Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 2780711, 14 pages
https://doi.org/10.1155/2022/2780711

Research Article

Rebalancing Docked Bicycle Sharing System with Approximate Dynamic Programming and Reinforcement Learning

Young-Hyun Seo,[1] **Dong-Kyu Kim**,[2] **Seungmo Kang**,[1] **Young-Ji Byon**,[3] **and Seung-Young Kho**[2,4]

[1]*School of Civil, Environmental and Architectural Engineering, Korea University, Seoul 02841, Republic of Korea*
[2]*Department of Civil and Environmental Engineering, Seoul National University, Seoul 08826, Republic of Korea*
[3]*Department of Civil Infrastructure and Environmental Engineering, Khalifa University of Science and Technology, Abu Dhabi 127788, UAE*
[4]*Institute of Construction and Environmental Engineering, Seoul National University, Seoul 08826, Republic of Korea*

Correspondence should be addressed to Seungmo Kang; s_kang@korea.ac.kr and Seung-Young Kho; sykho@snu.ac.kr

Received 2 November 2021; Revised 28 February 2022; Accepted 23 March 2022; Published 9 May 2022

Academic Editor: Rosa G. González-Ramírez

The bicycle, an active transportation mode, has received increasing attention as an alternative in urban environments worldwide. However, effectively managing the stock levels of rental bicycles at each station is challenging as demand levels vary with time, particularly when users are allowed to return bicycles at any station. There is a need for system-wide management of bicycle stock levels by transporting available bicycles from one station to another. In this study, a bicycle rebalancing model based on a Markov decision process (MDP) is developed using a real-time dynamic programming method and reinforcement learning considering dynamic system characteristics. The pickup and return demands are stochastic and continuously changing. As a result, the proposed framework suggests the best operation option every 10 min based on the realized system variables and future demands predicted by the random forest method, minimizing the expected unmet demand. Moreover, we adopt custom prioritizing strategies to reduce the number of action candidates for the operator and the computational complexity for practicality in the MDP framework. Numerical experiments demonstrate that the proposed model outperforms existing methods, such as short-term rebalancing and static lookahead policies. Among the suggested prioritizing strategies, focusing on stations with a larger error in demand prediction was found to be the most effective. Additionally, the effects of various safety buffers were examined.

1. Introduction

A public bicycle sharing (PBS) system can contribute to alleviating several urban problems, such as traffic congestion and air pollution. It is a sustainable transportation mode that can satisfy last-mile traffic demands [1]. It has been observed that public bicycles are increasingly used for various trip purposes, including leisure trips in parks and commuting trips to subway stations and workplaces. Hence, the academic interest in PBS systems is gradually increasing in terms of planning and operational strategies [2].

The operational strategies pursue short-term optimization using existing infrastructure and resources, including bicycle rebalancing problems. The random demand of the PBS system users may cause an imbalance in the station inventory. Therefore, it is necessary for operators to establish an efficient rebalancing strategy to prevent system failures—such as empty (i.e., no bicycles available) or full stations (i.e., no available rack for bicycles or underutilized inventory)—that generate an unmet demand for pickup or return, respectively, particularly in a dock-based system.

A rebalancing strategy aims to determine an optimized route for a cargo vehicle to transport bicycles from and to stations and determine the optimal number of bicycles to load or unload at each station [3]. Accordingly, the bicycle rebalancing problem can be represented as a sequential

decision-making problem at each stop and formulated using a Markov decision process (MDP) [4]. However, the MDP formulation for the bicycle rebalancing problem may involve a combinatorial-dimension characteristic because each state is defined with a combination of the bicycle stock at each station, location, load level of the cargo vehicle, and time horizon. Studies in the existing literature where the MDP was applied to this problem have been limited to simplified approaches, including considering deterministic demands, delivering bikes with a safety buffer margin or target fill levels, or avidly visiting the nearest unbalanced station.

The dynamic programming (DP) method is one of the most widely used methods for MDP. DP decomposes the original problem into simpler subproblems. DP requires a complete and accurate environmental model [5]. When the size of the state and action spaces of the MDP increases, it is not feasible to calculate the expected cost for all states and actions (i.e., the curse of dimensionality) [5]. Therefore, approximate approaches to DP are often adopted for practical-size problems.

The main objective of this study is to develop a rebalancing model with stochastic demands and dynamic PBS system characteristics. "Stochastic" refers to the fact that the demand is not known in advance and that the demand follows a stochastic distribution. Meanwhile, "dynamic" means that subsequent decisions are made over a planning horizon [6]. As the demand fluctuates stochastically, an error occurs because of the difference between the forecasted and observed demands. This fluctuation necessitates rebalancing by operators. For this purpose, a stochastic distribution of user demand was applied based on historical data, and approximate DP and reinforcement learning (RL) were utilized to find an optimal solution in this study. The performance of the adopted models is compared with the performance of strategies in the literature. Finally, policy implications are presented by proposing an appropriate prioritizing strategy. PBS service operators have accumulated a large amount of trip data; however, many existing studies have not considered the stochastic nature of demand, which can be inferred from the data [6, 7]. Indeed, numerous studies have solved the rebalancing problem with deterministic demands. However, a stochastic demand approach that utilizes big data has the potential to provide more accurate solutions. In this study, a dynamic rebalancing algorithm is proposed to reflect the operational reality in the field under the assumption that the operator cannot predict the actual demand in advance.

The remainder of this study is organized as follows: Section 2 reviews the literature on PBS systems, particularly on rebalancing issues. Section 3 describes the assumptions adopted in this study, the model formulations, and the MDP. Section 4 describes the real-time DP and RL algorithms. In Section 5, numerical examples are presented and the results are discussed. Data descriptions and descriptive statistics have also been presented. Section 6 presents conclusions and ideas for future research.

2. Literature Review

2.1. Bicycle Rebalancing Problem. In general, studies on rebalancing public bicycles can be categorized into two types depending on the assumptions of when the operation is conducted: the first is the static bicycle rebalancing problem (SBRP), which ignores user activities. This type can be applied to operations at night. However, as the user demand increases during the daytime, there is a limitation of that inventory can be adjusted only after work hours to suitably accommodate the demand [8]. In addition, demand often occurs randomly, and user activities can stochastically vary as a result of rebalancing. In other words, although an actual system is a dynamic bicycle rebalancing problem (DBRP) that changes over time, the associated complexity of the problem has bounded and enforced the scopes of some previous research works to a simplified version of the problem as an SBRP. Therefore, it is necessary to establish rebalancing strategies that reflect the real environment with forecasted demands to respond effectively to changing inventory levels in real time.

The DBRP prioritizes minimizing unmet demand (or user dissatisfaction) during the rebalancing process rather than minimizing travel costs. As mentioned earlier, most DBRP studies assume a deterministic demand. For instance, Contardo et al. improved the computation time to solve the problem using the Dantzig–Wolfe decomposition and Benders decomposition, and they obtained the upper and lower bounds of the unmet demand [7]. However, the assumption of nontime-varying demand is a limitation of the study. Moreover, Wang proposed an original mixed-integer programming (MIP) model that considers the dynamic characteristics of demand and compares the performances of two heuristic solutions with exact solutions [9]. The rolling horizon approach and Benders decomposition were applied in the study. In addition, Zhang et al. developed an integrated model to forecast the inventory level and demand, rebalancing, and routing; the authors allowed operators to visit a station once within the time window [8]. Gleditsch and Hagen simplified and approximated the problem into a deterministic subproblem under the assumption of known demands, and column generation heuristics were applied to the problem [3]. Moreover, Fernández et al. presented four dynamic strategies: keeping a high inventory level, maintaining high inventory rates, considering travel distances with inventory or inventory rates, and considering the inventory of neighborhood stations [10]. Furthermore, Chiariotti et al. presented a strategy that first modeled a station inventory rate, followed by determining a rebalancing time, and then selected cargo vehicle routes and stations [11]. In a dynamic system, the dynamic characteristics that can be altered through decision-making processes need to be considered. The assumptions made in previous studies do not accurately reflect reality, as the predetermined route may be different from the actual route owing to the uncertainty in demand.

2.2. Markov Decision Process (MDP). The bicycle rebalancing problem can be represented as a sequential decision-making problem [1]. An operator determines the number of bicycles to load or unload based on the current system state and moves to the next station, where rebalancing is required. Consequently, more users can pick up or return bicycles, and the operator repeats the process based on the transitioned environment.

Research on bicycle rebalancing using MDP is rare and has recently begun in academia. For instance, Legros attempted to minimize the long-run rate of unsatisfied user demand and analyzed the case of a single cargo vehicle and a time horizon segmented into periods of equal length, without considering a predefined route [12]. In addition, Brinkmann et al. developed a dynamic lookahead policy (DLA) heuristic and demonstrated that the rollout algorithm (RA) failed to obtain competitive results within a reasonable computational time [6]. Accordingly, an inventory decision was made to minimize the unmet demand at the current station within a horizon, and a routing decision was made to select a station that could avert the largest unmet demand within the horizon. The horizons per hour were determined using the nonparametric value function approximation (VFA) method. However, the study was limited by the adoption of a lengthy window period of 1 h.

Based on a review of the above-mentioned related works, although the system changes over time, the SBRP has primarily been studied academically, simplifying the complexity associated with an actual system. In fact, most DBRP-related studies considered deterministic demand and focused on minimizing the unmet demand during a rebalancing process using a time-space network or MDP. Research using a time-space network to implement decision-making behaviors, including future information, is challenging.

It is necessary to develop a model that simulates stochastic demands and dynamic programming for public bicycles to reflect the system realistically. Since the demands are stochastic and the system states are dynamic, stochastic dynamic programming for rebalancing PBS systems is required. Related stochastic dynamic studies have focused on short-term strategies without considering future demands [11, 13]. This strategy is similar to that of a simple heuristic. Even if future demand is considered, only the next station to be visited is considered [12], or several target inventory levels are established [6].

Furthermore, it is necessary to develop a rebalancing strategy to cope with the inevitable emergencies caused by these dynamic characteristics. The strategy should proactively respond to inventory shortages or excess that may occur because of inaccuracies in demand forecasts or rapid demand fluctuations. In existing studies, the time unit of analysis is a lengthy time window, which is unsuitable for providing detailed responses during peak periods. Besides, detailed information on demand distribution is missing in most of the existing studies.

2.3. Reinforcement Learning. The RL method inspires actions in response to changes in an environment to maximize an agent's cumulative rewards through their interactions with the environment [5]. Hence, the RL approach has a distinct advantage in that it does not require excessive prior information about the environment unlike optimal transport or pickup and delivery problems [14]. Initially, the agent does not have information about actions to take; however, the agent progressively learns to identify actions that result in the highest reward. This method has been applied to transportation systems [15–19] and has primarily been applied to traffic signal control [15–17] or route planning systems [18, 19]. This method has not been extensively applied to sharing systems [14, 20–23].

There are two main RL approaches to the sharing system: vehicle-based and user-based. Li et al. proposed a clustering algorithm and spatiotemporal RL method for a vehicle-based approach [21]. The clustering algorithm grouped stations and multiple trikes to reduce problem complexity. The RL model learns an optimal rebalancing policy for each cluster, thereby minimizing the total unmet demand over a long-term horizon. Li et al. proposed a static rebalancing method using a policy gradient-based RL method to enhance user experience and reduce operational costs in the PBS system [14]. A case study was conducted on eight central hubs to test the performance of a deep RL method and it was demonstrated that as the learning progressed, an agent found a suboptimal policy to reposition the bikes for each hub. Mao et al. solved a taxi dispatch problem by developing an RL framework to rebalance autonomous vehicles when the travel demand and taxi supply were imbalanced [23]. The results showed that the algorithm always converges close to a theoretical optimum.

In a user-based approach, Pan et al. determined the amount payable to users to incentivize them to rebalance the system using a hierarchical reinforcement pricing algorithm with an MDP model [22]. The objective of the study was to maximize the total number of satisfied requests subject to the rebalancing budget available. Moreover, the study considered each region's fill levels, budget, previous pickup and return demand, previous expenses, and past unmet-service rate as state factors. An et al. established an MDP problem with the goal of minimizing the cost of a car-sharing system and applied a deep deterministic policy gradient (DDPG) method (actor-critic method) [20]. The study introduced two rewarding mechanisms, picking bonuses and parking bonuses, to encourage users to balance the car-sharing system. The research showed that these mechanisms increased the service provider's long-term profit by inducing users' behaviors through price leverage, increased user persistence, and cultivated usage habits. In general, conflicting objectives to be optimized are usually formalized as multi-objective optimization problems [16, 24, 25]. The problem avoids extreme solutions that can be obtained from the single-objective optimization problem [17–23].

Table 1: The nomenclatures used in this study.

Sets	
$N = \{n_0, \ldots, n_{\max}\}$	Set of stations (0: depot)
$T = \{t_0, \ldots, t_{\max}\}$	Set of time steps
$S = \{s_0, \ldots, s_{\max}\}$	Set of states
$A_s = \{a_0, \ldots, a_{\max} \mid a = (\iota^a, n^a)\}, \forall s \in S$	Set of feasible actions
$\Pi = \{\pi_0, \ldots, \pi_{\max} \mid \pi\colon S \to A\}$	Set of policies
$K = (0, \ldots, k_{\max})$	Sequence of decision points
Indices	
k	Decision point
$s_k \in S$	Decision state
$t_k \in T$	Point in time in state s_k
Parameters	
c_v	Cargo vehicle capacity
$\tau(\cdot, \cdot)$	Travel time between two stations
τ_r	Service time for rebalancing per bicycle
c^n	Station capacity
β	Safety buffer
z	z-score for the safety stock
p_k	Station observed pickup demand at time t_k
d_k	Station observed return demand at time t_k
$\widehat{p}_k$	Station predicted pickup demand at time t_k
$\widehat{d}_k$	Station predicted return demand at time t_k
Variables	
f_k^v	Cargo vehicle load at time t_k
$n_k^v \in N$	Cargo vehicle location at time
$t_k\ y_k$	The number of delivered bikes from the cargo vehicle at time t_k
$f_k = (f_k^{n_0}, \ldots, f_k^{n_{\max}})$	Station fill levels at time t_k
$r_k = (r_k^{n_0}, \ldots, r_k^{n_{\max}})$	Station fill rate index in time t_k
ι^a	Delivery decision
n^a	Next station decision

3. Model Formulation

3.1. Nomenclature. The nomenclatures used in this study are listed in Table 1. The adopted nomenclatures were mostly referenced from the work by Seo and Brinkmann et al. [1, 6].

3.2. Problem Definition. Each station comprises an initial inventory f_0^n, capacity c^n, and predicted pickup and return demands $\widehat{p}_k^n$, $\widehat{d}_k^n$ in the time interval $[t_0, t_{k_{\max}}]$. The observed pickup demand p_k and return demand d_k at time step t_k are not known in advance. At time step t_k, the observed demands p_{k-1}^n and d_{k-1}^n are revealed, and inventory f_k^n is changed by the observed demands and delivery. An operator driving a cargo vehicle with a loading capacity c_v should start from depot n_0 and return to the depot at the end of the time horizon. Meanwhile, the operator determines the number of bicycles to be delivered to or withdrawn from the current station and the next station to visit at each decision point. However, an unmet demand occurs if a pickup demand or a return demand is left unsatisfied for each station owing to a lack of bicycles for pickups or available docks for return.

The present study aims to determine the route of a cargo vehicle and the number of bicycles to pick up from and deliver to stations to minimize the expected unmet demand. In this study, several assumptions were made to simplify the problem, with reference to Seo [1]. First, it was assumed that public bicycle trips have a spatiotemporal pattern and that future demand follows a historical pattern. Based on the first assumption, future demand can be predicted using historical demand. Second, the observed demands are assumed to follow a nonhomogeneous Poisson distribution to represent customers' random arrival processes. Accordingly, a station can be visited only once, excluding a depot. This assumption was made because an operator may distribute or withdraw bicycles at stations to ensure a safety stock. The safety stock refers to an inventory level that prevents future stockouts, so that a single visit can prevent out-of-stock scenarios within a given time horizon. A cargo vehicle is assumed to have a capacity of 15 bicycles and travel to stations at a Euclidean distance at a speed of 20 km/h. The handling time was set to a constant value of one minute per bicycle.

3.3. Markov Decision Process. The scheme of the PBS system is represented by the MDP [6, 26]. An operator serves as an agent and the system corresponds to an environment to rebalance the PBS system. The operator determines the number of bicycles to be delivered to a station from the cargo vehicle or to withdraw from the station and load onto the cargo vehicle. Minimizing the expected costs is a potential solution to this problem.

An MDP is based on the interaction between an agent and the environment. The agent makes a decision in a given state that changes the environment. Subsequently, the agent gains a reward, collects information about the next state from the environment, and makes a subsequent decision. In

this context, the MDP model can be represented by a five-tuple (S, A, Pr, R, γ) [20]. State (S) represents the entire environment's information at each moment, and action (A) indicates a set of decisions, from which the agent can choose. The transition probability (Pr) is defined as the probability of the transition from state s_k to state s_{k+1} when taking action a_k at time t_k. The reward (R) corresponds to a value used by the agent to determine the action, and the discount rate (γ) is the reduction rate of the reward over time. Each tuple is described in detail in the following section:

The solution to this problem involves determining the optimal policy $\pi^* \in \Pi$ that describes the best action for each state in the MDP [6]. The penalty function $U\ D(s_k, \pi)$ describes the number of unmet demands of policy π in state s_k. A policy alludes to a rule that determines a decision based on the available information in a state [27]:

$$\pi^* = \arg\min_{\pi \in \Pi} \mathbb{E}\left[\sum_{k=0}^{k_{\max}} U\ D(s_k, \pi(s_k))|s_0\right]. \tag{1}$$

3.3.1. State. In the PBS system, the states are composed of the inventory levels of stations, locations, and contents of cargo vehicles [2]. The state space for this study was constructed according to Seo and Brinkmann et al. [1, 6]. The state space representation includes three factors: t_k is a time step, n_k^v is a station where cargo vehicle v is currently positioned, and r_k denotes binary variables representing a station's fill rate index. Moreover, r_k has a value of zero if the fill rate is between the safety buffers and one otherwise. The safety buffer is defined as the interval of a certain percentage of capacity c^n, which can be adjusted by β:

$$r_k^n = \begin{cases} 0, & \text{if } \beta c^n < f_k^n < (1-\beta)c^n \\ 1, & \text{otherwise} \end{cases}, \quad \forall n \in N \backslash \{n_0\}. \tag{2}$$

This state-space definition makes it easy to extend the spatial and temporal ranges; however, the computation time is multiplied linearly by this extension. In practice, demand forecasting and relocation routing should not be time consuming. Therefore, we set an appropriate time step to reduce the complexity and used the fill rate index instead of the fill level. The fill rate index is less accurate in indicating the station's information than the inventory. However, this index can significantly reduce the number of states by aggregating the fill level.

3.3.2. Action. The operator's action at each decision point comprises two consecutive decisions: a delivery decision and a next-station decision. A delivery decision represents the number of bikes to be loaded or unloaded at the current station to satisfy the target fill level of the station. The next-station decision chooses a station toward which the operator will move to rebalance. This study prioritizes the next-station decision, whereas the former decision is automatically determined by external factors to reduce the associated action space.

Since the predicted demand may fluctuate owing to weather conditions or random incidents, the concept of safety stock is introduced. Safety stock refers to the inventory level required to prevent stockouts [28]. Stockouts stem from numerous factors, such as demand fluctuations and prediction inaccuracies. The safety stock equation is as follows: under the assumption that demand follows a Poisson distribution, the standard deviation of demand can be replaced by the mean of demand. In addition, the total lead time (PC) and time increment used to calculate the standard deviation of the demand (T_1) are also assumed to be identical:

$$(\text{Safety stock}) = z\sqrt{\frac{PC}{T_1}}\sigma_D, \tag{3}$$

where z, Z-score; PC, total lead time; T_1, time increment used for calculating the standard deviation of the demand; σ_D, standard deviation of the demand.

3.3.3. Reward. A reward is a quantitative value associated with the actions of an operator. In this study, the reward was set as the sum of the total unmet demands from all stations given action a_k and the realization of the transition ω: $S \times A \longrightarrow S$. The operator adopts a policy that minimizes the reward.

3.3.4. Transition Probability. The postaction state is changed by the operator's action and the users' pickup or return demands at the corresponding time step. In a scenario where the demands are deterministic and known in advance, the postaction state can be uniquely determined, as the transition probability to the relevant state would be one, whereas the probabilities of the other states are zero. As a result, the calculation of the Bellman optimality equality becomes simple. However, in this study, stochastic demands were considered, and the transition probabilities to postaction states were calculated. Accordingly, the pickup and return demands were assumed to follow a time-dependent Poisson distribution. The Skellam distribution, a discrete probability distribution of the difference between two Poisson distributions with respective expected values, was applied to calculate the transition probability.

3.4. Demand Forecasting. In 2001, Breiman introduced the random forest technique, an ensemble learning method employed for classification and regression. The general idea of this technique is to combine multiple decision trees, each of which is developed individually on bootstrapped data samples. Predictions were made by obtaining the mean of the output from each decision tree [29].

In this study, historical pickup data were utilized for demand prediction. Temporal factors, such as month, day of the week, time of the day, and meteorological factors, such as temperature, wind speed, and precipitation, were used as explanatory variables. In addition, the hourly pickup and return demands were forecasted using random forest.

Observed and predicted demands can be compared at each time step by uniformly distributing predicted values on a 10-minute basis, which is a unit of time step in this study, and the associated prediction errors can be calculated. Considering that usage patterns vary depending on weekday and subscription type, four different demand prediction models were constructed. The experiment was performed with the "randomForest" package of R. Details regarding the demand forecasting technique used in this study can be found in Seo, Seo et al., and Cho et al. [1, 30, 31].

4. Solution Algorithms

4.1. Real-Time Dynamic Programming. In this study, an asynchronous technique, real-time dynamic programming (RTDP), proposed by Barto et al., was developed to derive an approximate solution [32]. The term "asynchronous" indicates that all states are not updated the same number of times but different states are updated with different frequencies. RTDP is an on-policy trajectory-sampling version of the DP value-iteration algorithm [5]. The central idea of RTDP is that an agent only visits the states that are relevant to the agent. Accordingly, RTDP updates the values of states visited in actual or simulated trajectories through expected tabular value-iteration updates. For certain types of problems satisfying reasonable conditions, RTDP guarantees determining an optimal policy for relevant states without visiting every state infinitely often [5].

For these problems, since each episode beginning in a state randomly selected from a set of starting states and ending at a goal state, the RTDP converges (with a probability of one) to an optimal policy for all relevant states, provided the following conditions are satisfied [32]:

(1) The initial value of each goal state is zero.

(2) At least one policy guarantees that a goal state is reached with a probability of one from any starting state.

(3) All rewards for transitions from nongoal states are strictly negative.

(4) All initial values are equal to or greater than their optimal values (which can be satisfied by simply setting all states' initial values to zero).

The pseudocode of RTDP in Algorithm 1 is presented as follows: initialize an estimate of the value function $\overline{V}^0(s_k)$ for all states, and choose an initial state, s_k^0. Once a sample of random demands ω^i is selected, the computed value function and resulting optimal action are updated using the Bellman optimal equation. Consequently, only the corresponding state value is updated to $\widehat{v}_k^i$, and the values of the rest are not updated. The agent moves to the following state s_k^i according to action a_k^i and sample demand ω^i and repeats the above process during I iterations [1, 27]. The computational complexity of the algorithm is affected by the number of iterations, time steps, states, and actions; therefore, the time complexity of Algorithm 1 is $O(I|T|^2|N|^2 2^{|N|}|c^n|)$. This is much less than that of DP, which iteratively computes for all states.

4.2. Manipulating Algorithm. Owing to the nature of the Bellman optimal equation, all the possible next states and actions must be considered. Accordingly, two manipulations can reduce the required computational effort. First, the associated action space is reduced. Regarding the routing decision, an operator only considers stations that satisfy certain conditions to visit the next station [1, 31].

Strategy 1: Consider all the stations.

Strategy 2: Consider the stations close to the current station.

Strategy 3: Consider the stations with significant forecasting errors.

Second, the next state may vary owing to stochastic demand; however, it is inefficient to consider all states. For instance, if most stations have relatively low predicted demands, the probability that $r_k = [0, \ldots, 0]$ transitions to $r_{k+1} = [1, \ldots, 1]$ is relatively low. The following state's possible fill levels are assumed to be within the standard deviation of the Skellam distribution, $\sqrt{\widehat{p}_k + \widehat{d}_k}$.

4.3. Reinforcement Learning Method. For reliable and effective operation, a feasible solution should be obtained in real time, even for a large-scale bike-sharing system. Updating and saving value functions in a table form for all combinations of state-action pairs are not feasible and nearly impossible. RL presents a method that enables agents to learn without prior knowledge of the environment and its associated model. Given various rewards depending on the actions, an agent attempts to take action for a high reward. Unlike DP and RTDP, the RL method approximates a value function using an artificial neural network (ANN) without storing it in a table.

An actor-critic technique is a combination of policy-based learning and value-based learning with two neural networks. Each of the two models calculates an action based on a state (actor) and an action's Q-value (critic). The Q-value takes an extra parameter, a current action a. $Q_\pi(s, a)$ refers to the long-term return of a current state s, taking action a under policy π.

Furthermore, the actor receives the state as the input and outputs the probability of each action. This method is referred to as policy-based learning and controls how the agent moves by learning the optimal policy. However, the critic evaluates the action by considering the state as input and calculating a value function (value-based learning). The two networks were trained separately and a gradient ascent method was adopted to update the weights. Consequently, the more time steps that are repeated, the better the actor will perform, and the better the critic evaluates the actions. The pseudocode of the actor-critic methods is shown in Algorithm 2 [5, 33]. The state, parameters, and learning rates are intialized. For each time step, the agent selects an action and obtains the following state and reward. The critic then updates the action-value function parameters w using the TD error δ, which is the difference between the unbiased estimate of the approximate value function (Q_w) and the actual value of the function. The actor updates the policy parameter θ using the policy gradient theorem. The time complexity of actor-critic methods is

Initialize the value function approximation $\overline{V}^0(s_k)$ for all states S
$i \longleftarrow 1$
while $i < I$ do
  Choose an initial state s_k^i and a sample path ω^i.
  for $k = 0, 1, 2, \ldots, k_{\max}$ do
    $\widehat{v}_k^i \longleftarrow \max_{a_k \in A^i} (C(s_k^i, a_k) + \gamma \sum_{s' \in S} \mathbb{P}(s' | s_k^i, a_k) \overline{V}_{k+1}^{i-1}(s'))$
    $a_k^i \longleftarrow \operatorname{argmax}_{a_k \in A^i} (C(s_k^i, a_k) + \gamma \sum_{s' \in S} \mathbb{P}(s' | s_k^i, a_k) \overline{V}_{k+1}^{i-1}(s'))$
    if $s_k = s_k^i$ then
      $\overline{V}_k^i(s_k) \leftarrow \widehat{v}_k^i$
    else
      $\overline{V}_k^i(s_k) \leftarrow \overline{V}_k^{i-1}(s_k)$
    $s_k^i \leftarrow \omega^i(s_k^i, a_k^i)$
  $i \leftarrow i + 1$

Algorithm 1: RTDP.

Initialize state s, parameters w, θ, and learning rates α, β.
$i \longleftarrow 1$
while $i < I$ do
  for $k = 0, 1, 2, \ldots, k_{\max}$ do
    Choose an action $a_k \sim \pi(s_k^i)$ and observe following state s_{k+1}^i and reward r.
    $\delta \longleftarrow r + \gamma Q_w(s_{k+1}^i, a_{k+1}) - Q_w(s_k^i, a_k)$.
    $w \longleftarrow w + \beta \delta \nabla_w Q_w(s_k^i, a_k)$.
    $\theta \longleftarrow \theta + \alpha \nabla_\theta \log \pi(s_k^i) Q_w(s_k^i, a_k)$.
    Update $s_k^i \leftarrow s_{k+1}^i$, $a_k \leftarrow a_{k+1}$.
  $i \leftarrow i + 1$

Algorithm 2: Actor-critic.

$O(I|T|)$) because only the number of iterations and time steps affect the complexity of the algorithm.

In this study, the actor-critic method was employed rather than REINFORCE, which is a Monte–Carlo variant of policy gradient methods. The primary advantage of the actor-critic method is that the learning speed is faster because it learns at every time step [34]. In REINFORCE, the agent learns each episode, so the learning speed is slower than that of the actor-critic method. As illustrated in Figure 1, the actor-critic (A2C) method has advantages in that it converges faster and performs better than the REINFORCE method. A2C uses an advantage function for the actor-critic method. The advantage function compares how rewarding an action is compared to other actions in each state. The reason for using the advantage function is to evaluate the advantage of an action instead of directly comparing Q values. In addition, a larger Q function value is associated with a greater error function variance, which is intuitive. Accordingly, the value function, which is the baseline, is subtracted to reduce the degree of change in the Q function.

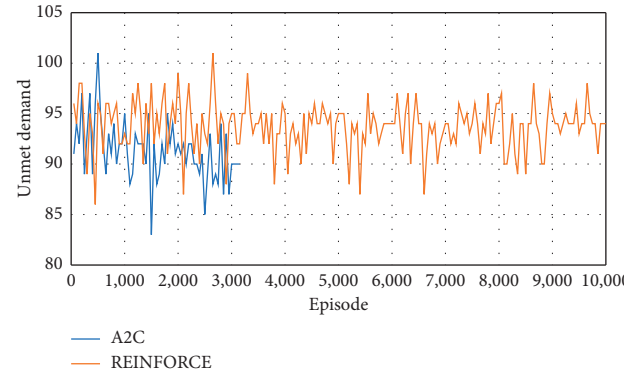

Figure 1: Performance analysis by reinforcement learning algorithm.

5. Numerical Experiment

5.1. Experimental Design. The spatial scope of this study is Yeouido District, an island with an area of 2.9 km^2 located in the southwestern part of Seoul, South Korea. Yeouido is suitable as an experimental area, primarily because the island has business areas and parks, and the demand for commuters and park users is higher than in other areas. The network used in this chapter consisted of 31 stations installed in Yeouido and a depot outside the island (Figure 2), and Euclidean distances among the stations were arbitrarily assumed. It was assumed that the stations were rebalanced using one cargo vehicle with a capacity of 15. When moving between a pair of stations, the speed of the cargo vehicle was assumed to be 20 km/h, and considering the average speed of traffic in Yeouido and the fact that it takes one minute per bicycle to withdraw or distribute the bicycles from or onto a bicycle rack at a station, and based on the historical rebalancing log, the unit time-step was set to 10 min.

Figure 2: Location of stations and depot in Yeouido, Seoul.

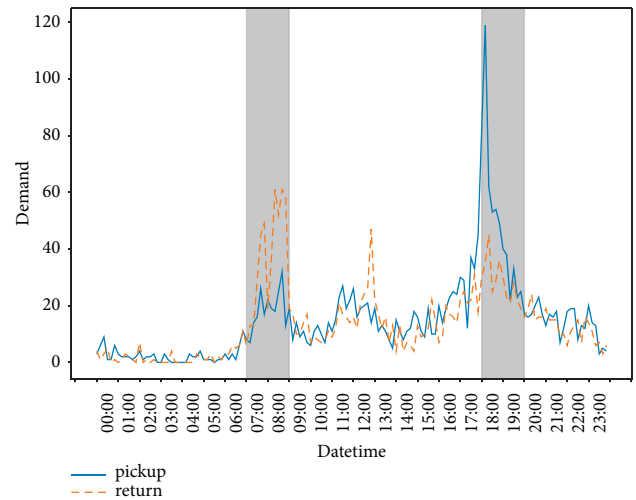

Figure 3: Daily pickup and return frequency by the time of day.

Two types of demand are used in this study: predicted demand and observed demand. For the simulation of stochastic demand, the observed demand is assumed to follow a Poisson distribution. Meanwhile, the predicted demand is used as a parameter of the distribution. Pickup and return demands within one hour are estimated at each station using three datasets (bicycle sharing dataset, holiday calendar, and meteorological dataset), which are uniformly assigned every 10 min within a period of one hour. The bicycle sharing system dataset, including the time-stamped pickup and return and the stock of each station for every 10 min, was provided by the Seoul Facilities Corporation, a public bicycle management agency based in Seoul, South Korea. The meteorological dataset was collected from the Korean Meteorological Administration (KMA) website (https://data.kma.go.kr/). Moreover, the time scope of the dataset was from August 2016, when the stations were initially installed, to September 2017. Reportedly, there were 474,123 usage records during the analysis period, with an average of 1,142 trips per day. The length of the prediction horizon was two hours for each period, from 7 a.m. to 9 a.m. or from 6 p.m. to 8 p.m. on September 20, 2017. The pickup and return demand patterns in the area are illustrated in Figure 3, with the analysis period shown in grey. Since the analysis area comprises a business district, it can be confirmed that the number of returns is higher in the morning and the number of pickups is higher in the evening.

Five to seven stations and a depot were selected from the Yeouido district. The discount factor γ, which denotes the reduction rate of the reward over time, was set to 0.9. Experiments were performed on a personal computer with an Intel® Core™ i7 2.93 GHz CPU and 8 GB of memory. The solution algorithms were coded and compiled in a Python environment based on Lee et al. [34].

5.2. Algorithm Performance

5.2.1. Number of Iterations and Computation Time. The calculation time of RTDP for 10 iterations according to the number of stations is depicted in Figure 4. The computation time for the number of stations increased exponentially with $|N|$. These observations are consistent with the computational results concerning the dimension of the state space. Specifically, each station has two values for the fill rate index (0 or 1), and thus, the state space dimension is $|S| \leq |T||N|2^{|N|}$.

5.2.2. Comparison with Benchmark Policies. Two benchmark policies were analyzed to compare the effectiveness of the strategy developed in this study. Brinkmann et al. and Brinkmann et al. proposed these strategies [6, 13]. The unmet demands averaged from 30 simulations for each policy after 1,000 iterations were performed. Accordingly, the unmet demands were calculated for all stations, and the delivery amount was the sum of the number of loaded and unloaded bicycles.

Brinkmann et al. introduced a short-term rebalancing (STR) policy [13]. Given state (t_k, n_k^v, f_k^v, f_k), an action is determined by the STR policy. If the fill levels of the current station are outside the safety buffer, then a rebalancing operation is implemented for the deficit (or excess). Moreover, Brinkmann et al. proposed static and dynamic lookahead policies [6]. This study modeled a static lookahead policy (SLA). The inventory level determined at the current station is 25%, 50%, or 75% of the station's capacity, leading to the least unmet demand as the sum of the failed pickup and failed return demands at this station.

Table 2 presents the results of these comparisons. It is common for the delivery amount to decrease as the travel time of the cargo vehicle increases (or vice versa) within a limited time. However, both the delivery and travel time decrease when a cargo vehicle's idling is long or the movement owing to the meaningless inventory decision ($\iota = 0$) increases. SLA, a strategy that utilizes predicted demand information, has a lower performance than STR. The stations that need to be rebalanced are efficiently selected in the SLA. However, they cannot be rebalanced owing to restrictions on inventory decisions (at least 25% of the station capacity). Since the analysis period was on a weekday morning, the demand for returns was higher than that for

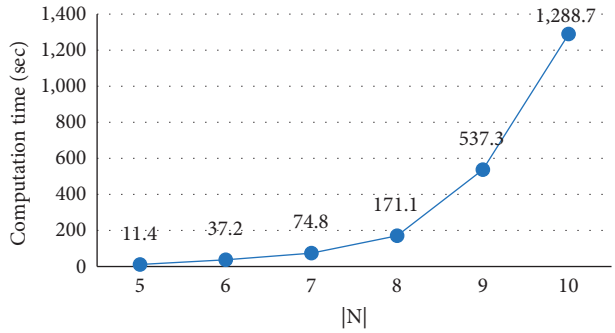

Figure 4: Computation time needed for 10 iterations.

Table 2: Key performance indicators using different benchmark strategies.

Strategy	Average unmet demand (person)	Average travel time (min)	Average delivery amount (bike)
No rebalance	10.6	—	—
STR	8.5	19.8	2.7
SLA	9.6	40.2	11.7
RTDP (1.00)	3.8	37.1	16.8
RTDP (1.65)	3.5	37.2	21.9
RTDP (2.33)	2.3	38.3	21.4

pickups. Even if all bicycles at a station need to be withdrawn to make the bicycle racks available for returns, an operator must choose 25% of capacity. In addition, if revisits were allowed, the operator could have rebalanced the bicycles later if the operator could not rebalance them during the first visit.

STR is a strategy to visit as many stations as possible by rebalancing the station's minimum number of bicycles. However, it was found that the strategy is susceptible to sudden high demands. The failed demand occurred owing to an intensive demand during the safety buffer or after the withdrawal of bicycles to prevent the station from filling up. Since STR is a reactive strategy, an operator visits a station following the occurrence of a failure. SLA has limited inventory decisions (25%, 50%, or 75% of the station capacity), thereby promoting unnecessary travel with no inventory changes. In addition, the constraint that the operator can visit a station only once worsens performance.

Overall, the RTDP outperformed the benchmark policies. In the RTDP cases, the delivery amount was higher than in the other policies, and delivery reduced unmet demands. It exhibited better performance when the z-score was high. Consequently, a higher demand can be met by withdrawing bikes in the morning, when the return demand is high.

5.2.3. Comparison by Prioritizing Strategies. For seven stations and one depot, 100 iterations were performed for each strategy. Table 3 lists the KPIs for each prioritizing strategy. A "do nothing" strategy implies that a rebalance is not performed, and 19.00 failures occurred for two hours. It is impossible to stay self-sufficient from user usage alone because the pickup demand is higher than the return demand from 7 a.m. to 9 a.m. Therefore, rebalancing using a cargo vehicle is required.

Strategies 1 and 3 exhibited better performances than the "do nothing" policy, and the performance of Strategy 2 was similar to that of the "do nothing" strategy. The reward for Strategy 1, considering all stations for the next visit, was the lowest (i.e., the best). In Strategy 3, the reward was similar to that of Strategy 1; however, it is noteworthy that the computation time was reduced by approximately 28.5% compared to Strategy 1.

5.2.4. Sensitivity Analysis. A sensitivity analysis was performed for five initial inventory cases and three station capacity cases (Figure 5). The initial inventory levels were set as percentages of the station capacities (10%, 30%, 50%, 70%, or 90%). Based on a given station capacity, both smaller capacity cases (80%) and larger capacity cases (120%) were analyzed. The unmet demand rate is calculated as the ratio of unmet demands to observed pickup or return demands.

For pickup demands during the morning peak, a slight difference was found in all capacity cases. However, the gaps in unmet return demands by station capacity are significantly different from those of unmet pickup demands. This result can be attributed to a characteristic of the system, which is a return service that connects existing bicycles. Regardless of the station capacity, the service can lead to unlimited returns on bicycles. Since predicted demands are estimated from observed demands, pickup demands can be underestimated, whereas all return demands can be observed. As the initial inventory increases, unmet demand increases. In the morning, the available slots on the dock are insufficient because there are many returns in the analysis area. It can be further confirmed by the fact that unmet pickup demand rarely occurs from 30% or more of the initial inventory.

Table 3: Key performance indicators by strategies.

Strategy	Average unmet demand (person)	Time (s)
Do nothing: no rebalance	19.00	—
Strategy 1: all stations	13.67	2,825.9
Strategy 2: near stations	19.67	2,113.0
Strategy 3: stations with large errors	14.00	2,021.6

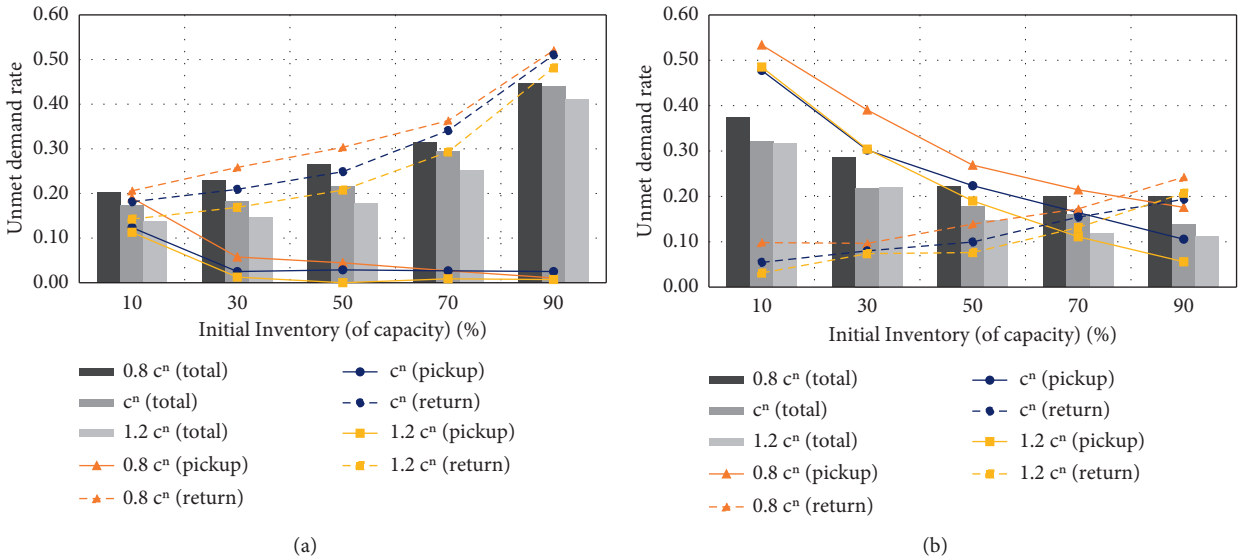

Figure 5: Sensitivity analysis with initial inventory and station capacity. (a) Morning peak; (b) evening peak.

As the initial inventory levels increase at the evening peak, the average unmet pickup demand decreases, and the average unmet return demand increases. It can be observed in Figure 3 that many pickups occurred in the analysis area in the afternoon. This phenomenon tends to make stations empty; therefore, the more initial inventory level the station has, the lower the users' pickup failure. As capacity increased and the number of bicycles increased, the unmet demand tended to decrease. In particular, the larger the initial inventory, the greater the decrease in the unmet pickup demand for capacity changes, suggesting that pickup-active stations should be considered bicycle-supplying stations.

As shown in Figure 6, the relationship between the initial inventory level, net demand, and total unmet demand was investigated. At the morning peak, all stations have a negative net demand, meaning that returns dominate pickups. Notably, Station 4 is the most active return station, and as the initial number of available bicycles increases, users find it challenging to return bicycles, increasing unmet demand. All stations, except Station 4, had lower unmet demands, regardless of the initially available bikes.

At the evening peak, three stations (Stations 2, 3, and 4) had positive net demands, and the other stations (Stations 1 and 5) had negative net demands. Station 1 shows a cluster of dots, indicating that its capacity is small. The unmet demand is high because both the pickup demand and return demand are high. Therefore, expanding the capacity will decrease the unmet demand. Station 3 has zero unmet demand even if the initially available bicycles increase unless there are few initial bicycles. This result suggests that the capacity of the station is too high compared to the demand, so its capacity can be reduced.

5.3. Large-Scale Cases

5.3.1. Deterministic Demand Contexts. All 31 stations and a depot in the Yeouido district were selected for a large-scale case analysis ($|N| = 32$). The hyperparameters in this analysis were set according to the values used in the literature [20] and adjusted to fit this study. The learning rate of both the actor and critic networks was 5×10^{-4}, and the two networks had three layers with 24 hidden layer units.

Figure 7 illustrates the performance analysis in the context of deterministic demand, which is the observed demand on that day. Strategy 1, which searches for all stations, requires more iterations to converge because it

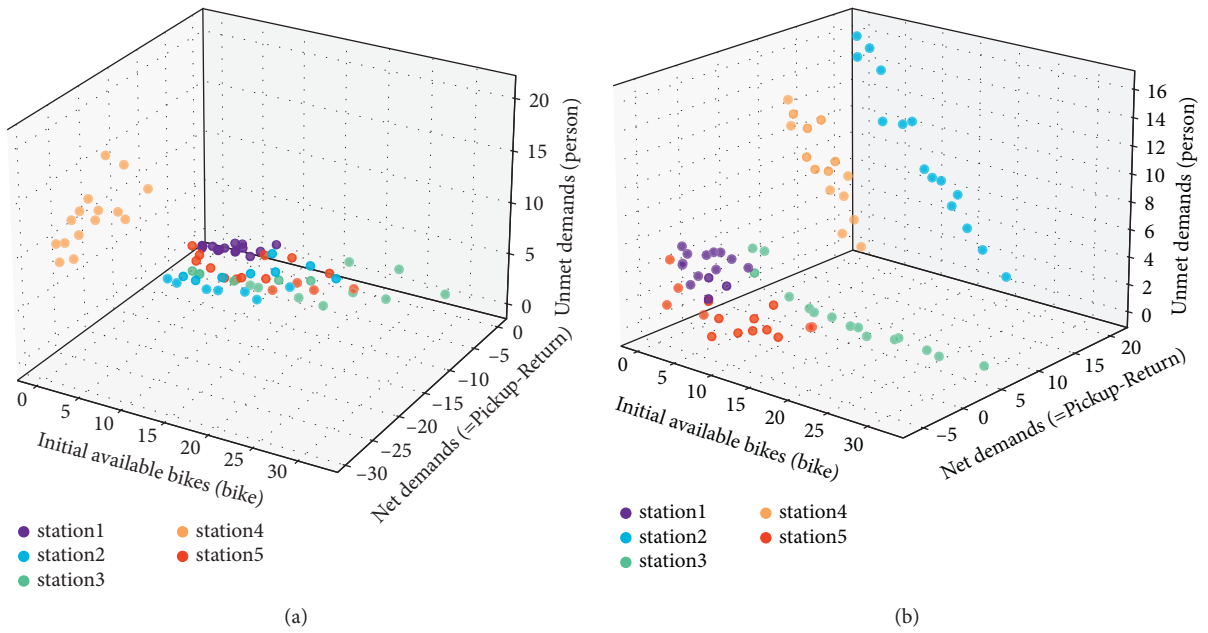

Figure 6: Relationship among initial inventory level, net demands, and total unmet demands. (a) Morning peak; (b) evening peak.

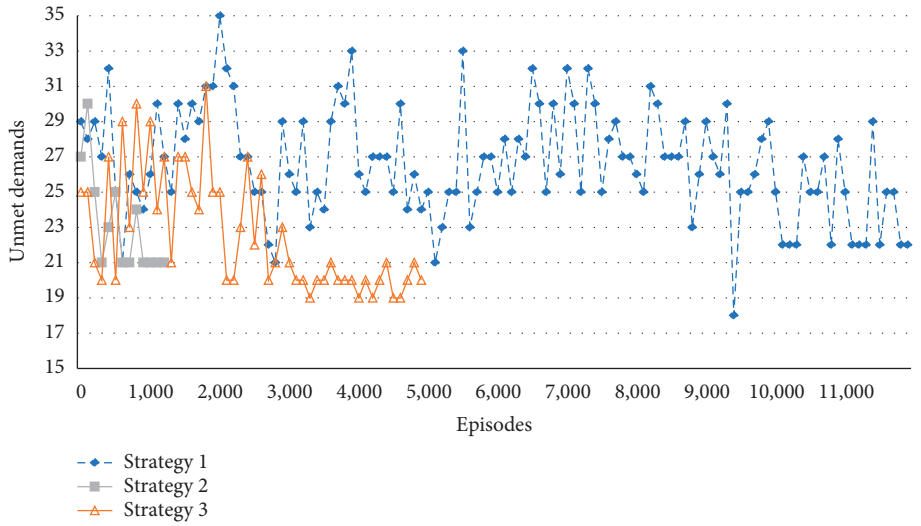

Figure 7: Performance analysis in deterministic demand contexts.

requires more trial and error to explore all stations. Strategies 2 and 3, which search only a few stations, converge faster than Strategy 1 owing to the size reduction in the search area. Among these strategies, Strategy 3 had the lowest convergence.

The inefficient movement was observed in Strategy 1. As all stations are candidates for routing decisions, the delivery amount is reduced. In fact, it is impossible to serve all stations within a limited working period. Operators in the PBS system serve only about 20 stations for nine hours (working time) owing to handling broken bicycles or citizens' complaints. Therefore, a strategy is required to select the station that requires the most rebalancing. Compared to Strategy 1, Strategy 2, which searches for nearby stations, can reduce the travel time; however, Strategy 2 cannot reduce the total unmet demand by failing to serve distant stations requiring urgent rebalancing treatments. Accordingly, Strategy 2 may be applicable for SBRP that minimizes the total travel time, however, it is not suitable for DBRP, where pickup and return demands change in real time. In Strategy 3, the operator visits the depot again during rebalancing to withdraw bicycles for more delivery. On a weekday morning, most stations in the Yeouido district have more return demands than pickup demands because of commuting trips. Since the analysis duration was as short as 2 h, the rebalancing effect could not be identified after the analysis period. If the analysis period is further extended, then Strategy 3 will exhibit better performance than the other prioritizing strategies.

Table 4: Mean and standard deviation by prioritizing strategies.

Strategy	50%		100%		150%	
	Mean	Standard deviation	Mean	Standard deviation	Mean	Standard deviation
Strategy 1: all stations	92.95	1.74	93.54	4.11	94.16	2.66
Strategy 2: near stations	94.06	1.08	92.27	1.08	91.76	3.5
Strategy 3: stations with large errors	98.56	1.73	93.70	1.97	90.87	1.45
Constraint-free strategy	96.46	2.19	96.93	4.99	96.85	3.84

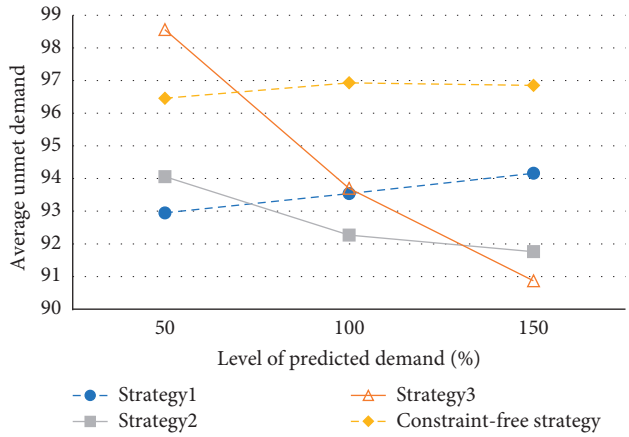

Figure 8: Average unmet demand by strategies.

5.3.2. Stochastic Demand Contexts. Three demand scenarios are considered for analysis in the stochastic demand context: predicted demand, half of the predicted demand, and 1.5 times the predicted demand.

Accordingly, we trained the agent 5,000 times on stochastic demand, and then tested it 100 times on observed demand situations. The mean and standard deviation of the unmet pickup demand and return demand by prioritizing strategies are presented in Table 4 and Figure 8, respectively. Strategies 1 and 2 did not exhibit a difference in terms of the level of predicted demand; however, Strategy 3 exhibited a lower unmet demand as the level of predicted demand increased. A Poisson distribution is assumed for the observed demand during the training process; therefore, the higher the level of predicted demand, the greater the difference between the observation and the predicted value, thereby making it easier for the agent to determine which station has a greater gap between the observations and predictions.

In addition to the three strategies, a constraint-free strategy was compared, where the agent selects both the next station to be visited and the number of (un)loading bicycles at the current station by itself. Accordingly, the average unmet demand of the constraint-free strategy was 2.55–6.58% higher than that of the other strategies, except for Strategy 3, which had a predicted demand of 50%. The standard deviation is up to two times higher than that of the suggested strategy. These results indicate that the use of the suggested strategies is suitable for deriving an efficient solution within a limited time.

6. Conclusions

In this study, RTDP and RL methods were developed in the context of a dynamic PBS system with stochastic demands. The analysis was performed on user demand patterns that differed by period, based on historical pickup data. The forecasted demand was constructed stochastically using the random forest technique. The movements of cargo vehicles were analyzed over time by introducing the MDP. We compared the performance of the proposed strategies with other benchmark strategies by using the developed model and algorithm. It was found that the strategy that considers the prediction error resulted in better performance than other benchmark strategies. More specifically, the strategy of focusing on stations with more realistic fluctuations in the dynamic factors exhibited an improved rebalancing effect. Our proposed strategy provides a guideline for selecting the next visiting stations; for instance, among stations with a high error of predicted demand, as a minimum constraint, the proposed strategy can be used to find a practical solution with limited resources and a short time between the demand forecasting and rebalancing operations. A future research direction will be to find a way to increase the efficacy and accuracy of a constraint-free RL method.

Most existing research on the rebalancing problem of public bicycles has adopted the SBRP, in which the rebalancing process was assumed to occur primarily at night. Even though studies using the DBRP have recently increased slightly, user demands have been regarded as constant or deterministic values based on historical usage data. In this study, the cargo vehicle route and number of bicycles to load or unload were determined to minimize the unmet demands by considering the stochastic forecasted demands. This study proposes a more efficient and reliable prioritizing strategy with operational policy options that can be practically applied under relevant conditions in the field.

The main contributions of this study are as follows: in this study, an MDP-based DP method is developed that simulates PBS system rebalancing with stochastic demands, and this long-term strategy can consider future stochastic demands. A statistical distribution of demand is assumed through a statistical test based on historical demand data to reflect the demand uncertainty. Moreover, the operator's action in each state is not determined through simulations but by an algorithm. An approximate dynamic programming method is developed to reduce the computational effort required for DP calculations owing to the large state space and action space. In the present analysis, a policy that

effectively reduces the operators' action candidates and derives implications through performance comparisons for each policy is developed. The action space can be reduced by responding proactively to unexpected fluctuations.

This study considered a dock-based system with discrete stations; however, future research can apply the methodology proposed in this study to a dockless system where pickup and return points are not restricted to certain stations but extend to a continuous plane. In a dockless system, clustered areas with certain resolutions are considered instead of fixed points, and the demand in the cluster can be predicted using a deep learning approach, such as a graph neural network [35]. Extensive research is required to address the two remaining issues in demand forecasting for public bicycles: the accuracy of the forecasted demand itself and the estimation of true demand. If the demand forecasting accuracy is high, the cargo vehicle movement is closer to the movement in the hindsight problem where the agent knows the user demands in advance. Meanwhile, unmet demand can become more realistic if an estimated true demand is used rather than an observed demand. This study tested rebalancing strategies using limited resources within a relatively small spatial scope. Future research can consider expanding the range of additional inputs of resources, such as cargo vehicles and operators. Finally, the reward function in this study was set as an unmet demand, as an example of an operational goal. However, depending on the priority given by the operating agencies to other metrics such as the travel distance of cargo vehicles or overall operational cost, rewards can be determined by considering the metrics collectively under the same framework and methodologies.

Data Availability

Data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This paper is a revision of the first author's doctoral dissertation from Seoul National University. This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (2020R1F1A1061802) and Ministry of Education (2020R1A6A1A03045059).

References

[1] Y.-H. Seo, "A Dynamic Rebalancing Strategy in Public Bicycle Sharing Systems Based on Real-Time Dynamic Programming and Reinforcement Learning," 2020, https://hdl.handle.net/10371/169096.

[2] R. B. Nath and T. Rambha, "Modelling methods for planning and operation of bike-sharing systems," *Journal of the Indian Institute of Science*, vol. 99, no. 4, pp. 621–645, 2019.

[3] M. D. Gleditsch and K. Hagen, "A Column Generation Heuristic for the Dynamic Rebalancing Problem in Bike Sharing Systems," Trondheim, 2018, https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2575682.

[4] O. Alagoz, H. Hsu, A. J. Schaefer, and M. S. Roberts, "Markov decision processes: a tool for sequential decision making under uncertainty," *Medical Decision Making*, vol. 30, no. 4, pp. 474–483, 2010.

[5] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, London, 2018.

[6] J. Brinkmann, M. W. Ulmer, and D. C. Mattfeld, "Dynamic lookahead policies for stochastic-dynamic inventory routing in bike sharing systems," *Computers & Operations Research*, vol. 106, pp. 260–279, 2019.

[7] C. Contardo, C. Morency, and L.-M. Rousseau, "Balancing a Dynamic Public Bike-Sharing System," Cirrelt, 2012, https://www.cirrelt.ca/DocumentsTravail/CIRRELT-2012-09.pdf.

[8] D. Zhang, C. Yu, J. Desai, H. Y. K. Lau, and S. Srivathsan, "A time-space network flow approach to dynamic repositioning in bicycle sharing systems," *Transportation Research Part B: Methodological*, vol. 103, pp. 188–207, 2017.

[9] T. Wang, "Solving Dynamic Repositioning Problem for Bicycle Sharing Systems: Model, Heuristics, and Decomposition," Austin, 2014, https://repositories.lib.utexas.edu/handle/2152/28258.

[10] A. Fernández, H. Billhardt, S. Timón, C. Ruiz, Ó. Sánchez, and I. Bernabé, "Balancing strategies for bike sharing systems," in *Proceedings of the International Conference on Agreement Technologies*, vol. 43, pp. 208–222, Bergen, Norway, April 2019.

[11] F. Chiariotti, C. Pielli, A. Zanella, and M. Zorzi, "A dynamic approach to rebalancing bike-sharing systems," *Sensors*, vol. 18, no. 2, pp. 512–22, 2018.

[12] B. Legros, "Dynamic repositioning strategy in a bike-sharing system; how to prioritize and how to rebalance a bike station," *European Journal of Operational Research*, vol. 272, no. 2, pp. 740–753, 2019.

[13] J. Brinkmann, M. W. Ulmer, and D. C. Mattfeld, "Short-term strategies for stochastic inventory routing in bike sharing systems," *Transportation Research Procedia*, vol. 10, pp. 364–373, 2015.

[14] G. Li, N. Cao, P. Zhu et al., "Towards smart transportation system," *Journal of Organizational and End User Computing*, vol. 33, no. 3, pp. 35–49, 2021.

[15] I. Arel, C. Liu, T. Urbanik, and A. G. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," *IET Intelligent Transport Systems*, vol. 4, no. 2, pp. 128–135, 2010.

[16] M. A. Khamis and W. Gomaa, "Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework," *Engineering Applications of Artificial Intelligence*, vol. 29, pp. 134–151, 2014.

[17] H. Wei, G. Zheng, H. Yao, and Z. Li, in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2496–2505, London, UK, July 2018.

[18] M. Zolfpour-Arokhlo, A. Selamat, S. Z. Mohd Hashim, and H. Afkhami, "Modeling of route planning system based on Q value-based dynamic programming with multi-agent reinforcement learning algorithms," *Engineering Applications of Artificial Intelligence*, vol. 29, pp. 163–177, 2014.

[19] J. J. Q. Yu, W. Yu, and J. Gu, "Online vehicle routing with neural combinatorial optimization and deep reinforcement

learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3806–3817, 2019.

[20] L. An, C. Ren, Z. Gu, Y. Wang, and Y. Gao, "Rebalancing the car-sharing system: a reinforcement learning method," in *Proceedings of the 2019 IEEE 4th International Conference on Data Science in Cyberspace*, pp. 62–69, Hangzhou, China, June 2019.

[21] Y. Li, Y. Zheng, and Q. Yang, in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1724–1733, London, UK, July 2018.

[22] L. Pan, Q. Cai, Z. Fang, P. Tang, and L. Huang, "A deep reinforcement learning framework for rebalancing dockless bike sharing systems," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 1393–1400, 2019.

[23] C. Mao, Y. Liu, and Z.-J. Shen, "Dispatch of autonomous vehicles for taxi services: a deep reinforcement learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 115, Article ID 102626, 2020.

[24] L. Zhang, K. Li, C. Li, and K. Li, "Bi-objective workflow scheduling of the energy consumption and reliability in heterogeneous computing systems," *Information Sciences*, vol. 379, pp. 241–256, 2017.

[25] J. Chen, K. Li, K. Li, P. S. Yu, and Z. Zeng, "Dynamic bicycle dispatching of dockless public bicycle-sharing systems using multi-objective reinforcement learning," *ACM Transactions on Cyber-Physical Systems*, vol. 5, no. 4, pp. 1–24, 2021.

[26] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, New Jersey, 2014.

[27] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, John Wiley & Sons, New Jersey, 2014.

[28] L. P. King, *Crack the Code: Understanding Safety Stock and Mastering its Equations,"* pp. 33–36, APICS Magazine, July/August 2011.

[29] Z. Yang, J. Hu, Y. Shu, P. Cheng, J. Chen, and T. Moscibroda, "Mobility modeling and prediction in bike-sharing systems," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services - MobiSys '16*, pp. 165–178, Singapore, June 2016.

[30] Y.-H. Seo, S. Yoon, D.-K. Kim, S.-Y. Kho, and J. Hwang, "Predicting demand for a bike-sharing system with station activity based on random forest," *Proceedings of the Institution of Civil Engineers - Municipal Engineer*, vol. 174, no. 2, pp. 97–107, 2021.

[31] J.-H. Cho, Y.-H. Seo, and D.-K. Kim, "Efficiency comparison of public bike-sharing repositioning strategies based on predicted demand patterns," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2675, no. 11, pp. 104–118, 2021.

[32] A. G. Barto, S. J. Bradtke, and S. P. Singh, "Learning to act using real-time dynamic programming," *Artificial Intelligence*, vol. 72, no. 1–2, pp. 81–138, 1995.

[33] D. Silver, "Lecture 7: Policy Gradient," 2015, https://blogs.cuit.columbia.edu/zp2130/files/2019/04/pg.pdf.

[34] W. Lee, H. Yang, G. Kim, Y. Lee, and E. Lee, *Reinforcement Learning with Python and Keras*, Wikibooks, Paju, 2017.

[35] J. Chen, K. Li, K. Li, P. S. Yu, and Z. Zeng, "Dynamic planning of bicycle stations in dockless public bicycle-sharing system using gated graph neural network," *ACM Transactions on Intelligent Systems and Technology*, vol. 12, no. 2, pp. 1–22, 2021.