamma, \\ & \ell \le d + b \le u,\ d, b \in \mathbb{Z}^n_+. \end{array} \right.$$

Here, $n \in \mathbb{Z}$ denotes the number of bike stations and put $N = \{1, 2, \ldots, n\}$. We denote by $d = (d(1), d(2), \cdots, d(n)) \in \mathbb{Z}^n_+$ and $b = (b(1), b(2), \cdots, b(n)) \in \mathbb{Z}^n_+$, respectively, the vectors of decision variables representing the numbers of docks with bike and of open docks allocated at the stations. The expected number of dissatisfied users at the station i is represented by a function $c_i : \mathbb{Z}^2_+ \to \mathbb{R}$ in variables $d(i)$ and $b(i)$, and shown to have the property of *multimodularity* (see Section 2 for the definition).

The first constraint in (DR) means that the total number of docks (i.e., docks with bike and open docks) is equal to a fixed constant $D + B$. The second constraint gives an upper bound for the total number of docks with bike. The third constraint, given in the form of L1-distance constraint, means that the difference between the current and the new allocations of docks should be small, where $\bar{d}(i)$ and $\bar{b}(i)$ denote, respectively, the numbers of docks with bike and of open docks at the station i in the current allocation. In addition, the number of docks $d(i) + b(i)$ at each station i should be between lower and upper bounds $[\ell(i), u(i)]$, as represented by the fourth constraint.

For the problem (DR), Freund et al. [1] propose a steepest descent (or greedy) algorithm that repeatedly update a constant number of variables by ± 1, and prove by using the multimodularity of the objective function that the algorithm finds an optimal solution of (DR) in at most γ iterations. Hence, the problem (DR) can be solved in pseudo-polynomial time, while it is not known so far whether (DR) can be solved in polynomial time.

The main aim of this paper is to develop a polynomial-time algorithm for (DR). For this, the following relaxation problem of (DR) obtained by removing the L_1-distance constraint plays an important role:

[1] Department of Industrial Engineering and Economics, Tokyo Institute of Technology, Tokyo 152-8550, Japan
[a)] shioura.a.aa@m.titech.ac.jp

$$\text{(DA)}\quad \left|\begin{array}{ll} \text{Minimize} & c(d,b) \\ \text{subject to} & d(N)+b(N)=D+B, \\ & b(N)\le B, \\ & \ell \le d+b \le u,\ d,b\in\mathbb{Z}_+^n. \end{array}\right.$$

We first show that the problem (DA) can be solved in $O(n \log n \log((D+B)/n))$ time by a proximity-scaling algorithm. In a proximity-scaling algorithm, we deal with the "scaled" problem (DA(λ)), which is the problem (DA) with an additional assumption that $d(i)$ and $b(i)$ are multiples of the scaling parameter $\lambda \in \mathbb{Z}_{++}$ (see Section 3 for more precise definition of (DA(λ))). By the definition of multimodularity, it is easy to see that the problem (DA) is closed under the scaling operation. Hence, the steepest descent algorithm for (DA) can be also applied to the scaled problem (DA(λ)).

Our proximity-scaling algorithm consists of several scaling phase and in each of the scaling phase, the problem (DA(λ)) is solved for some λ. The parameter λ is set to a large number in the first phase, then it is gradually reduced, and finally, it is set to 1 to obtain an optimal solution of the original problem (DA). In each scaling phase, we apply the steepest descent algorithm for (DA) to (DA(λ)), where the solution obtained in the previous phase is used as an initial solution. To bound the number of iterations of the steepest descent algorithm, we show a "proximity" theorem, stating that for an optimal solution of a scaled problem (DA(λ)), there exists an optimal solution of the origianl problem (DA).

To obtain a polynomial-time algorithm for our original problem (DR), we show that the L1-distance constraint in (DR) can be replaced with a simple linear constraint by using an optimal solution of the problem (DA); we denote by (DR-L) the problem obtained by this replacement. Using a new parameter, we decompose the problem (DR-L) into two independent subproblems, both of which have the same structure as the problem (DA) and therefore can be solved efficiently. We show that the "best" value of the parameter can be determined by application of binary search. As a result, we obtain a polynomial-time algorithm for (DR) that runs in $O(n \log n \log((D+B)/n) \log B)$ time.

2. Preliminaries

Throughout the paper, let n be a positive integer with $n \ge 2$ and put $N = \{1, 2, \ldots, n\}$. We denote by $\mathbb{R}$ the sets of real numbers, and by $\mathbb{Z}$ (resp., by $\mathbb{Z}_+$) the sets of integers (resp., nonnegative integers); $\mathbb{Z}_{++}$ denotes the set of positive integers.

Let $x = (x(1), x(2), \ldots, x(n)) \in \mathbb{R}^n$ be a vector. We denote $\mathrm{supp}^+(x) = \{i \in N \mid x(i) > 0\}$ and $\mathrm{supp}^-(x) = \{i \in N \mid x(i) < 0\}$. For a subset $Y \subseteq N$, we denote $x(Y) = \sum_{i\in Y} x(i)$. We define $\|x\|_1 = \sum_{i\in N} |x(i)|$ and $\|x\|_\infty = \max_{i\in N} |x(i)|$.

We define $\mathbf{0} = (0, 0, \ldots, 0) \in \mathbb{Z}^n$. For $j \in N$, we denote by $\chi_j \in \{0,1\}^n$ the characteristic vector of j, i.e., $\chi_j(i) = 1$ if $i = j$ and $\chi_j(i) = 0$ otherwise. Inequality $x \le y$ for vectors $x, y \in \mathbb{R}^n$ means component-wise inequality $x(i) \le y(i)$ for all $i \in N$. For vectors $x, y \in \mathbb{R}^n$ and a positive integer, we write $x \equiv y \bmod \lambda$ if $x(i) \equiv y(i) \bmod \lambda$ for every $i \in N$.

We then explain the concept of multimodularity. A function $\varphi : \mathbb{Z}_+^2 \to \mathbb{R}$ in two variables is called *multimodular* if it satisfies the following conditions:

$$\begin{aligned}
&\varphi(\eta+1,\zeta+1)-\varphi(\eta+1,\zeta) \ge \varphi(\eta,\zeta+1)-\varphi(\eta,\zeta) && (\forall \eta,\zeta\in\mathbb{Z}_+),\\
&\varphi(\eta-1,\zeta+1)-\varphi(\eta-1,\zeta) \ge \varphi(\eta,\zeta)-\varphi(\eta,\zeta-1) && (\forall \eta,\zeta\in\mathbb{Z}_{++}),\\
&\varphi(\eta+1,\zeta-1)-\varphi(\eta,\zeta-1) \ge \varphi(\eta,\zeta)-\varphi(\eta-1,\zeta) && (\forall \eta,\zeta\in\mathbb{Z}_{++}).
\end{aligned}$$

Multimodular functions satisfy the following inequalities.

Proposition 2.1. *Let $\varphi : \mathbb{Z}_+^2 \to \mathbb{R}$ be a multimodular function, and $\eta, \zeta, \eta', \zeta' \in \mathbb{Z}_+$.*
(i) *If $\eta > \eta'$ and $\zeta < \zeta'$, then it holds that*

$$\varphi(\eta,\zeta)+\varphi(\eta',\zeta') \ge \varphi(\eta-1,\zeta+1)+\varphi(\eta'+1,\zeta'-1). \quad (2.1)$$

(ii) *If $\eta > \eta'$ and $\eta+\zeta > \eta'+\zeta'$, then it holds that*

$$\varphi(\eta,\zeta)+\varphi(\eta',\zeta') \ge \varphi(\eta-1,\zeta)+\varphi(\eta'+1,\zeta'). \quad (2.2)$$

3. Algorithms for (DA)

3.1 Review of Steepest Descent Algorithm

We first review the steepest descent algorithm for (DA) in [1]. Denote by $R \subseteq \mathbb{Z}^n \times \mathbb{Z}^n$ the feasible region of the problem (DA), i.e.,

$$R = \{(d,b) \in \mathbb{Z}^n \times \mathbb{Z}^n \mid d(N)+b(N) = D+B,\ b(N) \le B,\ \ell \le d+b \le u,\ d \ge \mathbf{0},\ b \ge \mathbf{0}\}.$$

Recall that given a feasible solution $(d,b) \in R$, the vectors $d+b$ and b, respectively, represent the number of docks (i.e., empty docks and docks with bike) and the number of bikes at each station. A feasible solution $(d,b) \in R$ is said to be *bike-optimal* if $c(d,b) \le c(d',b')$ holds for every $(d',b') \in R$ with $d'+b' = d+b$. That is, a bike-optimal feasible solution is a feasible solution such that under the condition that the number of docks at each station $i \in N$ is fixed to $d(i)+b(i)$, the allocation of bikes given by b is optimal. For a vector $x \in \mathbb{Z}_+^n$ with $x(N) = D+B$ and $\ell \le x \le u$, a bike-optimal feasible solution $(d,b) \in R$ with $d+b = x$ is an optimal solution of the following problem:

$$\text{(SRA}(x)\text{)}\quad \left|\begin{array}{ll} \text{Minimize} & c(x-b,b) \equiv \sum_{i=1}^n c_i(x(i)-b(i),b(i)) \\ \text{subject to} & b(N) \le B, \\ & b \le x, \\ & b \in \mathbb{Z}_+^n. \end{array}\right.$$

The problem (SRA(x)) can be seen as a simple resource allocation problem and can be solved efficiently.

Proposition 3.1 ([3], [4])**.** *The problem* (SRA(x)) *can be solved in* $O(n \log(B/n))$ *time and in* $O(n + B \log n)$ *time. Moreover, if a feasible solution $b' \in \mathbb{Z}_+^n$ of* (SRA(x)) *is available, then the problem can be solved in* $O(n + B' \log n)$ *time with* $B' = \min\{\|b-b'\|_1 \mid b$ *is an optimal solution of* (SRA(x))$\}$.

We also use the following property of the problem (SRA(x)) in the proximity-scaling algorithm for (DA).

Proposition 3.2. *Suppose that $b' \in \mathbb{Z}_+^n$ is a feasible solution of* (SRA(x)) *such that $c(x-b',b') \le c(x-b,b)$ holds for every feasible solution $b \in \mathbb{Z}_+^n$ of* (SRA(x)) *with $b \equiv b' \bmod 2$. Then, there exists an optimal solution $b^* \in \mathbb{Z}_+^n$ of* (SRA(x)) *with $\|b^*-b'\|_1 \le n$.*

For $(d,b) \in R$, we denote by $N(d,b) \subseteq \mathbb{Z}^n \times \mathbb{Z}^n$ the neighborhood of (d,b) defined by

$$N(d,b) = N_1(d,b) \cup N_2(d,b) \cup \cdots \cup N_6(d,b),$$
$$N_1(d,b) = \{(d+\chi_i-\chi_j, b) \in \mathbb{Z}^n \times \mathbb{Z}^n \mid i,j \in N,\ i \neq j\},$$
$$N_2(d,b) = \{(d-\chi_j, b+\chi_i) \in \mathbb{Z}^n \times \mathbb{Z}^n \mid i,j \in N,\ i \neq j\},$$
$$N_3(d,b) = \{(d+\chi_i, b-\chi_j) \in \mathbb{Z}^n \times \mathbb{Z}^n \mid i,j \in N,\ i \neq j\},$$
$$N_4(d,b) = \{(d, b+\chi_i-\chi_j) \in \mathbb{Z}^n \times \mathbb{Z}^n \mid i,j \in N,\ i \neq j\},$$
$$N_5(d,b) = \{(d-\chi_j+\chi_t, b+\chi_i-\chi_t) \in \mathbb{Z}^n \times \mathbb{Z}^n \mid i,j \in N,\ i \neq j,\ t \in N \setminus \{i,j\}\},$$
$$N_6(d,b) = \{(d-\chi_s+\chi_i, b+\chi_s-\chi_j) \in \mathbb{Z}^n \times \mathbb{Z}^n \mid i,j \in N,\ i \neq j,\ s \in N \setminus \{i,j\}\}.$$

The steepest descent algorithm in [1] is described as follows.

Algorithm SteepestDescentDA

Step 0:

Set (d_0, b_0) be an arbitrarily chosen bike-optimal feasible solution of (DA), and $k := 1$.

Step 1:

If $c(d', b') \geq c(d_{k-1}, b_{k-1})$ for every $(d', b') \in N(d_{k-1}, b_{k-1}) \cap R$, then output the solution (d_{k-1}, b_{k-1}) and stop.

Step 2:

Find $(d', b') \in N(d_{k-1}, b_{k-1}) \cap R$ that minimizes $c(d', b')$.

Step 3:

Set $(d_k, b_k) := (d', b')$, $k := k+1$, and go to Step 1. □

Theorem 3.3 ([1])**.** *The algorithm* SteepestDescentDA *outputs an optimal solution of the problem* (DA) *in* $O(n + \nu \log n)$ *time with*

$$\nu = \min\{\|(d+b) - (d_0+b_0)\|_1 \mid (d,b) \text{ is an optimal solution of (DA)}\}.$$

3.2 Proximity-Scaling Algorithm

We propose a polynomial-time proximity-scaling algorithm for (DA). In the following, we fix an arbitrarily chosen feasible solution $(\check{d}, \check{b})$ of (DA). Let λ be a positive integer. A feasible solution $(d,b) \in R$ is said to be a *λ-feasible solution* of (DA) if $d \equiv \check{d} \bmod \lambda$ and $b' \equiv \check{b} \bmod \lambda$. We also say that (d,b) is *λ-optimal* for (DA) if it is a λ-feasible solution minimizing the objective function $c(d,b)$ among all λ-feasible solutions. That is, a λ-optimal solution is an optimal solution of the following problem:

$$\text{(DA}(\lambda)\text{)} \quad \begin{array}{ll} \text{Minimize} & c(d,b) \\ \text{subject to} & d(N) + b(N) = D + B, \\ & b(N) \leq B, \\ & \ell \leq d + b \leq u, \\ & d, b \in \mathbb{Z}^n_+, \\ & d \equiv \check{d} \bmod \lambda,\ b \equiv \check{b} \bmod \lambda. \end{array}$$

For $i \in N$, the function

$$c_i^\lambda(\eta, \zeta) = c_i(\lambda\eta + \check{d}(i), \lambda\zeta + \check{b}(i))$$

is also a multimodular function in (η, ζ). Therefore, the problem (DA(λ)) has the same combinatorial structure as (DA), and any algorithm for (DA) can be applied to (DA(λ)). Our proximity-scaling algorithm is based on this observation and the following proximity theorem for (DA):

Theorem 3.4. *Let λ be a positive integer with $\lambda \geq 2$, and $(d,b) \in R$ be a λ-optimal solution of* (DA)*. Then, there exists some optimal solution $(d^*, b^*) \in R$ of* (DA) *such that*

$$\|(d^* + b^*) - (d+b)\|_1 \leq 8\lambda n.$$

Proof is given later in this subsection.

Algorithm ProximityScalingDA

Step 0:

Let (d_0, b_0) be an arbitrarily chosen feasible solution of (DA) and $x_0 = d_0 + b_0$. Set $\lambda = 2^{\lceil \log_2((D+B)/4n) \rceil}$ and $p := 1$.

Step 1:

Let $b'_{p-1} \in \mathbb{Z}^n$ be a vector that is an optimal solution of the problem (SRA(x_{p-1})) with an additional condition that $b'_{p-1} \equiv b_{p-1} \bmod \lambda$.

Step 2:

Apply the algorithm SteepestDescentDA to (DA(λ)) with the initial solution $(x_{p-1} - b'_{p-1}, b'_{p-1})$ to find a λ-optimal solution (d_p, b_p).

Step 3:

If $\lambda = 1$, then output (d_p, b_p) and stop. Otherwise, set $x_p = d_p + b_p$, $\lambda := \lambda/2$, $p := p+1$, and go to Step 1. □

We analyze the time complexity of the algorithm ProximityScalingDA. The number of iterations is $O(\log((D+B)/n))$. We will show that each iteration of the algorithm can be done in $O(n \log n)$ time.

The definition of the initial λ in Step 0 implies that there exists a λ-optimal solution (d,b) with

$$\|(d+b) - (d_0+b_0)\|_1 \leq \|d+b\|_1 + \|d_0+b_0\|_1 \leq 2(D+B) \leq 8\lambda n.$$

Also, in the p-th iterations with $p \geq 2$, Theorem 3.4 implies that there exists a λ-optimal solution (d,b) with $\|(d+b) - (d_{p-1} + b_{p-1})\|_1 \leq 8\lambda n$. Hence, it follows from Theorem 3.3 that each iteration, except for Step 1, can be done in $O(n \log n)$ time. Step 1 can be also done in $O(n \log n)$ time by Propositions 3.1 and 3.2.

Hence, we obtain the following bound for the algorithm ProximityScalingDA.

Theorem 3.5. *The algorithm* ProximityScalingDA *finds an optimal solution of the problem* (DA) *in* $O(n \log n \log((D+B)/n))$ *time.*

3.2.1 Proof of Theorem 3.4

Let (d^*, b^*) be an optimal solution of (DA) that minimizes the value $\|d^* - d\|_1 + \|b^* - b\|_1$. We prove that (d^*, b^*) satisfies the inequality $\|x^* - x\|_1 \leq 8\lambda n$ with $x = d + b$ and $x^* = d^* + b^*$.

In the proof we consider the following six sets.

$$I_1 = \{i \in N \mid d(i) - d^*(i) \geq \lambda,\ b(i) - b^*(i) \leq -\lambda\}, \quad (3.1)$$
$$I_2 = \{i \in N \mid d(j) - d^*(j) \leq -\lambda,\ b(j) - b^*(j) \geq \lambda\}, \quad (3.2)$$
$$I_3 = \{i \in N \mid x(i) - x^*(i) \geq \lambda,\ d(i) - d^*(i) \geq \lambda\}, \quad (3.3)$$
$$I_4 = \{i \in N \mid x(j) - x^*(j) \leq -\lambda,\ d(j) - d^*(j) \leq -\lambda\}, \quad (3.4)$$
$$I_5 = \{i \in N \mid x(i) - x^*(i) \geq \lambda,\ b(i) - b^*(i) \geq \lambda\}, \quad (3.5)$$
$$I_6 = \{i \in N \mid x(j) - x^*(j) \leq -\lambda,\ b(j) - b^*(j) \leq -\lambda\}. \quad (3.6)$$

Lemma 3.6.

(i) *At least one of I_1 and I_2 is an empty set.*

(ii) *If $b^*(N) < B$ then $I_2 = \emptyset$ holds; if $b(N) - B \leq -\lambda$ then $I_1 = \emptyset$*

holds.

Proof. We first prove (i). Assume, to the contrary, that both of $I_1 \neq \emptyset$ and $I_2 \neq \emptyset$ hold. Then, there exist distinct $i, j \in N$ such that

$$d(i) - d^*(i) \geq \lambda,\ b(i) - b^*(i) \leq -\lambda,$$
$$d(j) - d^*(j) \leq -\lambda,\ b(j) - b^*(j) \geq \lambda.$$

We consider the pair of vectors $(d - \lambda\chi_i + \lambda\chi_j, b + \lambda\chi_i - \lambda\chi_j)$, which is a feasible solution of (DA(λ)) since $d + b = (d - \lambda\chi_i + \lambda\chi_j) + (b + \lambda\chi_i - \lambda\chi_j)$ and $b(N) = (b + \lambda\chi_i - \lambda\chi_j)(N)$. We show below that

$$\begin{aligned} c(d, b) &> c(d - \chi_i + \chi_j, b + \chi_i - \chi_j) \\ &> c(d - 2\chi_i + 2\chi_j, b + 2\chi_i - 2\chi_j) \\ &> \cdots > c(d - \lambda\chi_i + \lambda\chi_j, b + \lambda\chi_i - \lambda\chi_j). \end{aligned}$$

This, however, is a contradiction to the choice of (d, b).

For an integer λ' with $0 \leq \lambda' < \lambda$, put

$$d' = d - \lambda'\chi_i + \lambda'\chi_j, \quad b' = b + \lambda'\chi_i - \lambda'\chi_j.$$

Since $i \in \mathrm{supp}^+(d' - d^*) \cap \mathrm{supp}^-(b' - b^*)$ and $j \in \mathrm{supp}^-(d' - d^*) \cap \mathrm{supp}^+(b' - b^*)$, Proposition 2.1 (i) implies that

$$\begin{aligned} &c_i(d'(i), b'(i)) + c_i(d^*(i), b^*(i)) \\ &\quad \geq c_i(d'(i) - 1, b'(i) + 1) + c_i(d^*(i) + 1, b^*(i) - 1), \\ &c_j(d'(j), b'(j)) + c_j(d^*(j), b^*(j)) \\ &\quad \geq c_j(d'(j) + 1, b'(j) - 1) + c_j(d^*(j) - 1, b^*(j) + 1). \end{aligned}$$

Hence, we have

$$c(d', b') + c(d^*, b^*) \geq c(d' - \chi_i + \chi_j, b' + \chi_i - \chi_j) + c(d^* + \chi_i - \chi_j, b^* - \chi_i + \chi_j). \tag{3.7}$$

Note that $(d^{**}, b^{**}) \equiv (d^* + \chi_i - \chi_j, b^* - \chi_i + \chi_j)$ is also a feasible solution of (DR-AP) since $d^{**} + b^{**} = d^* + b^*$ and $b^{**}(N) = b^*(N)$. Since (d^{**}, b^{**}) satisfies

$$\|d^{**} - d\|_1 + \|b^{**} - b\|_1 < \|d^* - d\|_1 + \|b^* - b\|_1,$$

we have $c(d^*, b^*) < c(d^{**}, b^{**})$, which, together with (3.7), implies $c(d', b') > c(d' - \chi_i + \chi_j, b' + \chi_i - \chi_j)$.

Proof of (ii) is similar to (i) and omitted. □

The following two lemmas can be proven in a similar way as Lemma 3.6, and therefore proofs are omitted.

Lemma 3.7. *At least one of $I_3 = \emptyset$ and $I_4 = \emptyset$ holds.*

Lemma 3.8. *At least one of $I_5 = \emptyset$ and $I_6 = \emptyset$ holds.*

Lemma 3.9.

(i) *At least one of I_4, I_5, and I_1 is an empty set.*

(ii) *If $b^*(N) < B$ then at least one of I_4 and I_5 is an empty set.*

(iii) *At least one of I_3, I_6, and I_2 is an empty set.*

(iv) *If $b(N) - B \leq -\lambda$ then at least one of I_3 and I_6 is an empty set.*

Proof. We prove (i) only. Assume, to the contrary, that all of the sets I_4, I_5, and I_1 are nonempty, and let $i \in I_4$, $j \in I_5$, and $s \in I_1$. Then, elements i, j, s are distinct by the definitions of I_4, I_5, and I_1.

We put

$$(d', b') = (d + \lambda\chi_i - \lambda\chi_s, b - \lambda\chi_j + \lambda\chi_s),$$
$$(d^{**}, b^{**}) = (d^* - \chi_i + \chi_s, b^* + \chi_j - \chi_s).$$

Since (d', b') and (d^{**}, b^{**}) satisfy

$$d'(P) + b'(P) = d(P) + b(P),\ b'(N) = b(N),$$
$$d^{**}(P) + b^{**}(P) = d^*(P) + b^*(P),\ b^{**}(N) = b^*(N),$$

(d', b') (resp., (d^{**}, b^{**})) is a feasible solution of (DA(λ)) (resp., (DA)). Using this fact, we can derive a contradiction as in Lemma 3.6. □

Lemma 3.10. *We have $\|x - x^*\|_1 \leq 4\lambda n$ if at least one of the following two conditions holds:*

(a) $I_3 = I_5 = \emptyset$, (b) $I_4 = I_6 = \emptyset$.

Proof. Suppose that $I_4 = I_6 = \emptyset$ holds. Then, we have $x(i) - x^*(i) \geq -2\lambda$ for every $i \in N$. Let $N_- = \mathrm{supp}^-(x - x^*)$. Since $x(N) - x^*(N) = 0$, we have

$$\begin{aligned} \|x - x^*\|_1 &= [x(N \setminus N_-) - x^*(N \setminus N_-)] + [x^*(N_-) - x(N_-)] \\ &= [x(N) - x^*(N)] + 2[x^*(N_-) - x(N_-)] \\ &= 4\lambda|N_-| \leq 4\lambda n. \end{aligned}$$

Proof for the case with $I_3 = I_5 = \emptyset$ is similar. □

Lemma 3.11. *We have $\|x - x^*\|_1 \leq 8\lambda n$ if at least one of the following two conditions holds:*

(a) *$I_2 = I_4 = I_5 = \emptyset$ and $b(N) - b^*(N) > -\lambda$,*

(b) *$I_1 = I_3 = I_6 = \emptyset$ and $b(N) - b^*(N) < \lambda$.*

Proof. We consider the case where (a) holds, and show that $\|d - d^*\|_1 \leq 4\lambda n$ and $\|b - b^*\|_1 \leq 4\lambda n$ hold, which implies

$$\|x - x^*\|_1 \leq \|d - d^*\|_1 + \|b - b^*\|_1 \leq 8\lambda n.$$

Since $I_2 = I_4 = I_5 = \emptyset$, it holds that

$$d(i) - d^*(i) \geq -2\lambda, \quad b(i) - b^*(i) \leq 2\lambda \qquad (i \in N). \tag{3.8}$$

Since $b(N) - b^*(N) > -\lambda$ and $x(N) - x^*(N) = 0$, we have $d(N) - d^*(N) < \lambda$.

To prove the inequality $\|d - d^*\|_1 \leq 4\lambda n$, let $H = \mathrm{supp}^-(d - d^*)$. If $H = N$, then we have $d(i) - d^*(i) < 0$ for every $i \in N$, implying that

$$\begin{aligned} \|d - d^*\|_1 &= \sum_{i \in N} |d(i) - d^*(i)| \\ &= \sum_{i \in N} [d^*(i) - d(i)] = d^*(N) - d(N) < \lambda \leq 4\lambda n. \end{aligned}$$

If $H \neq N$, then we have

$$\begin{aligned} \|d - d^*\|_1 &= \sum_{i \in N'} |d(i) - d^*(i)| \\ &= [d(N \setminus H) - d^*(N \setminus H)] + [d^*(H) - d(H)] \\ &= [d(N) - d^*(N)] + 2[d^*(H) - d(H)] \\ &< \lambda + 4\lambda|H| \leq 4\lambda n, \end{aligned}$$

where the first inequality is by $d(N) - d^*(N) < \lambda$ and $d(i) - d^*(i) \geq$

-2λ for $i \in N$, and the second inequality is by $|H| < n$. The inequality $\|b - b^*\|_1 \leq 4\lambda n$ can be proved similarly by using the inequalities $b(N) - b^*(N) > -\lambda$ and $b(i) - b^*(i) \leq 2\lambda$ for $i \in N$. □

Lemma 3.12. *We have* $\|x - x^*\|_1 \leq 8\lambda n$.

Proof. By Lemmas 3.7 and 3.8, we have the following four possibilities:

(Case 1) $I_4 = I_6 = \emptyset$,
(Case 2) $I_3 = I_5 = \emptyset$,
(Case 3) $I_4 = I_5 = \emptyset$, $I_3 \neq \emptyset$, $I_6 \neq \emptyset$,
(Case 4) $I_3 = I_6 = \emptyset$, $I_4 \neq \emptyset$, $I_5 \neq \emptyset$.

If Case 1 or 2 holds, then we have $\|x - x^*\|_1 \leq 8\lambda n$ by Lemma 3.10. Below we give proofs for Cases 3 and 4.

[Proof for Case 3] By Lemma 3.9 (iii) and (iv), we have $I_2 = \emptyset$ and $b(N) - B > -\lambda$; the second inequality implies $b(N) - b^*(N) > -\lambda$ since $b^*(N) \leq B$. Hence, we have $\|x - x^*\|_1 \leq 8\lambda n$ by Lemma 3.11.

[Proof for Case 4] By Lemma 3.9 (i) and (ii), we have $I_1 = \emptyset$ and $b^*(N) = B$; the second equation implies $b(N) - b^*(N) < \lambda$ since $b(N) \leq B$. Hence, we have $\|x - x^*\|_1 \leq 8\lambda n$ by Lemma 3.11. □

4. Polynomial-Time Algorithm for (DR)

We propose a polynomial-time algorithm for the problem (DR). For this, we first show that using a parameter, the problem (DR) can be decomposed into two subproblems that have the same structure as (DA).

Given a problem (DR), we consider its relaxation (DA) obtained by removing the L1-distance constraint in (DR). Let $(d^\bullet, b^\bullet) \in \mathbb{Z}^n_+ \times \mathbb{Z}^n_+$ be an optimal solution of the problem (DA). If $(d^\bullet, b^\bullet)$ is a feasible solution of (DR), then it is an optimal solution of (DR). Hence, in the following we assume that $(d^\bullet, b^\bullet)$ is not a feasible solution of (DR), i.e., $\|(d^\bullet + b^\bullet) - (\bar{d} + \bar{b})\|_1 > 2\gamma$. In this case, there exists an optimal solution (d^*, b^*) of (DR) such that

$$\|(d^* + b^*) - (\bar{d} + \bar{b})\|_1 = 2\gamma \tag{4.1}$$

(see [1]). Moreover, using $(d^\bullet, b^\bullet)$ we can restrict the region containing an optimal solution of (DR), as follows.

Lemma 4.1. *Let* $(d^\bullet, b^\bullet) \in \mathbb{Z}^n_+ \times \mathbb{Z}^n_+$ *be an optimal solution of the problem* (DA)*. Then, there exists some optimal solution* $(d^*, b^*) \in \mathbb{Z}^n_+ \times \mathbb{Z}^n_+$ *of the problem* (DR) *such that*

$$\bar{d}(i) + \bar{b}(i) \leq d^*(i) + b^*(i) \leq d^\bullet(i) + b^\bullet(i) \quad (\forall i \in \mathrm{supp}^+((d^\bullet + b^\bullet) - (\bar{d} + \bar{b}))),$$
$$\bar{d}(i) + \bar{b}(i) \geq d^*(i) + b^*(i) \geq d^\bullet(i) + b^\bullet(i) \quad (\forall i \in N \setminus \mathrm{supp}^+((d^\bullet + b^\bullet) - (\bar{d} + \bar{b}))).$$

Let $P = \mathrm{supp}^+((d^\bullet + b^\bullet) - (\bar{d} + \bar{b}))$. Lemma 4.1 and (4.1) imply that there exists some optimal solution (d^*, b^*) of the problem (DR) such that

$$d^*(P) + b^*(P) = \bar{d}(P) + \bar{b}(P) + \gamma,$$
$$d^*(N \setminus P) + b^*(N \setminus P) = \bar{d}(N \setminus P) + \bar{b}(N \setminus P) - \gamma.$$

Hence, the problem (DR) can be reformulated as a problem without L1-distance constraint, which we denote (DR-L):

$$\begin{array}{ll} \text{Minimize} & c(d, b) \\ \text{subject to} & d(N) + b(N) = D + B, \\ & b(N) \leq B, \\ & d(P) + b(P) = \bar{d}(P) + \bar{b}(P) + \gamma, \\ & d(N \setminus P) + b(N \setminus P) = \bar{d}(N \setminus P) + \bar{b}(N \setminus P) - \gamma, \\ & \ell \leq d + b \leq u,\ d, b \in \mathbb{Z}^n_+. \end{array}$$

To solve the problem (DR-L) efficiently, we consider two subproblems (DR-L-A(α)) and (DR-L-B(α)) with parameter $\alpha \in \mathbb{Z}_+$; the subproblem (DR-L-A(α)) is given as

$$\begin{array}{ll} \text{Minimize} & \sum_{i \in P} c_i(d(i), b(i)) \\ \text{subject to} & b(P) \leq \alpha, \\ & d(P) + b(P) = \bar{d}(P) + \bar{b}(P) + \gamma, \\ & \ell(i) \leq d(i) + b(i) \leq u(i),\ d(i), b(i) \in \mathbb{Z}_+\ (i \in P), \end{array}$$

and (DR-L-B(α)) is defined similarly to (DR-L-A(α)), where P is replaced with $N \setminus P$ and the first constraint $b(P) \leq \alpha$ is replaced with $b(N \setminus P) \leq B - \alpha$. The two subproblems have the same structure as the problem (DA), and therefore can be solved in $O(n \log n \log((D + B)/n))$ time by Theorem 3.5.

We denote by $\psi_A(\alpha)$ (resp., $\psi_B(\alpha)$) the optimal value of the problem (DR-L-A(α)) (resp., (DR-L-B(α))). Then, the optimal value of the problem (DR-L) is given by $\min_{0 \leq \alpha \leq B}[\psi_A(\alpha) + \psi_B(\alpha)]$. The next property shows that the minimum value of $\psi_A(\alpha) + \psi_B(\alpha)$ can be computed by binary search with respect to α.

Proposition 4.2. *The values* $\psi_A(\alpha)$ *and* $\psi_B(\alpha)$ *are convex functions in* $\alpha \in [0, B]$.

Since the binary search terminates in $O(\log B)$ iterations, we obtain the following time bound.

Theorem 4.3. *The problem* (DR) *can be solved in* $O(n \log n \log((D + B)/n) \log B)$ *time.*

References

[1] D. Freund, S.G. Henderson, and D.B. Shmoys. Minimizing multimodular functions and allocating capacity in bike-sharing systems. In *Proc. 19th IPCO*, LNCS 10328, pages 186–198. Springer, Berlin, 2017.
[2] D. Freund, S.G. Henderson, and D.B. Shmoys, Minimizing multimodular functions and allocating capacity in bike-sharing systems. preprint, arXiv:1611.09304, 2017.
[3] D. S. Hochbaum, Lower and upper bounds for the allocation problem and other nonlinear optimization problems. *Math. Oper. Res.*, 19:390–409, 1994.
[4] T. Ibaraki and N. Katoh: *Resource Allocation Problems: Algorithmic Approaches.* MIT Press, Cambridge, MA, 1988.