# A* Search Algorithm for an Optimal Investment Problem in Vehicle-Sharing Systems

# A* Search Algorithm for an Optimal Investment Problem in Vehicle-Sharing Systems

Ba Luat Le[1], Layla Martin[2], Emrah Demir[3], and Duc Minh Vu[1(✉)]

[1] ORLab and Faculty of Computer Science, Phenikaa University, Hanoi, Vietnam
minh.vuduc@phenikaa-uni.edu.vn
[2] Department of Industrial Engineering and Eindhoven AI Systems Institute, Eindhoven University of Technology, Eindhoven, The Netherlands
[3] Cardiff Business School, Cardiff University, Cardiff, UK

**Abstract.** We study an optimal investment problem that arises in the context of the vehicle-sharing system. Given a set of locations to build stations, we need to determine *i*) the sequence of stations to be built and the number of vehicles to acquire in order to obtain the target state where all stations are built, and *ii*) the number of vehicles to acquire and their allocation in order to maximize the total profit returned by operating the system when some or all stations are open. The profitability associated with operating open stations, measured over a specific time period, is represented as a linear optimization problem applied to a collection of open stations. With operating capital, the owner of the system can open new stations. This property introduces a set-dependent aspect to the duration required for opening a new station, and the optimal investment problem can be viewed as a variant of the Traveling Salesman Problem (TSP) with set-dependent cost. We propose an A* search algorithm to address this particular variant of the TSP. Computational experiments highlight the benefits of the proposed algorithm in comparison to the widely recognized Dijkstra algorithm and propose future research to explore new possibilities and applications for both exact and approximate A* algorithms.

**Keywords:** Autonomous Mobility on-demand · vehicle-sharing · traveling salesman problem · A* algorithm

## 1 Introduction

Mobility on demand (MoD) is a rapidly growing market[1]. With the advanced technology of autonomous vehicles, Autonomous Mobility on demand (AMoD) is becoming increasingly popular because it alleviates some operational difficulties of MoD. The global market for autonomous mobility is projected to grow from 5 billion USD (in 2019) to 556 billion USD (in 2026)[2], promising safety (94% of accidents caused by human factors), increased performance, improved efficiency, and more affordable services.

[1] https://www.alliedmarketresearch.com/mobility-on-demand-market.
[2] https://www.alliedmarketresearch.com/autonomous-vehicle-market.


M. H. Hà et al. (Eds.): CSoNet 2023, LNCS 14479, pp. 162–173, 2024.
https://doi.org/10.1007/978-981-97-0669-3_16

Although auto manufacturers and major technology firms have the resources to quickly establish an AMoD system, smaller operators of shared mobility and public authorities may encounter challenges in securing enough initial capital to launch the service with a sufficient fleet[3]. Consequently, small companies start to operate in a smaller region, as studied in the literature on optimal service region design, e.g., [7,8]. As operators accumulate profits, they can gradually acquire more vehicles and expand their active sites. This research considers such a *refinancing* model of the AMoD system, where the operator aims to achieve the desired service area and the size of the fleet as quickly as possible.

The existing literature covers a spectrum of topics related to AMoD systems, including aspects such as vehicle-sharing system operations, strategic decision-making, and regulatory and subsidy considerations. Relevant sources can be found in works such as [3,5–7]. To the best of our knowledge, the question of what is the optimal investment sequence to build an AMoD has not been addressed yet. In this research, we consider an AMoD with a target service area, as well as a current set of open stations. The operator decides on the sequence in which they open the stations. The more profit they make, the faster they can open new stations.

In the following sections, we address the above questions and then analyze the performance of our proposed algorithm. To do so, we review publications close to our research in Sect. 2. Next, we present the problem statement and related formulations in Sect. 3. Section 4 presents our solution approach based on the A* search algorithm. Numerical experiments and some promising results are presented and analyzed in Sect. 5. Finally, Sect. 6 concludes and points out further research directions based on the current research.

## 2 Literature

This section provides a brief literature review on AMoD systems. Research into the operation and planning of AMoD systems encompasses a range of questions. However, its main emphasis lies in optimizing an existing vehicle-sharing network. Regarding fleet optimization, we can refer to [5,6,10,12]. George and Xia [6] study a fleet optimization problem in a closed queue network. This work suggests basic principles for the design of such a system. Nair and Miller-Hooks [13] use the equilibrium network model to find the optimal configuration of a vehicle-sharing network. The solutions to the model explain the correctness of the equilibrium condition, the trade-offs between operator and user objectives, and the insights regarding the installation of services. Freund et al. [5] address how to (re-)allocate dock capacity in vehicle-sharing systems by presenting mathematical formulations and a fast polynomial-time allocation algorithm to compute an optimal solution. Lu et al. [10] consider the problem of allocating vehicles to service zones with uncertain one-way and round-trip rental demand.

Regarding policies, Martin et al. [11] conclude that the use of driverless vehicles and human-driven vehicles can improve profits and operators can gain new unprofitable markets for them. The authors propose a model and an algorithm to find maximum profit

[3] https://www.weforum.org/agenda/2021/11/trends-driving-the-autonomous-vehicles-industry/.

while considering driverless and human-driven vehicles. Hao and Martin [7] present a model that studies the impact of regulations on the decisions of vehicle-sharing operators and measures the efficiency and effectiveness of these regulations. The results show that the interdependencies between regulations and societal welfare indicators are non-trivial and possibly counterintuitive.

To conclude, we observe that all the research so far has tried to address different questions with the goal of optimizing an already established vehicle-sharing network. However, the question of how to establish new stations and acquire new vehicles has not been addressed yet. In the following, we introduce an optimization problem aimed at identifying the optimal sequence for station establishment and the fleet size required to reach the end state where all stations are operational in the shortest possible time.

## 3 Problem Statement and Formulation

We study an optimal investment strategy for an AMoD (Autonomous Mobility-on-Demand) operator to increase their fleet size and operating area. The AMoD operator's business area comprises stations, ($\mathcal{R} : \{1, ..., R\}$). "Station" can also refer to a virtual location, e.g., the center of a region in a free-floating system. The operating station $i$ incurs an initial cost $c_i^b$ related to construction, permits, or marketing. Some stations are already open, and profits will be collected from already open stations to increase the budget for new stations. The operator incrementally grows the fleet to reach the optimal size promptly while ensuring acceptable service levels within a gradually expanding operating area.

At a given open station $i$, customers begin their journeys to a different station $j$. When a station is not operational, customers intending to start or complete their journeys there can opt for a neighboring station. Customer arrivals are modeled by a Poisson distribution with an arrival rate denoted as $\lambda_{ij}$, and 0 when at least on of the stations is closed. The travel times between the stations are exponentially distributed, with an average of $1/\mu_{ij}$, where $\mu_{ij}$ denotes the return rate. These arrival and return rates remain constant and are determined solely by whether stations $i$ and $j$ are open.

The operator determines the fleet size $n$ at any given time, allowing it to grow during expansion. Each new vehicle acquisition comes with a procurement cost of $c^p$. The fleet size must be large enough to serve at least a fraction $\alpha$ of all customers, meeting the minimum service level requirement for the AMoD system. Throughout the development of the AMoD service, it is crucial to keep the service level constant to offset the potential learning effects that could deter customers from using the service [4]. To maintain the service level, the operator can rebalance vehicles between stations, incurring a cost of $c_{ij}^r$. The operator receives a contribution margin of $\delta_{ij}$ for each served customer traveling from station to station, representing the payoff minus direct operating costs such as fuel and periodic repairs.

Consequently, this problem involves two decision-making components: establishing the optimal investment plan, which includes timing, locations, and quantity for opening new stations and vehicle acquisition, and overseeing fleet operations, which includes vehicle rebalancing. The model for determining the optimal fleet size and an algorithm for determining investment sequence are introduced in the subsequent sections.

### 3.1 Semi-Markov Decision Process for Determining the Optimal Fleet Size

We see the optimal investment scheduling problem of AMoD operators as a semi-Markov decision process (SMDP) due to the nature of the investment problem. In an SMDP, the system's state evolves according to a semi-Markov process, and the decision-maker selects actions based on the current state.

**States.** Each state $s \in \mathcal{S}$ describes the current fleet of size $n$ and the currently open stations, given by $x_i = 1$ if station $i \in \mathcal{R}$ is open, 0 otherwise.

$$s = \langle n, x_1, \ldots, x_R \rangle$$

Each state $s$ is associated with an operational profit $p(s)$ per period, which is calculated by subtracting the rebalancing costs from the contribution margins and an acquisition cost $c(s)$ related to the procurement cost of all vehicles and the cost incurred due to the opening of the station. Apparently, we only need to consider states with positive operational profit in our investment scheme. Regarding this point, the set of states with positive operational profit and the starting state is denoted as $\mathcal{S}$. Also, if a state $s'$ contains all open stations in a state $s$, we can easily see and prove that $p(s') \geq p(s)$. For referencing the fleet size and open stations of a specific state $s$, the notation $n(s)$ and $x_i(s)$ are utilized, respectively. Then, the value of $c(s)$ is determined as follows:

$$c(s) = n(s) \cdot c^p + \sum_{i \in \mathcal{R}} x_i(s) c_i^b$$

**Actions.** Actions refer to the operator's procurement decision, resulting in a state transition to the target state $t \in \mathcal{S}$. Every state $s \in \mathcal{S}$ allows transitions to all other states such that no stations are being closed, that is, $s \to t$ exists if $x_i(s) \leq x_i(t)\ \forall i$.

The time $\tau(s,t)$ necessary for a state transition from state $s$ to a state $t$ depends on the operational profit $p(s)$ and the necessary investment volume $C(s,t)$ where

$$C(s,t) = c(t) - c(s) = (n(t) - n(s)) \cdot c^p + \sum_{i \in \mathcal{R}} (x_i(t) - x_i(s)) \cdot c_i^b.$$

Given that we do not consider partial states (e.g., a state without optimal fleet size), this means that $p(s)$ is considered the maximum profit corresponding to state $s$, and the optimal decision is to transition to the next state as soon as possible. Thus, $\tau(s,t) = \frac{C(s,t)}{p(s)}$.

We notice that if $|t| \geq |s| + 2$, it is more advantageous to transition to an immediate state $s'$ where $|s| < |s'| < |t|$ because $\frac{C(s,t)}{p(s)} \geq \frac{C(s,s')}{p(s)} + \frac{C(s',t)}{p(s')}$ due to the fact that $p(s) \leq p(s')$ and $C(s,t) = C(s,s') + C(s',t)$. Therefore, we only need to consider actions between two consecutive states in any optimal investment scheme.

### 3.2 A Model for Calculating Optimal Profit and Minimum Acquisition Cost

To compute the operational profit $p(s)$ per state $s \in \mathcal{S}$, we formulate the rebalancing problem as an open-queueing network (in line with, e.g., [3,7,8,11]), and optimize

over it to maximize operational profits. Given a set of available stations, the model determines the necessary size of the fleet to reach the level of service and rebalance. Since we want to maximize profit and minimize the corresponding acquisition cost, our objective function is hierarchical since we optimize the second objective after minimizing the first objective.

To start, we denote $f_{ij}, e_{ij}$ $(i \neq j)$ as the number of occupied and empty vehicles traveling from $i$ to $j$ and $e_{ii}$ as the number of idle vehicles currently parked at station $i$. To determine the maximum operational profit per period for state $s$, we solve (1) - (7) for all opening stations in $R_s = \{i \in \mathcal{R} | x_i(s) = 1\}$. The mathematical formulation is expressed as follows:

$$P(obj_1, obj_2) = \Big(\max \ \alpha\Big(\sum_{i\in R_s}\sum_{j\in R_s} \lambda_{ij}\delta_{ij} - \sum_{i\in R_s}\sum_{j\in R_s} c^r_{ij}\mu_{ij}e_{ij}\Big), \min \ \Big(n \cdot c^p + \sum_{i\in R_s} c^b_i\Big)\Big) \quad (1)$$

subject to

$$\lambda_{ij} = \mu_{ij} f_{ij}, \qquad \forall i, j \in R_s \quad (2)$$

$$\sum_{j\in R_s\setminus\{i\}} \mu_{ji} e_{ji} \leq \sum_{j\in R_s\setminus\{i\}} \lambda_{ij}, \qquad \forall i \in R_s \quad (3)$$

$$\sum_{j\in R_s} \lambda_{ij} + \sum_{j\in R_s} \mu_{ij} e_{ij} = \sum_{j\in R_s} \mu_{ji} e_{ji} + \sum_{j\in R_s} \lambda_{ji}, \qquad \forall i \in R_s \quad (4)$$

$$\frac{\alpha}{1-\alpha} \leq e_{ii}, \qquad \forall i \in R_s \quad (5)$$

$$\sum_{i,j\in R_s} \big(e_{ij} + f_{ij}\big) = n, \quad (6)$$

$$e_{ij}, f_{ij} \geq 0, \qquad \forall i, j \in R_s \quad (7)$$

The objective function (1) maximizes profit by dividing the contribution margin of all served customers by rebalancing costs, multiplied by availability $\alpha$, and minimizing set-up fees. Constraints (2) - (4) linearize flow constraints in queueing networks, almost directly follow from [3] and requiring the system to achieve a service level of at least $\alpha$, eliminating any upper bound on demand, unlike [3]. Constraints (5) set the required safety stock, following the fixed population mean approximation in open queueing networks due to [14]. Constraints (6) bound fleet size, and constraints (7) defined the domain.

## 4 Solution Approach

It is important to note that in our problem, the optimal time for opening a new station depends on profits from existing stations, resulting in a set-dependent cost. The exponential growth of these sets makes mathematical representations potentially too complex, making contemporary solvers unsuitable for modeling and solving this formulation.

We can consider the investment problem as a variant of the well-known Traveling Salesman Problem (TSP) with set-dependent travel costs. Taking into account a permutation $(u_1, u_2, .., u_n)$ that presents an order that the stations are opened. Each subpath $(u_1, u_2, .., u_i)$ is assigned a state $s_i$ where $x_k(s_i) = 1$ if $u_j = k$ for some $j = 1..i$. The cost between two consecutive states, $s_i$ and $s_{i+1}$, is calculated using the formulations in Sect. 3.1, which depend on the set of open stations in $s_i$. In other words, it is a set-dependent cost function. While there is much research for TSP in general and several studies on level-dependent travel cost TSP [1,2] in particular (the cost associated with each city depends on the index of that city in the solution), our cost function makes the problem cannot be modeled with formulations similar to the ones for TSPs.

### 4.1 Heuristic Strategy for A* Algorithm

We model our investment problem as a shortest path problem. Consider a graph $G = (V, A)$ where each node $n_s \in V$ corresponds to the state $s$. Each arc $(n_s, n_{s'}) \in A$ corresponds to a feasible action between two consecutive states $s$ and $s'$ with cost $C(s, s')$. Finding the shortest investment time is equivalent to finding the shortest path from node $n_{s_0}$ to node $n_{s_f}$ where $s_0$ and $s_f$ are the initial state and the final state, respectively. Since we can define a 1–1 mapping between $s$ and $n_s$, we subsequently use $s$ instead of $n_s$ to simplify the notation.

To solve this shortest-path problem, we rely on the A* algorithm. Given a state $s$, unlike the classic Dijkstra algorithm, which only evaluates the cost of the shortest path $g(s)$ from the source $s_0$ to $s$, A* also evaluates the cost $h(s)$ from $s$ to the final state $s_f$, and the cost for each node $s$ is then $f(s) = g(s) + h(s)$ instead of $g(s)$. The A * algorithm can always find the shortest path from $s_0$ to $s_f$ if $h(s)$ does not exceed the cost of the shortest path from $s$ to $s_f$ for any $s$. Otherwise, A* becomes a heuristic algorithm.

**Simple Heuristic for A*.** We start with some of the simplest heuristics for A*. Given that the current, next, and final states are $s, s'$ and $s_f$, the cost of the shortest path from $n_0$ to $s'$, $g(s')$, is $g(s') = g(s) + \frac{c(s')-c(s)}{p(s)}$. Several simple ways to calculate $h(s')$ are as follows (where $eh$ and $ah$ denote exact and approximate heuristics, respectively):

$$eh_1(s') = \frac{c(s_f) - c(s')}{P_{R-1}} \tag{8}$$

$$ah_1(s') = \frac{c(s_f) - c(s')}{p(s')} \tag{9}$$

Heuristic functions (8), (9) underestimate and overestimate the shortest time of the optimal path from $s_0$ to $s_f$ that passes through $s'$. Here, $P_{R-1}$ denotes the maximum profit for any state that has $R - 1$ open stations. Using a linear combination, we obtain other heuristics where $\gamma \in [0, 1]$ is a parameter that can be a fixed constant or dynamically adjusted during the execution of the algorithm. We aim to test whether we can obtain simple heuristics that may not be optimal but can quickly find reasonable solutions.

$$ah_2(s') = \gamma eh_1(s') + (1 - \gamma)ah_1(s') \tag{10}$$

**Stronger Lower Bound Heuristics for A*.** Assume that $s = s_1$ is the current state. Let $s_1, s_2, .., s_k$ be a sequence of states where $s_{i+1}$ is obtained from $s_i$ by adding a new station and $s_k = s_f$ be the final state where all stations are open. The total transition time from state $s_1$ to state $s_k$, $\tau(s_1, .., s_k)$, is:

$$\tau(s_1, .., s_k) = \frac{c(s_2) - c(s_1)}{p(s_1)} + \frac{c(s_3) - c(s_2)}{p(s_2)} + \ldots + \frac{c(s_k) - c(s_{k-1})}{p(s_{k-1})} \tag{11}$$

We denote $\underline{\Delta_c}(s_i)$ as a lower bound of the difference of the acquisition cost $c(s_{i+1}) - c(s_i)$ between two consecutive states $s_i$ and $s_{i+1}$. Let $P_m$ be a state with the maximum profit among all states with $m$ opening stations. We can find the value of $P_m$ by solving the model which is an extended version of (1)–(7) (see online Appendix [9]), which aims to maximize the profit and minimize the corresponding acquisition cost given a fixed number of stations that can be opened. Then, we obtain $p(s_i) \leq P_{|s_i|}, \forall i = 1, \ldots, k$. The values of $P_{|s_i|}$ define an increasing sequence since we open more stations. Therefore, we have $p(s_i) \leq P_{|s_i|} \leq P_{|s_{k-1}|} = P_{R-1}, \forall i = 1, \ldots, k-1$. Given that $c(s_{i+1}) - c(s_i) \geq \underline{\Delta_c}(s_i)$ or $c(s_{i+1}) - c(s_i) - \underline{\Delta_c}(s_i) \geq 0$, therefore, $\forall i = 1, \ldots, k-1$ we have the following.

$$\frac{c(s_{i+1}) - c(s_i)}{p(s_i)} = \frac{\underline{\Delta_c}(s_i) + (c(s_{i+1}) - c(s_i) - \underline{\Delta_c}(s_i))}{p(s_i)} \tag{12}$$

$$= \frac{\underline{\Delta_c}(s_i)}{p(s_i)} + \frac{c(s_{i+1}) - c(s_i) - \underline{\Delta_c}(s_i)}{p(s_i)} \tag{13}$$

$$\geq \frac{\underline{\Delta_c}(s_i)}{P_{|s_i|}} + \frac{c(s_{i+1}) - c(s_i) - \underline{\Delta_c}(s_i)}{P_{R-1}} \tag{14}$$

and consequently:

$$\tau(s_1, \ldots, s_k) \geq \left( \sum_{i=1}^{k-1} \frac{\underline{\Delta_c}(s_i)}{P_{|s_i|}} \right) + \frac{c(s_k) - c(s_1) - \sum_{i=1}^{k-1} \underline{\Delta_c}(s_i)}{P_{R-1}} \tag{15}$$

Inequality (15) gives us a more robust lower bound than the simple one presented in (8).

**Evaluating the Lower Bound $\underline{\Delta_c}(s_1)$.** From (6), we see that with each state $S$, the optimal number of vehicles $n$ is equal to $\sum_{i,j \in S}(e_{ij} + f_{ij})$. Following $obj_2$, when we open a new station, the acquisition cost includes the cost of station setup and the new vehicle acquisition cost. We assume that the difference in acquisition cost between two consecutive states depends on the values $f_{ij}, e_{ij}$ of the new station $i$. With this assumption, the minimum acquisition cost of opening station $i$ from a given state $S$ ($i \notin S$) to obtain maximum profit is determined by $\Delta_c(S, i)$. In other words, $\Delta_c(S, i)$ presents the lower difference in acquisition cost between two consecutive states in which the next state is reached by opening station $i$ from the state $S$.

$$\Delta_c(S, i) = \sum_{j \in S}(e_{ij} + e_{ji} + f_{ij} + f_{ji})c^p + c_b^i \quad \forall i \notin S \tag{16}$$

Then, $c(s_{i+1}) - c(s_i) \geq \min_{o \notin s_i} \Delta_c(s_i, o) \quad \forall i = 1, 2, \ldots, k-1$.

Next, we show how to obtain the lower bound $\Delta_c(s_i, o)$ using Eq. (16). Assuming $T \subset \mathcal{R}$ be the state with any $t$ stations not in $S$, $t < R - |S|$, $S \cap T = \emptyset$. Let $o \notin S \cup T$, and we evaluate $\Delta C(S, t, i)$ - the minimum acquisition cost difference when building a new station $i$ starts from state $S \cup T$ with any state $T$ such that $|T| = t$.

**Underestimate Acquisition Cost.** We rewrite $\Delta C(S, t, i)$ using equation (16) as follows:

$$\Delta C(S, t, i) = \min_{T \subset \mathcal{R}, |T|=t} \sum_{j \in S \cup T} (e_{ij} + e_{ji} + f_{ij} + f_{ji})c^p + c_i^b \tag{17}$$

and since $e_{ij}, e_{ji}$ and $c^p$ are non-negative, we have

$$\Delta C(S, t, i) \geq \min_{T \subset \mathcal{R}, |T|=t} \sum_{j \in S \cup T} (f_{ij} + f_{ji})c^p + c_i^b \tag{18}$$

Since $c_i^b + \sum_{j \in S}(f_{ij} + f_{ji})c^p$ is a constant, we will develop a lower bound for the sum $\sum_{j \in T}(f_{ij} + f_{ji})c^p$. Apparently, $\sum_{j \in T}(f_{ij} + f_{ji})c^p$ cannot be smaller than the sum of $|T|$ smallest values of $(f_{ij} + f_{ji})c^p$ where $j \notin S \cup \{i\}$. Therefore, we developed the Algorithm 1 to evaluate a lower bound of $\Delta C(S, t, i)$.

**Algorithm 1.** Lower bound evaluation of acquisition cost

**Require:** $S, i, t$
**Ensure:** A lower bound of $\Delta C(S, t, i)$
1: Let $\alpha \leftarrow c_i^b + \sum_{j \in S}(f_{ij} + f_{ji})c^p$
2: Sort $(f_{ij} + f_{ji})_{j \in \mathcal{R} \setminus (S \cup \{i\})}$ increasingly.
3: Let $(f_{ij_1} + f_{j_1 i}) \leq (f_{ij_2} + f_{j_2 i}) \leq \ldots \leq (f_{ij_k} + f_{j_k i}) \leq \ldots$ be the array after sorting.
4: Let $\beta \leftarrow \sum_{k=1}^{t}(f_{ij_k} + f_{j_k i})c^p$
5: **Return** $\alpha + \beta$

Using Algorithm 1, we have that

$$\Delta_c(s_i, o) \geq \Delta C(s_1, i-1, o) \geq c_o^b + \sum_{j \in s_1}(f_{oj} + f_{jo})c^p + \sum_{k=1}^{i-1}(f_{oj_k} + f_{j_k o})c^p \tag{19}$$

and consequently,

$$c(s_{i+1}) - c(s_i) \geq \min_{o \notin s_i} \Delta_c(s_i, o) \tag{20}$$

$$\geq \min_{o \notin s_i} \left( c_o^b + \sum_{j \in s_1}(f_{oj} + f_{jo})c^p + \sum_{k=1}^{i-1}(f_{oj_k} + f_{j_k o})c^p \right) \tag{21}$$

Use $\underline{\Delta_c}(s_i) = \min_{o \notin s_i} \left( c_o^b + \sum_{j \in s_1}(f_{oj} + f_{jo})c^p + \sum_{k=1}^{i-1}(f_{oj_k} + f_{j_k o})c^p \right)$ in inequality (15), we obtain a lower bound heuristic for A*, called $eh_2$, which is stronger than the simple one $eh_1$. However, we need to solve it online.

We obtain a weaker lower bound version of $eh_2$ by fixing $s_1$, e.g., to the initial state $s_0$. Still, this strategy may reduce the total running time since the value of $\underline{\Delta_c}(s_i)$ needs to be calculated only once, while with $eh_2$, we will calculate $\underline{\Delta_c}(s_i)$ for each extracted state $s = s_1$ from the queue. Let $eh_3$ be this lower bound heuristic.

## 5 Numerical Experiments

In this section, we present the numerical design and then report the experiment results of exact and heuristic algorithms to find an optimal schedule investment. The algorithm and formulations were written in C++, and the MILP models were solved by CPLEX 22.1.1. The experiments were run on an AMD Ryzen 3 3100 machine with a 4-core processor, 3.59 GHz, and 16 GB of RAM on a 64-bit Windows system.

### 5.1 Numerical Design

We conducted experiments on randomly generated datasets, following a similar approach as Martin et al. [7]. To model the real-world transportation network structure, our datasets vary in size ($R \in \{7, 9, 16, 19, 25\}$ and geographic distributions of station locations, including circular (C), hexagonal (H), and quadratic (Q) layouts. The methodology for generating data and configuring model parameters is elucidated in the online Appendix [9].

The investment starts with a set of initially open stations. We assume that initially, there is a budget of $B = 10000$, optimally utilized to construct the initial stations to maximize the initial profit. With smaller instances (less than 10 stations), we use a dynamic budget of $500 \times R$ to avoid opening too many stations in the initial state. We simulate this process through a formulation which is an extended version of (1)–(7) with additional budget constraints, detailed in online Appendix [9]. With this budget, the initial state has $5 - 7$ open stations for larger instances and 2–3 for instances with fewer than 10 stations. Then, the A* algorithm will find an optimal investment plan starting from the initial state with a certain number of already opened stations obtained from the formulations.

### 5.2 Results

In the following, we assess the following two points:

1. We compare the performance of the exact A* heuristics and Dijkstra algorithm based on the execution time, the number of states explored, and the number of states remaining in the priority queue.
2. We compare the performance of approximate A* heuristics in terms of optimal gap and execution time.

Table 1 analyzes the performance of the exact A * algorithms and the Dijkstra algorithm by reporting their running time in seconds (column Time (s)), the number of nodes extracted by the A* algorithm (column Exp.), and the number of nodes still in the queue (column Rem.) with the optimal value (column Opt.) obtained from all exact algorithms.

The experiments show that datasets with imbalanced arrival rates take longer to open stations due to decreased profit margins. The strongest lower bound heuristic, $eh_2$, has the shortest running time, number of expanded nodes, and number of remaining nodes among all exact methods, detailed in Table 1. Using the A* algorithm

**Table 1.** Results of exact A* heuristic and Dijkstra algorithms

| Instance | Opt. | Dijkstra | A* + $eh_1$ | | | A* + $eh_2$ | | | A* + $eh_3$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Time (s) | Exp. | Rem. | Time (s) | Exp. | Rem. | Time (s) | Exp. | Rem. | Time (s) |
| C-7-BAL | 1563.19 | <1 | 17 | 144 | <1 | 12 | 10 | <1 | 12 | 12 | <1 |
| H-7-BAL | 1524.71 | <1 | 17 | 13 | <1 | 11 | 9 | <1 | 12 | 12 | <1 |
| Q-9-BAL | 435.53 | <1 | 33 | 31 | <1 | 14 | 25 | <1 | 20 | 25 | <1 |
| Q-16-BAL | 420.87 | 1 | 392 | 173 | 1 | 227 | 236 | 1 | 243 | 231 | 1 |
| Q-16-IMB | 723.69 | 1 | 340 | 176 | 1 | 150 | 208 | 1 | 170 | 195 | 1 |
| C-19-BAL | 1054.83 | 14 | 2733 | 2060 | 12 | 1453 | 2234 | 9 | 1603 | 2156 | 9 |
| C-19-IMB | 1681.48 | 14 | 2292 | 1473 | 11 | 902 | 1421 | 7 | 1103 | 1378 | 7 |
| H-19-BAL | 1028.32 | 13 | 2303 | 1710 | 11 | 903 | 1568 | 7 | 1149 | 1656 | 8 |
| H-19-IMB | 1833.02 | 14 | 1812 | 1720 | 10 | 592 | 1299 | 5 | 888 | 1385 | 6 |
| Q-25-BAL | 711.31 | 1313 | 140878 | 119407 | 1032 | 35068 | 84174 | 452 | 44551 | 90771 | 508 |
| Q-25-IMB | 1185.37 | 1311 | 124532 | 105743 | 978 | 18440 | 55015 | 328 | 24975 | 61453 | 372 |

with $eh_2$ significantly reduces computation time and vertice exploration compared to underestimating the optimal path's shortest time $eh_1$. The exact heuristic $eh_3$ also provides computational stability without online updates.

We observe that the number of visited vertices and execution time increases exponentially with the number of stations. To find a suitable investment schedule, the researchers experimented with various heuristic approximation approaches in the A* search algorithm. Results in Table 2 showed that larger values of $\gamma$ resulted in better objective values and longer running time. Although these approaches achieve excellent time efficiency and small gaps, they are highly dependent on data and can become less effective when parameter ranges are modified.

Finally, we report the performance of weighted A* variants in Table 3, which multiply the values of $eh_2$ and $eh_3$ by 1.05 or 1.1. The best solutions ensure a gap between optimal and best solutions of at most 5% or 10%. Although slower than the ones mentioned in Table 2, it ensures an optimal gap that the approximate heuristics cannot. The results show that the optimal gap obtained by these approximation algorithms is very small, highlighting the effectiveness of both heuristics.

**Table 2.** Non-bounded approximation algorithms with simple heuristics.

| Instance | Opt. | A* + $ah_1$ | | A* + $ah_2(\gamma = 0.3)$ | | A* + $ah_2(\gamma = 0.5)$ | | A* + $ah_2(\gamma = 0.7)$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | Gap (%) | Time (s) | Gap (%) | Time (s) | Gap (%) | Time (s) | Gap (%) | Time (s) |
| C-7-BAL | 1563.19 | 9.01 | <1 | 4.34 | <1 | 0.00 | <1 | 0.00 | <1 |
| H-7-BAL | 1524.71 | 8.85 | <1 | 4.37 | <1 | 4.37 | <1 | 0.00 | <1 |
| Q-9-BAL | 435.53 | 4.63 | <1 | 1.29 | <1 | 1.29 | <1 | 0.00 | <1 |
| Q-16-BAL | 420.87 | 7.18 | <1 | 3.86 | <1 | 1.78 | <1 | 0.00 | 1 |
| Q-16-IMB | 723.69 | 6.00 | <1 | 3.82 | <1 | 1.47 | <1 | 0.00 | 1 |
| C-19-BAL | 1054.83 | 12.76 | <1 | 7.78 | <1 | 0.94 | <1 | 0.00 | 3 |
| C-19-IMB | 1681.48 | 17.25 | <1 | 9.62 | <1 | 0.33 | <1 | 0.00 | 3 |
| H-19-BAL | 1028.32 | 6.40 | 1 | 2.11 | <1 | 1.20 | <1 | 0.00 | 2 |
| H-19-IMB | 1833.02 | 11.98 | <1 | 9.19 | <1 | 4.48 | <1 | 0.00 | 1 |
| Q-25-BAL | 711.31 | 15.10 | 1 | 10.08 | 1 | 6.84 | 1 | 0.53 | 39 |
| Q-25-IMB | 1185.37 | 11.87 | 1 | 7.28 | 1 | 3.39 | 1 | 1.26 | 15 |

**Table 3.** Bounded approximation algorithms based on stronger lower-bound heuristic

| Instance | Opt. | A* + 1.1 ∗ $eh_2$ | | A* + 1.1 ∗ $eh_3$ | | A* + 1.05 ∗ $eh_2$ | | A* + 1.05 ∗ $eh_3$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | Gap (%) | Time (s) | Gap (%) | Time (s) | Gap (%) | Time (s) | Gap (%) | Time (s) |
| C-7-BAL | 1563.19 | 0.00 | <1 | 0.00 | <1 | 0.00 | <1 | 0.00 | <1 |
| H-7-BAL | 1524.71 | 0.00 | <1 | 0.00 | <1 | 0.00 | <1 | 0.00 | <1 |
| Q-9-BAL | 435.53 | 0.67 | <1 | 0.67 | <1 | 0.00 | <1 | 0.00 | <1 |
| Q-16-BAL | 420.87 | 0.17 | 1 | 0.13 | 1 | 0.13 | 1 | 0.13 | 1 |
| Q-16-IMB | 723.69 | 0.73 | 1 | 0.00 | 1 | 0.00 | 1 | 0.00 | 1 |
| C-19-BAL | 1054.83 | 0.40 | 6 | 0.09 | 6 | 0.09 | 7 | 0.09 | 8 |
| C-19-IMB | 1681.48 | 0.10 | 4 | 0.04 | 5 | 0.04 | 5 | 0.04 | 6 |
| H-19-BAL | 1028.32 | 0.24 | 3 | 0.07 | 4 | 0.07 | 5 | 0.01 | 6 |
| H-19-IMB | 1833.02 | 0.27 | 4 | 0.07 | 4 | 0.14 | 4 | 0.00 | 5 |
| Q-25-BAL | 711.31 | 0.26 | 194 | 0.08 | 241 | 0.04 | 306 | 0.03 | 370 |
| Q-25-IMB | 1185.37 | 0.43 | 117 | 0.09 | 135 | 0.10 | 181 | 0.00 | 241 |

To conclude the section, we observe that for those benchmark instances, exact methods can provide optimal solutions in a reasonable amount of time for those benchmark instances. The proposed lower bound heuristic $eh_2$ beats simple heuristic $eh_1$ and the Dijkstra algorithm. The simple approximate A* heuristic can give quite good results with a small computation time, while the weighted A* heuristic based on the best lower bound heuristic can reduce the computation time and maintain a small optimal gap.

## 6 Conclusion

We have studied an investment problem that arises in the context of autonomous mobility on demand systems. Given some already open stations, the question is to determine the optimal sequence of opening the remaining stations to minimize the total opening time. We modeled this investment problem as a Semi-Markov Decision Process and viewed this problem as a variant of the TSP problem, where the cost between two vertices $s$ and $t$ depends on the set of already visited vertices belonging to the path from the source vertex to vertex $s$. This special cost function makes the problem impossible to model and solve with current mixed-integer solver technology. We then developed and solved this new variant using the A* algorithm. The experiment results show that the A* algorithm can reduce by half the running time of the Dijkstra algorithm and a simple, exact A* algorithm. Regarding the approximate A* search, the result shows that we can obtain reasonable solutions with a small computation effort.

It is still a challenging task to solve larger problems. Therefore, we are developing and testing more robust lower-bound heuristics for exact A* search. Also, we are testing new approximate heuristics for A* search that take ideas from the lower bound heuristics. The initial results show that we can solve larger instances in a shorter time using both methods. Also, the approximate A* heuristic gives similar results to those returned by the exact A* heuristic in many problem instances.

**Acknowledgement.** The work has been carried out partly at the Vietnam Institute for Advanced Study in Mathematics (VIASM). The corresponding author (Duc Minh Vu) would like to thank VIASM for its hospitality and financial support for his visit in 2023.

## References

1. Alkaya, A.F., Duman, E.: Combining and solving sequence dependent traveling salesman and quadratic assignment problems in PCB assembly. Discret. Appl. Math. **192**, 2–16 (2015)
2. Bigras, L.P., Gamache, M., Savard, G.: The time-dependent traveling salesman problem and single machine scheduling problems with sequence dependent setup times. Discret. Optim. **5**(4), 685–699 (2008)
3. Braverman, A., Dai, J.G., Liu, X., Ying, L.: Empty-car routing in ridesharing systems. Oper. Res. **67**(5), 1437–1452 (2019)
4. DeCroix, G., Long, X., Tong, J.: How service quality variability hurts revenue when customers learn: implications for dynamic personalized pricing. Oper. Res. **69**(3), 683–708 (2021)
5. Freund, D., Henderson, S.G., Shmoys, D.B.: Minimizing multimodular functions and allocating capacity in bike-sharing systems. Prod. Oper. Manag. **27**(12), 2346–2349 (2018)
6. George, D.K., Xia, C.H.: Fleet-sizing and service availability for a vehicle rental system via closed queueing networks. Eur. J. Oper. Res. **211**(1), 198–207 (2011)
7. Hao, W., Martin, L.: Prohibiting cherry-picking: regulating vehicle sharing services who determine fleet and service structure. Transp. Res. Part E: Logist. Transp. Rev. **161**, 102692 (2022)
8. He, L., Mak, H.Y., Rong, Y., Shen, Z.J.M.: Service region design for urban electric vehicle sharing systems. Manuf. Serv. Oper. Manage. **19**(2), 309–327 (2017)
9. Le, B.L., Martin, L., Demir, E., Vu, D.M.: A* search algorithm for an optimal investment problem in vehicle-sharing systems (2023). https://arxiv.org/abs/2311.08834
10. Lu, M., Chen, Z., Shen, S.: Optimizing the profitability and quality of service in carshare systems under demand uncertainty. Manuf. Serv. Oper. Manage. **20**(2), 162–180 (2018)
11. Martin, L., Minner, S., Pavone, M., Schiffer, M.: It's all in the mix: technology choice between driverless and human-driven vehicles in sharing systems (2021). Available at SSRN 4190991
12. Nair, R., Miller-Hooks, E.: Fleet management for vehicle sharing operations. Transp. Sci. **45**(4), 524–540 (2011)
13. Nair, R., Miller-Hooks, E.: Equilibrium network design of shared-vehicle systems. Eur. J. Oper. Res. **235**(1), 47–61 (2014)
14. Whitt, W.: Open and closed models for networks of queues. AT&T Bell Lab. Techn. J. **63**(9), 1911–1979 (1984)