Journal of Combinatorial Optimization (2024) 47:32
https://doi.org/10.1007/s10878-024-01114-0

Online car-sharing problem with variable booking times

Haodong Liu[2] · Kelin Luo[1,3,4] · Yinfeng Xu[2] · Huili Zhang[2]

Accepted: 13 February 2024 / Published online: 30 March 2024

Abstract

In this paper, we address the problem of online car-sharing with variable booking times (CSV for short). In this scenario, customers submit ride requests, each specifying two important time parameters: the booking time and the pick-up time (start time), as well as two location parameters—the pick-up location and the drop-off location within a graph. For each request, it's important to note that it must be booked before its scheduled start time. The booking time can fall within a specific interval prior to the request's starting time. Additionally, each car is capable of serving only one request at any given time. The primary objective of the scheduler is to optimize the utilization of k cars to serve as many requests as possible. As requests arrive at their booking times, the scheduler faces an immediate decision: whether to accept or decline the request. This decision must be made promptly upon request submission, precisely at the booking time. We prove that no deterministic online algorithm can achieve a competitive ratio smaller than $L+1$ even on a special case of a path (where L denotes the ratio between the largest and the smallest request travel time). For general graphs,

The preliminary results of this work has been published in the proceeding "COCOA 2019: Combinatorial Optimization and Applications" (LNTCS, volume 11949).

✉ Kelin Luo
luo.kelin@outlook.com

Haodong Liu
HaodongLiu0515@outlook.com

Yinfeng Xu
yfxu@xjtu.edu.cn

Huili Zhang
zhang.huilims@xjtu.edu.cn

1 Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY 14260, USA

2 School of Management, Xi'an Jiaotong University, Xi'an, China

3 Institute of Computer Science, University of Bonn, Bonn, Germany

4 Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, Netherlands

we give a Greedy Algorithm that achieves $(3L+1)$-competitive ratio for CSV. We also give a Parted Greedy Algorithm with competitive ratio $(\frac{5}{2}L + 10)$ when the number of cars k is no less than $\frac{5}{4}L + 20$; for CSV on a special case of a path, the competitive ratio of Parted Greedy Algorithm is $(2L + 10)$ when $k \geq L + 20$.

Keywords Car-sharing problem · Online scheduling · Competitive analysis

1 Introduction

Recently, car-sharing has gained popularity as a convenient mode of transportation. Customers make reservations for their rides, and car-sharing companies provide vehicles for a specified duration to fulfill these requests. This trend addresses the increasing demand for urban mobility while optimizing the use of urban space. Car-sharing has emerged as a prominent transportation service, offering customers the flexibility to book rides in advance. These bookings are made within a certain time frame before the requested start time, providing customers with greater flexibility compared to fixed-time bookings. Furthermore, the adoption of shared cars helps reduce the reliance on private vehicles, offering solutions to traffic congestion and environmental concerns (The future of driving 2012). As a result, the car-sharing problem has become a crucial area of study within the field of operations research.

Our study focused on optimizing the maximum number of satisfied customer requests. These ride requests are submitted prior to their desired start times. Additionally, the time at which a request is booked varies and falls within a specified time window before the request's start time. This flexible booking approach offers customers greater convenience compared to fixed booking schedules. We refer to this scenario as CSV (Car-Sharing with Variable booking times), and in this paper, we delve into the study of CSV. Specifically, we analyze the competitive ratio of two algorithms: the Greedy Algorithm, and the Parted Greedy Algorithm. Our analysis encompasses both general graphs and a specialized path graph.

Remark This work expanded the work of Liu et al. (2019). Our main contribution are summarized as follows.

- For CSV on general networks, we gave a precise upper bound and lower bound, which improved the known results.
- Compared with the conference paper, we studied a new subproblem, which is CSV on a path. We proposed a new algorithm named Part Greedy Algorithm, which had a well performence on both CSV on general networks and CSV on a path.

1.1 Related work

The *offline* car-sharing problem was initially investigated by Böhmová et al. (2016). In a general graph, there are a number of requests, each specified by a pick-up time, a pick-up location, and a drop-off location. The scheduler must decide whether to accept or decline each request and efficiently schedule the accepted requests. Böhmová et al. demonstrated that the car-sharing problem, which aims to maximize the accepted requests, can be solved in polynomial time. They also explored a problem variant in which each customer submits two ride requests in opposite directions. The scheduler faces the choice of either serving both requests or declining both. In this variant, Böhmová et al. proved that maximizing the number of satisfied customers is both NP-hard and APX-hard.

In the *online* car-sharing problem, algorithms face the challenge of making immediate decisions to accept or reject each request without having access to future information. Once a decision is made, it cannot be reversed. Due to the inherent uncertainty of the problem, it is often impossible to achieve the best possible solution. The competitive ratio serves as a performance measure, representing the ratio between the values of the algorithm's solution and an optimal offline solution for the most challenging inputs. The online car-sharing problem has been the subject of extensive research in various scenarios. This includes different types of graphs, such as two-location scenarios (Luo et al. 2018a, b, c; Li et al. 2020), star networks (Luo et al. 2019), varying numbers of servers (ranging from one server (Luo et al. 2018a) to two servers (Luo et al. 2018b) and k servers (Luo et al. 2018c, 2019; Li et al. 2020), and variations in request attributes (such as variable booking times or fixed booking times, where the interval between booking time and start time remains constant across all rides Liu et al. 2019). In most cases, researchers have successfully matched upper and lower bounds. For instance, in the two-location problem (Luo et al. 2018c), it was established that no deterministic algorithm can achieve a competitive ratio smaller than 1.5 for the fixed booking time variant and $5/3$ for the flexible booking time variant. To address this challenge, they introduced the balanced greedy algorithm (BGA), which achieves the best possible competitive ratio. In another study (Luo et al. 2019), inspired by car-sharing applications between airports and hotels, the authors explored online scheduling problems. They considered both the unit travel time variant and the arbitrary travel time variant. In the unit travel time variant, the travel time between the airport and any hotel is a fixed value t. They devised a 2-competitive algorithm for scenarios where the length of the booking interval (the time between pick-up and booking) is at least t and the number of servers is even. In the arbitrary travel time variant, the travel time between the airport and a hotel falls within the range of t to Lt where L is greater than or equal to 1 and represents the ratio of the longest to the shortest travel time. They demonstrated that the competitive ratio of the algorithm is $O(\log L)$ when the number of servers is at least $\log L$. For both variants, they established matching lower bounds on the competitive ratio for any deterministic online algorithm. In our research, we expand this problem to encompass more general graphs.

Another extensively studied problem is online interval scheduling. The online car-sharing problem can be redefined as an interval scheduling problem when all pick-up and drop-off locations are the same. In this context, each car can be seen as a machine, and each interval corresponds to a ride request. The interval's starting time corresponds to the pick-up time of a specific request, while the ending time corresponds to the drop-off time of that request. A feasible solution involves scheduling intervals on machines in such a way that the selected intervals on a machine do not overlap. Lipton and Tomkins (1994) conducted research on this problem in a one-machine setting. They demonstrated that no randomized algorithm can achieve a competitive ratio better than $O(\log \Delta)$, where Δ represents the ratio between the longest and shortest intervals,

Table 1 Results and theorems for the online car-sharing problem with variable booking times

Problem	Lower bound	Upper bound
CSV on general graphs	$L+1$ (Th. 2)	$\min\{3L+1, \frac{5}{2}L+10\}$ (Th. 3, 6)
CSV on a path	$L+1$ (Th. 1)	$\min\{3L+1, 2L+10\}$ (Th. 4, 5)

and it is unknown to the algorithm. Additionally, they presented an $O((\log \Delta)^{\epsilon})$-competitive randomized algorithm to address this challenge.

Another problem closely related to our setting is the online ride-sharing problem (Guo and Luo 2022) and the online dial-a-ride problem (OLDARP). The online dial-a-ride problem can be viewed as a version of the online car-sharing problem without booking times. In OLDARP, the objective is to minimize the makespan or minimize the maximum flow time (Krumke et al. 2005). The key distinction lies in OLDARP, where all requests must be serviced as soon as possible, whereas in the online car-sharing problem, orders need to be serviced at specific times. For a more detailed exploration of these problems, interested readers can refer to Christman et al. (2018).

1.2 Paper outline

We formulate and analyze the online car-sharing problem with variable booking times on two graphs: CSV on a path and CSV on a general graph. We propose two algorithms, the Greedy Algorithm (GA) and the Parted Greedy Algorithm (PGA). We formulate the results of this paper in Table 1: For CSV on a general graph, no deterministic online algorithm can achieve a competitive ratio smaller than $L+1$, and we prove that GA is $(3L+1)$-competitive, PGA is $(\frac{5}{2}L+10)$-competitive when $k \geq \frac{5}{4}L+20$; For CSV on a path, no deterministic online algorithm can achieve a competitive ratio smaller than $L+1$, and we prove that GA is $(3L+1)$-competitive, PGA is $(2L+10)$-competitive when $k \geq L+20$.

The rest of the paper is organized as follows. We give the preliminaries in Sect. 2. In Sect. 3, we present the lower bounds on the competitive ratio for CSV on a path. In Sect. 4, we propose two algorithms: the Greedy Algorithm and the Parted Greedy Algorithm, and prove the competitive ratios. Section 5 concludes the results of this work.

2 Preliminaries

Notation. We consider a setting with a graph $G = (V, E)$ with edge length $\ell : E \rightarrow_{\geq 0}$. For an edge $(p, q) \in E$ (p and q are two vertices in V), $\ell(p, q)$ represents the distance (also, the travel time) between p and q; for convenience of representation, for $\{p, q\} \notin E$, let $\ell(p, q)$ denote the shortest travel time between p and q. We assume that each travel time $\ell(p, q)$ is non-negative and symmetric in our setting. There are k cars denoted by $S = \{s_1, s_2, \ldots, s_k\}$, which will ride on the graph G. Assume that all the cars initially are at one location (in fact, the initial car locations do not affect our results). Let $R = \{r_1, r_2, \ldots, r_n\}$ denote a request sequence, where request r_i is specified by the booking time b_i, the pick-up time (start time) t_i, the pick-up location $p_i \in V$ and the drop-off location $\dot{p}_i \in V$, i.e., $r_i = \{b_i, t_i, p_i, \dot{p}_i\}$. According to the definition of travel time, the drop-off time of each request r_i satisfies $\dot{t}_i = t_i + t(p_i, \dot{p}_i)$. In our setting, suppose that the shortest travel time between two locations is t, and the longest travel time is Lt, i.e., the longest request is at most L times the length of the shortest request. Customer requests arrive online. When a request is arrive, we need to decide whether to accept it or not immediately and irrevocably, without knowledge of future informations. When a request is accepted, it is not necessary to deciede which car is assigned to serve it immediately. We only need to avoid the condition that no car could serve the request at the start time of it.

With respect to constraint on the booking times, one can consider the car-sharing problem with variable booking times. In this problem, customers may book a request at any time of an interval before the start time of the request, i.e. $b_l \leq t_i - b_i \leq b_u$ with $b_l < b_u$ holds for request $r_i \in R$. In this paper, we assume $b_l \geq Lt$ to ensure that an empty car is available for serving any arriving request. Once we accept a request r_i, we must assign a car to pick up the customer at p_i on t_i, and then drop off the customer at $\dot{p}_i$ on $\dot{t}_i$. Each car can serve one request at a time. Our aim is to accept the maximum number of requests.

In our problem, we require algorithms to decide either to accept or reject a request immediately when it arrives. Usually, we do not require that the algorithm assigns an accepted request to a specific car immediately, provided that it ensures that some cars could serve this request. In this paper, however, it is not necessary for an algorithm to use this flexibility. Our algorithm assigns a request immediately when it is accepted.

Note that two requests, r_i and r_j (w.l.o.g, suppose $t_i \leq t_j$), can be served by one car only if there is enough time to reach the pick-up location of r_j after serving request r_i, i.e., $t_j \geq \dot{t}_i + \ell(\dot{p}_i, p_j)$. If two requests can not be served by one car, we say that the two requests are *in conflict*, which will be of use in Sect. 3 and Sect. 4.

Methods. For an arbitrary algorithm ALG, the property of ALG is evaluated by its competitive ratio (Borodin and El-Yaniv 1998). For any sequence of requests R, let OPT_R denote the objective value produced by an optimal scheduler OPT, where OPT has full information about R in advance, and ALG_R denote the objective value produced by an online deterministic algorithm A. The competitive ratio of algorithm A is defined by ρ_A, which is shown by $\rho_A = \sup_R \frac{OPT_R}{ALG_R}$, and A is ρ_A-competitive. For a problem, we say β is the *lower bound* on the best possible competitive ratio if $\rho_A \geq \beta$ for all $A \in ON$, where ON is the set of all online deterministic algorithms. We say an algorithm A is optimal if $\rho_A = \beta$.

In the whole paper, we use ALG to denote any online algorithm, and we use OPT to denote an optimal scheduler. For any sequence of requests R, the set of requests accepted by ALG is denoted by R', the set of requests accepted by OPT is denoted by R^*.

3 Lower bounds

In this section, we present a lower bound for CSV on a special graph of a path.

Fig. 1 Illustration of the path with $m + 1$ locations

Recall that t is the shortest travel time, and Lt is the longest travel time among all the requests. We consider a path with $m + 1$ locations $\{0, 1, 2, \cdots, m\}$. The travel time between any two locations i and j satisfies $\ell(i, j) = t \cdot |i - j|$ for $0 \leq i, j \leq m$, as shown in Fig. 1. Note that on such a path, the travel time of any request is at most $m \cdot t$ and at least t, hence we have $L = m$.

Our strategy to obtain a lower bound is to ensure that most of the ALG cars each accepts exactly one request which is in conflict with the future requests. The adversary presents requests in m phases, where phase i $(1 \leq i \leq m)$ includes l_i groups of requests (where l_i will be specified later). Let $R_{i,j}$ $(1 \leq i \leq m, 1 \leq j \leq l_i)$ denote the set of requests in phase i group j. $R_{i,j}$ consists of k identical requests, which is denoted by $r_{i,j}$.

Now we place three principles for the adversary to release requests:

(a) Any two requests in the same phase are in conflict;
(b) In any two phases i and g $(i < g)$, any request in the last group of phase i, i.e. R_{i,l_i}, and a request in phase g are not in conflict;
(c) In any two phases i and g $(i < g)$, any request in phase i, except for requests in R_{i,l_i}, is in conflict with requests in phase g.

Let $k_{i,j}$ denote the number of requests accepted by ALG in $R_{i,j}$. Notice that any two requests in $R_{i,j}$ are in conflict, thus $k_{i,j}$ is also the number of cars that ALG uses to serve the accepted requests in $R_{i,j}$.

Next, we will specify the request instance in the following Theorem 1.

Theorem 1 *No deterministic online algorithm for CSV on a special graph of a path can achieve a competitive ratio smaller than* $L + 1$*, where L is the ratio of the longest request to the shortest request.*

Proof Suppose that $\nu \in \mathbb{N}$ and $\nu \cdot t - b_u \geq Lt$. To show the lower bound for CSV, the adversary releases requests based on the rules shown in Algorithm 1.

Algorithm 1 Releasing rule

```
1: Initialization: The first group in phase 1 (R_{1,1}) is released.
2: i = 1, j = 1.
3:    While i ≤ m + 1 do
4:       If k_{i,j} = 0, then l_i = j, i = i + 1, j = 1 and the adversary releases R_{i,j};
5:       Otherwise j = j + 1, and the adversary releases R_{i,j};
6: Output: l_i for all 1 ≤ i ≤ m; k_{i,j} for all 1 ≤ i ≤ m + 1 and 1 ≤ j ≤ l_i.
```

In Algorithm 1, when a group of requests $R_{i,j}$ is presented, If ALG accepts no request in $R_{i,j}$, i.e., $k_{i,j} = 0$, the adversary ends the current phase and releases reqeusts in $R_{i+1,1}$; if ALG accepts any request in $R_{i,j}$, i.e., $k_{i,j} > 0$, then the adversary releases requests in $R_{i,j+1}$, the subsequent group of $R_{i,j}$ (see line 5).

Now we specify the requests in $R_{i,j}$. For each request $r_{i,j}$, we set $\dot{p}_{i,j} = p_{i,j} + 1$. Thus when we specify $r_{i,j}$, we can ignore the drop-off location. Request $r_{i,j}$ is specified as following:

- $R_{1,1}$ consists of k copies of the request $r_{1,1}$, where $b_{1,1} = \nu \cdot t - b_u$, $t_{1,1} = \nu \cdot t$, $p_{1,1} = 0$;
- $R_{i,1}$ ($i > 1$) consists of k copies of the request $r_{i,1}$ with booking time $b_{i,1} = t_{i,1} - b_u$, start time $t_{i,1} = t_{i-1,l_{i-1}} + t + \frac{k}{k+1} \cdot \frac{\min\{b_u - b_l, t\}}{(k+1)^i}$ and pick-up location $p_{i,1} = i - 1$;
- $R_{i,j}$ ($i \geq 1$, $j > 1$) consists of k copies of the request $r_{i,j}$ with booking time $b_{i,j} = b_{i,1}$, start time $t_{i,j} = t_{i,1} - (j-1) \cdot \frac{\min\{b_u - b_l, t\}}{(k+1)^{i+1}}$, and pick-up location $p_{i,j} = i - 1$.

Firstly, we state that the requests in phase i group j are reasonable: each request $r_{i,j}$ is released no later than $b_u + b_{i,j}$ and no earlier than $b_l + b_{i,j}$. Observe that there are m phases and each phase consists of no more than $k+1$ groups, in phase i group j ($j \leq l_i \leq k+1$), since $b_{i,j} = b_{i,1}$ and $t_{i,j} = t_{i,1} - (j-1) \cdot \frac{\min\{b_u - b_l, t\}}{(k+1)^{i+1}}$, we have $t_{i,j} - b_{i,j} = b_u - (j-1) \cdot \frac{\min\{b_u - b_l, t\}}{(k+1)^{i+1}}$. Thus we know that $b_l \leq t_{i,j} - b_{i,j} \leq b_u$ since $1 \leq j \leq k+1$.

Note that the requests are presented in order of phases, and the requests in the same phase are presented in order of groups: In phase i group j and phase i group h ($1 \leq h < j$), since $b_{i,j} = b_{i,h}$, we know that the requests in one phase can be presented in order of groups. For any two consecutive phases $i-1$ and i ($1 < i \leq m$), since $t_{i,1} = t_{i-1,l_{i-1}} + t + \frac{k}{k+1} \cdot \frac{\min\{b_u - b_l, t\}}{(k+1)^i}$, we have $b_{i,1} = t_{i,1} - b_u > t_{i-1,l_{i-1}} + t - b_u$. Observe that $t_{i-1,l_{i-1}} + t \geq t_{i-1,1}$, we have $b_{i,1} > t_{i-1,1} - b_u = b_{i-1,1}$. It means that the requests are presented in order of phases.

Next, we state that the requests satisfy the three principles.

For any two groups $j, h \in \{1, 2, \ldots, l_i\}$ in phase i with $j \geq h$, we have $t_{i,j} = t_{i,1} - (j-1) \cdot \frac{\min\{b_u - b_l, t\}}{(k+1)^{i+1}}$. And we know that $0 < t_{i,h} - t_{i,j} = (j-h) \cdot \frac{\min\{b_u - b_l, t\}}{(k+1)^i} < t$. Notice that $\dot{p}_{i,j} = \dot{p}_{i,h} = i$, $p_{i,j} = p_{i,h} = i-1$, we have $\ell(\dot{p}_{i,h}, p_{i,j}) = t$. Hence $t_{i,h} < t_{i,j} < t_{i,h} + t < \dot{t}_{i,h} + \ell(\dot{p}_{i,h}, p_{i,j})$. It means that two requests $r_{i,h}$ and $r_{i,j}$ are in conflict, which satisfies principle (a).

For phase $i+1$ group h ($1 \leq h \leq l_{i+1} \leq k+1$), we have $t_{i+1,h} = t_{i,l_i} + t + \frac{k+1-h}{k+1} \cdot \frac{\min\{b_u - b_l, t\}}{(k+1)^{i+1}}$. Observe that $\dot{p}_{i,l_i} = p_{i+1,h}$, we have $\ell(\dot{p}_{i,l_i}, p_{i+1,h}) = 0$, and hence $t_{i+1,h} > \dot{t}_{i,l_i} + \ell(\dot{p}_{i,l_i}, p_{i+1,h})$. It means that any request in the last group of phase i and a request in phase $i+1$ group h ($h > 1$) are not in conflict, which satisfies principle (b). For phase i group j ($1 \leq j < l_i$), $t_{i,l_i} + t \geq t_{i,j} \geq t_{i,l_i} + \frac{1}{k+1} \cdot \frac{\min\{b_u - b_l, t\}}{(k+1)^i}$, we have $t_{i,j} \leq t_{i+1,h} \leq t_{i,j} + t - \frac{h}{k+1} \cdot \frac{\min\{b_u - b_l, t\}}{(k+1)^{i+1}}$ since $t_{i+1,h} = t_{i,l_i} + t + \frac{k+1-h}{k+1} \cdot \frac{\min\{b_u - b_l, t\}}{(k+1)^{i+1}}$. Observe that $\dot{p}_{i,j} = p_{i+1,h}$, we have $\ell(\dot{p}_{i,j}, p_{i+1,h}) = 0$, and hence $t_{i+1,h} < \dot{t}_{i,j} + \ell(\dot{p}_{i,j}, p_{i+1,h})$ since $h < k+1$. It means that any other requests (except the requests in the last group of phase i) in phase i and a request in phase $i+1$ group h are in conflict, which satisfies principle (c) (Fig. 2 shows an example).

In the end, we analyze the number of requests accepted by ALG and OPT. Observe that ALG accepts no request in $R_{i,j}$ ($\forall 1 \leq i \leq m+1$, $j = l_i$) (see line 4). Since there are k cars in our problem, and any two requests accepted by ALG are in conflict

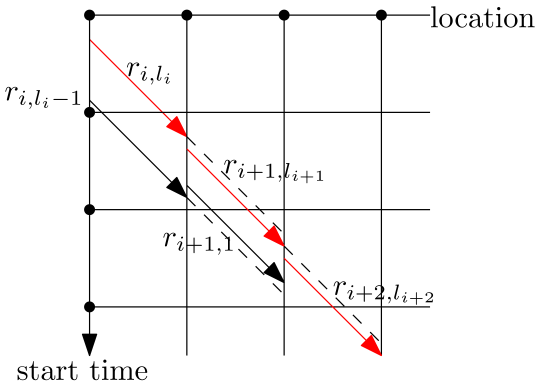

Fig. 2 Illustration of requests r_{i,l_i-1}, r_{i,l_i} and requests in phase $i+1$

based on principles (a) and (c), we have $|R'| \leq k$. Meanwhile, based on the principle (b), any two requests in the last group of two different phases are not in conflict, i.e., a request in R_{i,l_i} and a request in R_{h,l_h} are not in conflict. Thus, OPT accepts all requests in R_{i,l_i} for all $1 \leq i \leq m+1$, i.e., $|R^*| = k \cdot (m+1)$, and hence we get $|R^*|/|R'| \geq L+1$ $(L = m)$. □

By Theorem 1, we have the following Theorem:

Theorem 2 *No deterministic online algorithm for CSV on general graphs can achieve a competitive ratio smaller than* $L+1$.

4 Upper bounds

We formulated two algorithms: the Greedy Algorithm (GA) and the Parted Greedy Algorithm (PGA) for CSV on both the special graph of a path and general graphs. We prove that for CSV on a path, GA is $(3L+1)$-competitive and PGA is $(2L+10)$-competitive; for CSV on general graphs, GA is $(3L+1)$-competitive and PGA is $(\frac{5}{2}L+10)$-competitive.

For the sake of analysis, let S' and S^* denote the sets of cars in the algorithm and OPT. Consider a sequence $R = \{r_1, \ldots, r_n\}$. Let $R' \subseteq R$ denote the set of requests accepted by our algorithm, and $R^* \subseteq R$ denote the set of requests accepted by OPT. Let $\bar{R}$ be the set of requests accepted by OPT that are not accepted by the algorithm. For each car $s_e^* \in S^*$, let $\bar{R}_e$ be the requests in $\bar{R}$ and accepted by the car s_e^*. Observe that $\sum_{s_e * \in S^*} |\bar{R}_e| = |\bar{R}|$.

We claim that for each $\bar{R}_e \in \bar{R}$ and for any car $s_j' \in S'$, we have $|\bar{R}_e| \leq \alpha \cdot |R_j' \backslash R_e^*|$, where R_j' is the set of requests accepted by car $s_j' \in S'$, and R_e^* is the set of requests accepted by car $s_e^* \in S^*$. If this claim holds, since $R^* \backslash \bar{R} = R^* \cap R'$, we get that

$$|R^*| = \sum_{s_e * \in S^*} |\bar{R}_e| + |R^* \cap \bar{R}|$$

$$
\begin{aligned}
&\leq \sum_{j=1}^{k} \alpha |R'_j \setminus R^*_e| + |R^* \cap R'| \\
&= \alpha |R' \setminus R^*| + |R^* \cap R'| \\
&\leq \alpha |R'|
\end{aligned}
$$

with $\alpha \geq 1$.

To prove the above claim, we will adapt the "charging scheme", which is similar to lemma 9 in Luo et al. (2019). The difference here is that the requests in this paper have variable booking times, instead of fixed booking times, and thus we need to consider the number of requests which do not intersect with the "charging interval" at any time point, not just the start times. For completeness, we define the "charging interval" as follows.

For any request $r_i = (b_i, t_i, p_i, \dot{p}_i)$, we can find a time interval (α_i, β_i) of r_i, such that another request r_j is in conflict with r_i only if $(t_j, \dot{t}_j) \cap (\alpha_i, \beta_i) \neq \emptyset$, for any request $r_j = (b_j, t_j, p_j, \dot{p}_j)$. Notice that $(t_j, \dot{t}_j) \cap (\alpha_i, \beta_i) = \emptyset$ means that either $\dot{t}_i < \alpha_i$ or $t_i > \beta_i$ holds. We call the time interval (α_i, β_i) an *occupy interval* of r_i. For a request r_i, the occupy interval (α_i, β_i) is non-unique, we concern on the occupy interval (α_i, β_i) with a proper length.

Firstly, we consider the occupy interval of any request $r_i = (b_i, t_i, p_i, \dot{p}_i)$ on general graphs. Recall that the longest travel time is Lt. For any request r_j, observe that if $t_j > \dot{t}_i + Lt$, the Lt time units is sufficient for a car to make an empty movement from $\dot{p}_i$ to p_j, which means the two requests r_i and r_j are not in conflict since a car may serve both requests. Similarly, if $\dot{t}_j < t_i - Lt$, r_i and r_j are not in conflict. Thus we have the following observation.

Observation 1 *For CSV on a general graph, for any request $r_i = (b_i, t_i, p_i, \dot{p}_i)$, $(t_i - Lt, \dot{t}_i + Lt)$ is an occupy interval of r_i, and the length of the occupy interval is $2Lt + \dot{t}_i - t_i$.*

For CSV on a special graph of a path, we have the following lemma.

Lemma 1 *For any request $r_i = (b_i, t_i, p_i, \dot{p}_i)$ in CSV on a path, we can find an occupy interval (α_i, β_i), where the length of the occupy interval is as following*

$$\beta_i - \alpha_i = \max\{Lt + 2(\dot{t}_i - t_i), 2Lt\}.$$

Proof Consider a special graph of path $G = (V, E)$ where $V = \{v_1, v_2, \ldots, v_n\}$ and $E = \{e_1, e_2, \ldots, e_{n-1}\}$ with $e_i = \{v_i, v_{i+1}\}$ for all $i \in [n-1]$. We assume that $\min_{v_i, v_j \in V} \ell(v_i, v_j) = t$ and $\max_{v_i, v_j \in V} \ell(v_i, v_j) = Lt$.

Now for any request $r_i = (b_i, t_i, p_i, \dot{p}_i)$, suppose $\ell(v_1, p_i) = pt$ and $\ell(v_1, \dot{p}_i) = qt$ with $p < q$ (See Fig. 3 as an example). Then we can construct an interval (α_i, β_i) as following:

- $\alpha_i = t_i - \max\{pt, Lt - pt\}$,
- $\beta_i = \dot{t}_i + \max\{qt, Lt - qt\}$.

It is not difficult to show that (α_i, β_i) is an *occupy interval* of r_i. We assume that there is a request r_j which is in conflict with r_i and $(t_j, \dot{t}_j) \cap (\alpha_i, \beta_i) = \emptyset$. Since request r_j

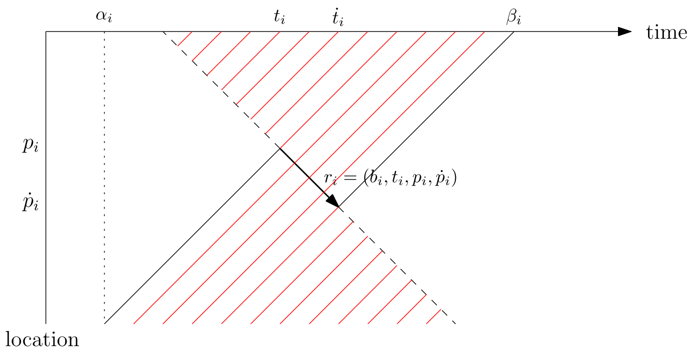

Fig. 3 Illustration of the request $r_i = (b_i, t_i, p_i, \dot{p}_i)$ and its occupy interval (α_i, β_i)

and r_i are in conflict, we have either $t_j < t_i < \dot{t}_j + \ell(p_i, \dot{p}_j)$ or $t_i < t_j < \dot{t}_i + \ell(p_j, \dot{p}_i)$. Furthermore, either $\dot{t}_j > t_i - \ell(p_i, \dot{p}_j) \geq \alpha_i$ or $t_j < \dot{t}_i + \ell(p_j, \dot{p}_i) \leq \beta_i$, and thus $(t_j, \dot{t}_j) \cap (\alpha_i, \beta_i) \neq \emptyset$, which derives a contradiction.

Consider the values of $\max\{pt, Lt - pt\}$ and $\max\{qt, Lt - qt\}$, we separate three cases to show the length of (α_i, β_i):

1. When $pt \geq Lt - pt$ and $qt \geq Lt - qt$, $\beta_i - \alpha_i = (p + q + q - p)t = 2qt$;
2. When $pt \leq Lt - pt$ and $qt \leq Lt - qt$, $\beta_i - \alpha_i = (L - p + L - q + q - p)t = 2(L - p_i)t$;
3. When $pt \leq Lt - pt$ and $qt \geq Lt - qt$, $\beta_i - \alpha_i = (L - p + q + q - p)t = (L + 2(q - p))t$.

We can see that for the first two cases (case 1, case 2), the length of the occupy interval is no greater than $2Lt$ since $pt \leq Lt$ and $qt \leq Lt$ hold, and for the case 3, the length of the occupy interval is no greater than $Lt + 2(\dot{t}_i - t_i)$. □

Before introducing the algorithms, we present the following lemma which will be used to bound the number of requests in the charging scheme analysis.

Lemma 2 *For a given bipartite graph $G = (U, V, E)$, if each vertex $v \in V$ is adjacent to at most m_1 vertices of U, and each vertex $u \in U$ is adjacent to at least m_2 vertices of V, then we have $|V| \geq |U|\frac{m_2}{m_1}$.*

The proof is quite simple: Since each vertex $u \in U$ is adjacent to at least m_2 vertices of V, there are at least $|U| \cdot m_2$ edges in the graph G, i.e., $|E| \geq |U| \cdot m_2$. Meanwhile, since each vertex $v \in V$ is adjacent to at most m_1 vertices in U, then $|V|$ is no less than $|E|/m_1$. Therefore, we have $|V| \geq |U|\frac{m_2}{m_1}$.

4.1 Greedy algorithm

In this section, we formulate a Greedy Algorithm (GA) for CSV on general graphs, and prove that GA is $(3L + 1)$-competitive. GA can be stated in a simple way: When a request r_i arrives, if r_i is acceptable to a car s_j from S, we accept r_i and assign it to s_j; Otherwise, reject it.

Request r_i is acceptable to car s_j: Let R_j denote the set of requests assigned to car s_j from $\{r_1, r_2, \dots r_{i-1}\}$. For any request $r_h \in R_j$, if we have $t_i \geq \dot{t}_h + \ell(\dot{p}_h, p_i)$ or $t_h \geq \dot{t}_i + \ell(\dot{p}_i, p_h)$, in other words, r_i and the requests in R_j are not in conflict, we say that request r_i is acceptable to car s_j.

Suppose the adversary releases requests $R = \{r_1, r_2, \dots, r_n\}$ for $1 \leq i < n$. Let R' denote the accepted requests by GA. Let R^* denote the accepted requests by the offline scheduler (optimal solution).

Algorithm 2 Greedy Algorithm (GA)

1: *Input*: k cars, requests arrive over time.
2: When request r_i arrives, if r_i is acceptable to a car $s \in S$, assign it to that car; Otherwise, reject it.

Theorem 3 *For CSV on general graphs, GA is* $(3L+1)$*-competitive.*

Proof Recall that $\bar{R}$ is the set of requests accepted by OPT which are not accepted by GA. For each car $s_e^* \in S^*$ of an OPT solution, $\bar{R}_e$ is the requests in $\bar{R}$ and accepted by s_e^*. We claim that for each $\bar{R}_e \in \bar{R}$ and for any car $s'_j \in S'$, we have $|\bar{R}_e| \leq \alpha \cdot |R'_j - R_e^*|$. If this claim holds, then we have $|R^*| \leq \alpha|R' - R^*| + |R^* \cap R'| \leq \alpha|R'|$ with $\alpha \geq 1$, since $R^* - \bar{R} = (R^* \cap R')$.

Consider any request $r_i \in \bar{R}_e$. since s'_j did not accept r_i, s'_j must have accept another request r_c which is in conflict with r_i. We charge r_i to r_c once we find a request r_c which is in conflict with r_i. In this way, every request in $\bar{R}_e$ is charged to a request in R'_j.

Next, we bound the number of requests in $\bar{R}_e$ that can be charged to a single request $r_c \in R'_j$. Observe that if interval $(t_h, \dot{t}_h)$ does not intersect with the occupy interval (α_c, β_c), it is sufficient for s'_j to serve both r_c and r_h according to the definition of the occupy interval. As all requests have travel time at least t, the start times of any two consecutive requests accepted by s_e^* differ by at least t. Here "consecutive requests" means the requests are consecutive according to the time order. Recall that there is an occupy interval (α_c, β_c) with length $3Lt$ by Observation 1, which can intersect with at most $3L+1$ consecutive requests. It means that r_c is charged by at most $3L+1$ requests from $\bar{R}_e$.

This establishes the claim, with $\alpha = 3L+1$. Thus we get $|R^*| \leq (3L+1) \cdot |R'|$. The theorem is proved. □

Notice that a path is a special graph, by Theorem 3, we have the following Theorem:

Theorem 4 *For CSV on a path, GA is* $(3L+1)$*-competitive.*

4.2 Parted greedy algorithm

According to Lemma 1, for any request r_i, when $\dot{t}_i - t_i$ is no less than $\frac{Lt}{2}$, we can find an occupy interval of r_i, the length of which is $2Lt$; when $\dot{t}_i - t_i$ is greater than $\frac{Lt}{2}$, we can find an occupy interval of r_i, the length of which is $3Lt$. We will take advantage of the length difference of the occupy intervals for different requests in the Parted Greedy Algorithm. In PGA, the cars are separated into two parts, one part is denoted by $S'(1)$, in which the cars only serve requests of length no larger than $\frac{L}{2}$, and the other part of cars, denoted by $S'(2)$, only serve requests of length larger than $\frac{L}{2}$. The accurate numbers of cars in $S'(1)$ and $S'(2)$ are shown in the following Theorems.

The definition of **acceptable** is similar to Greedy Algorithm. We say that request r_i is acceptable to car s_j if r_i and any request in R_j are not in conflict. Besides, for a request r_i, if the length of r_i is no larger than $\frac{L}{2}$, r_i is only acceptable to cars in $S'(1)$; Otherwise, r_i is only acceptable to cars in $S'(2)$.

Suppose the adversary releases requests $R = \{r_1, r_2, \ldots, r_n\}$. According to the length of each request in R, R can be separated into two sets $R(1)$ and $R(2)$, where $R(1)$ (resp. $R(2)$) denotes the set of requests in R with length no larger than $\frac{L}{2}$ (resp. larger than $\frac{L}{2}$).

Let $R'(1)$ and $R'(2)$ denote the requests accepted by PGA in $R(1)$ and $R(2)$. Notice that $|R'(1)| = \sum_{s_j \in S'(1)} |R'_j|$ (resp. $|R'(2)| = \sum_{s_j \in S'(2)} |R'_j|$). Let $R^*(1)$ and $R^*(2)$ denote the requests accepted by the offline scheduler (optimal solution) in $R(1)$ and $R(2)$.

Algorithm 3 Parted Greedy Algorithm (PGA)

1: *Input*: k cars, requests arrive over time.
2: When request r_i with $r_i \in R(\tau)$ ($\tau \in \{1, 2\}$) arrives, if it is acceptable to a car $s \in S'(\tau)$, assign it to that car; otherwise, reject it.

Recall that $\bar{R}$ denotes the set of requests accepted by OPT that are not accepted by PGA. For each car $s_e^* \in S^*$, let $\bar{R}_e$ denote the requests in $\bar{R}$ and accepted by the car s_e^*. For each car $s_e^* \in S^*$, we further denote the requests in $\bar{R}_e$ with length no larger than $\frac{L}{2}$ (resp. larger than $\frac{L}{2}$) by $\bar{R}_e(1)$ (resp. $\bar{R}_e(2)$). i.e., $\bar{R}_e(1) = \bar{R}_e \cap R(1)$ and $\bar{R}_e(2) = \bar{R}_e \cap R(2)$. Observe that $|\bar{R}_e(1)| + |\bar{R}_e(2)| = |\bar{R}_e|$, and $\sum_{e=1}^{k} |\bar{R}_e| = |\bar{R}|$.

Theorem 5 *For CSV on a path with $k \geq L + 20$, PGA is $(2L + 10)$-competitive when we set $|S(1)| = \lfloor \frac{(2L+1)k}{2L+8} \rfloor$.*

Proof Firstly, we focus on the requests in $R(1)$ that are either accepted by $s_e^* \in S^*$ or accepted by $s'_j \in S'(1)$. Consider any request $r_h \in \bar{R}_e(1)$, since PGA does not accept r_h, the car s'_j must have accepted another request r_c, such that r_c and r_h are in conflict. As for cars in $S'(2)$, since the length of r_h is no larger than $\frac{L}{2}$, PGA will not assign r_h to any request in $R'(2)$. We say that r_h charges to r_c once we find a $r_c \in R'(1)$ which is in conflict with $r_h \in R^*$.

We bound the number of requests in $\bar{R}_e(1)$ that can be charged to a single request $r_c \in R'(1)$ for any car $s_e^* \in S^*$. Observe that by Lemma 1, for a request r_h, if $(t_h, \dot{t}_h)$ does not intersect with an occupy interval of r_c, i.e., (α_c, β_c), it is sufficient for s'_j to serve both r_c and r_h. Notice that the length of request r_c is no larger than $\frac{L}{2}$, according to Lemma 1, there exists an occupy interval (α_c, β_c) such that $\beta_c - \alpha_c = 2Lt$. Since all requests in $\bar{R} \cap R(1)$ have travel time at least t, the start times of any two consecutive requests accepted by s_e^* differ by at least t, which means that (α_c, β_c) may intersect with at most $2L+1$ consecutive requests. Thus r_c is charged by at most $2L+1$ requests from $\bar{R}_e(1)$.

Then we consider the requests in $R(2)$ that are either accepted by s_e^* in S^* or accepted by $s'_j \in S'(2)$. Consider any request $r_h \in \bar{R}_e(2)$, since PGA does not accept r_h, the car s'_j must have accepted another request r_c, such that r_c and r_h are in conflict. As for cars in $S'(1)$, since the length of r_h is larger than $\frac{L}{2}$, PGA will not assign r_h to any request in $R'(1)$. Similarly, we charge r_h to r_c.

Observe that for a request r_h, if $(t_h, \dot{t}_h)$ does not intersect with the occupy interval (α_c, β_c), it is sufficient for $s'_j \in S'(2)$ to serve both r_c and r_h. As all requests in $\bar{R} \cap R(2)$ have travel time at least $\frac{Lt}{2}$, the start times of any two consecutive requests accepted by s_e^* differ by at least $\frac{Lt}{2}$. By Lemma 1, we can find an occupy interval (α_c, β_c) of r_c, where $\beta_c - \alpha_c = 3Lt$. Thus (α_c, β_c) may intersect with at most $\frac{3Lt}{Lt/2} + 1$ consecutive requests. It means that r_c is charged by at most 7 requests from $\bar{R}_e(2)$.

For a request $r_h \in \bar{R}_e(1)$ (resp. $r_h \in \bar{R}_e(2)$), we know that for each car $s'_j \in S'(1)$ (resp. $s'_j \in S'(2)$), s'_j can not serve r_h. Thus we charge r_h to at least $|S'(1)|$ requests in $R'(1)$ (resp. $|S'(2)|$ requests in $R'(2)$).

Set $|S(1)| = \lfloor \frac{(2L+1)k}{2L+8} \rfloor$, then $|S(2)| = k - |S_1| \geq \frac{7k}{2L+8}$. We can construct a bipartite graph between requests in $\bigcup_{s_e^* \in S^*} \bar{R}_e(1)$ and requests in $\bigcup_{s'_j \in S'(1)} (R'_j - R^*)$, and a bipartite graph between requests in $\bigcup_{s_e^* \in S^*} \bar{R}_e(2)$ and requests in $\bigcup_{s'_j \in S'(2)} (R'_j - R^*)$, separately. There is an edge (r^*, r') if r^* charges r'. Based on Lemma 2, we know $\sum_{s'_j \in S'(1)} |R'_j - R^*| \geq \sum_{s_e^* \in S^*} |\bar{R}_e(1)| \cdot \frac{|S'(1)|}{(2L+1)k}$ since each request in $\bigcup_{s'_j \in S'(1)} (R'_j - R^*)$ is charged to at most $2Lt+1$ requests of and each request in $\bigcup_{s_e^* \in S^*} \bar{R}_e(1)$ charges to at least $|S'(1)|$ requests of $\bigcup_{s'_j \in S'(1)} (R'_j - R^*)$. Similarly, we have $\sum_{s'_j \in S'(2)} |R'_j - R^*| \geq \sum_{s_e^* \in S^*} |\bar{R}_e(2)| \cdot \frac{|S'(2)|}{7k}$.

By the analysis above, we have

$$
\begin{aligned}
|R'(1)| &= \sum_{s'_j \in S'(1)} |R'_j - R^*| + |R'(1) \cap R^*| \\
&\geq \sum_{s_e^* \in S^*} |\bar{R}_e(1)| \cdot \frac{|S'(1)|}{(2L+1)k} + |R'(1) \cap R^*| \\
&\geq \sum_{s_e^* \in S^*} |\bar{R}_e(1)| \cdot (\frac{1}{2L+8} - \frac{1}{(2L+1)k}) + |R'(1) \cap R^*| \text{ (since } k > 1),
\end{aligned}
$$

where the first inequality follows from Lemma 2. Similarly, we also have

$$
|R'(2)| = \sum_{s'_j \in S'(2)} |R'_j - R^*| + |R'(2) \cap R^*|
$$

$$\geq \sum_{s_e^* \in S^*} |\bar{R}_e(2)| \cdot \frac{|S'(2)|}{7k} + |R'(2) \cap R^*|$$
$$\geq \sum_{s_e^* \in S^*} |\bar{R}_e(2)| \cdot \frac{1}{2L+8} + |R'(2) \cap R^*|.$$

Since $k \geq L + 20$, we have $(\frac{1}{2L+8} - \frac{1}{(2L+1)k}) \geq \frac{1}{2L+10}$, and then we get

$$\begin{aligned}
|R'| =& |R'(1)| + |R'(2)| \\
\geq& \sum_{s_e^* \in S^*} (|\bar{R}_e(1)| + |\bar{R}_e(2)|) \cdot \frac{1}{2L+10} + |R'(1) \cap R^*| + |R'(2) \cap R^*| \\
>& \frac{1}{2L+10} \cdot (|R'(1) \cap R^*| + \sum_{s_e^* \in S^*} |\bar{R}_e(1)| + |R'(2) \cap R^*| + \sum_{s_e^* \in S^*} |\bar{R}_e(2)|) \\
=& \frac{1}{2L+10} \cdot |R^*|.
\end{aligned}$$

The theorem is proved. □

Theorem 6 *For CSV on general graphs with $k \geq \frac{5}{4}L + 20$, PGA is $(\frac{5}{2}L + 10)$-competitive when we set $|S(1)| = \lfloor \frac{(5L+2)k}{5L+16} \rfloor$.*

Proof Similarly to the analysis in Theorem 5, we focus on the requests both in $R(1)$ and $R(2)$ that are either accepted by s_e^* in S^* or accepted by PGA. Recall that according to Observation 1, for any request r_i on a general graph, $(t_i - Lt, \dot{t}_i + Lt)$ is an occupy interval of r_i, and the length of the occupy interval is $(2Lt + \dot{t}_i - t_i)$.

Consider any request r_h with $r_h \in \bar{R}_e(1)$ (resp. $r_h \in \bar{R}_e(2)$), since PGA does not accept r_h, car $s'_j \in S'(1)$ (resp. $s'_j \in S'(2)$) must have accepted another request r_c, such that r_c and r_h are in conflict. Then we bound the number of requests in $\bar{R}_e(1)$ that can be charged to a single request $r_c \in R'(1)$ for any car $s_e^* \in S^*$. Observe that for a request r_h, if $(t_h, \dot{t}_h)$ does not intersect with the occupy interval of r_c, i.e., (α_c, β_c), it is sufficient for s'_j to serve both r_c and r_h. Notice that the length of request r_c is no larger than $\frac{L}{2}$, then there exists an occupy interval (α_c, β_c) with length $\frac{5Lt}{2}$. We know that all requests in $\bar{R} \cap R(1)$ have travel time at least t, (α_c, β_c) may intersect with at most $\frac{5L}{2} + 1$ consecutive requests. Thus r_c is charged by at most $\frac{5L}{2} + 1$ requests from $\bar{R}_e(1)$.

On the other hand, for any request $r_h \in \bar{R}_e(2)$, the length of r_h is larger than $\frac{L}{2}$. Since PGA does not accept r_h, car $s'_j \in S'(2)$ must have accepted another request r_c in $R(2)$ that r_c charges r_h.

Observe that if $(t_h, \dot{t}_h)$ does not intersect with the occupy interval (α_c, β_c), it is sufficient for $s'_j \in S'(2)$ to serve both r_c and r_h. Since all requests in $\bar{R} \cap R(2)$ have travel time at least $\frac{Lt}{2}$, the start times of any two consecutive requests accepted by s_e^* differ by at least $\frac{Lt}{2}$. Notice that we can find an occupy interval (α_c, β_c) of r_c, where

$\beta_c - \alpha_c = 3Lt$. Thus (α_c, β_c) may intersect with at most $\frac{3Lt}{Lt/2} + 1$ consecutive requests. It means that r_c is charged by at most 7 requests from $\bar{R}_e(2)$.

For a request $r_h \in \bar{R}_e(1)$ (resp. $r_h \in \bar{R}_e(2)$), we can find that for each car $s'_j \in S'(1)$ (resp. $s'_j \in S'(2)$), s'_j can not serve r_h. Thus r_h must be charged to at least $|S'(1)|$ requests in R'_1 (resp. $|S'(2)|$ requests in $R'(2)$).

Set $|S(1)| = \lfloor \frac{(5L+2)k}{5L+16} \rfloor$, then $|S(2)| = k - |S_1| \geq \frac{14k}{5L+16}$. Similar to the analysis in Theorem 5, we have

$$
\begin{aligned}
|R'(1)| &= \sum_{s'_j \in S'(1)} |R_j - R^*| + |R'(1) \cap R^*| \\
&\geq \sum_{s^*_e \in S^*} |\bar{R}_e(1)| \cdot \frac{2|S'(1)|}{(5L+2)k} + |R'(1) \cap R^*| \\
&\geq \sum_{s^*_e \in S^*} |\bar{R}_e(1)| \cdot (\frac{2}{5L+16} - \frac{2}{(5L+2)k}) + |R'(1) \cap R^*| \text{ (since } k > 1),
\end{aligned}
$$

and

$$
\begin{aligned}
|R'(2)| &= \sum_{s'_j \in S'(2)} |R_j - R^*| + |R'(2) \cap R^*| \\
&\geq \sum_{s^*_e \in S^*} |\bar{R}_e(2)| \cdot \frac{|S'(2)|}{7k} + |R'(2) \cap R^*| \\
&\geq \sum_{s^*_e \in S^*} |\bar{R}_e(2)| \cdot \frac{2}{5L+16} + |R'(2) \cap R^*|.
\end{aligned}
$$

Since $k \geq \frac{5}{4}L + 20$, we have $(\frac{2}{5L+8} - \frac{2}{(2L+1)k}) \geq \frac{2}{5L+20}$, and then we get

$$
\begin{aligned}
|R'| =& |R'(1)| + |R'(2)| \\
\geq& \sum_{s^*_e \in S^*} (|\bar{R}_e(1)| + |\bar{R}_e(2)|) \cdot \frac{2}{5L+20} + |R'(1) \cap R^*| + |R'(2) \cap R^*| \\
>& \frac{2}{5L+20} \cdot (|R'(1) \cap R^*| + \sum_{s^*_e \in S^*} |\bar{R}_e(1)| + |R'(2) \cap R^*| + \sum_{s^*_e \in S^*} |\bar{R}_e(2)|) \\
=& \frac{2}{5L+20} \cdot |R^*|,
\end{aligned}
$$

which proves the theorem. □

5 Conclusion

We have analyzed online car-sharing problem with variable booking times on both general graphs and a special graph of a path. For CSV on general graphs, we have proved that no deterministic algorithm can achieve a competitive ratio smaller than $L + 1$. For CSV on a path, we have also proved that no deterministic algorithm can achieve a competitive ratio smaller than $L + 1$. We came up with two algorithms: the Greedy Algorithm (GA) and the Parted Greedy Algorithm (PGA). According to the analysis of two algorithms, we proved that GA is $3L + 1$-competitive for CSV on general graphs, and PGA is $(\frac{5}{2}L + 10)$-competitive for CSV on a general graph. For CSV on a path, the competitive ratio of GA and PGA are proved to be $3\,L + 1$ and $2\,L + 10$.

There are still some new interesting questions, such as the online car-sharing problem under the stochastic viewpoint, or CSV with different booking time constraints.

Acknowledgements This research is supported by the National Natural Science Foundation of China (Grant Nos. 71832001, 72071157 and 72192834).

Author Contributions All authors contributed to the study conception and design. The first draft of the manuscript was written by HL and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Data Availibility Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

References

Böhmová K, Disser Y, Mihalák M, Šrámek R (2016) Scheduling transfers of resources over time: towards car-sharing with flexible drop-offs. In: Kranakis E, Navarro G, Chávez E (eds) LATIN 2016: theoretical informatics—12th Latin american symposium, Ensenada, Mexico, April 11–15, 2016, proceedings, lecture notes in computer science, vol 9644. Springer, pp 220–234. https://doi.org/10.1007/978-3-662-49529-2_17

Borodin A, El-Yaniv R (1998) Online computation and competitive analysis. Cambridge University Press, Cambridge

Christman A, Forcier W, Poudel A (2018) From theory to practice: maximizing revenues for on-line dial-a-ride. J Comb Optim 35(2):512–529. https://doi.org/10.1007/s10878-017-0188-z

Guo X, Luo K (2022) Algorithms for online car-sharing problem. In: Balachandran N, Inkulu R (eds) Algorithms and discrete applied mathematics-8th international conference, CALDAM 2022, Puducherry, India, February 10–12, 2022, Proceedings, Lecture Notes in Computer Science, vol. 13179. Springer, pp 224–236. https://doi.org/10.1007/978-3-030-95018-7_18

Krumke SO, de Paepe W, Poensgen D, Lipmann M, Marchetti-Spaccamela A, Stougie L (2005) On minimizing the maximum flow time in the online dial-a-ride problem. In: Erlebach T, Persiano G (eds) Approximation and online algorithms, third international workshop, WAOA 2005, Palma de Mallorca, Spain, October 6–7, 2005, Revised papers, lecture notes in computer science, vol 3879. Springer, pp 258–269. https://doi.org/10.1007/11671411_20

Li S, Zheng L, Lee VCS (2020) Car-sharing: online scheduling k cars between two locations. In: Li M (ed) Frontiers in algorithmics-14th international workshop, FAW 2020, Haikou, China, October 19–21, 2020, proceedings, lecture notes in computer science, vol 12340. Springer, pp 49–61. https://doi.org/10.1007/978-3-030-59901-0_5

Lipton RJ, Tomkins A (1994) Online interval scheduling. In: Sleator DD (ed) Proceedings of the fifth annual ACM-SIAM symposium on discrete algorithms. 23–25 January 1994, Arlington, Virginia, USA. ACM/SIAM, pp 302–311. http://dl.acm.org/citation.cfm?id=314464.314506

Liu H, Luo K, Xu Y, Zhang H (2019) Car-sharing problem: online scheduling with flexible advance bookings. In: Li Y, Cardei M, Huang Y (eds) Combinatorial optimization and applications-13th international conference, COCOA 2019, Xiamen, China, December 13–15, 2019, proceedings, lecture notes in computer science, vol 11949. Springer, pp 340–351. https://doi.org/10.1007/978-3-030-36412-0_27

Luo K, Erlebach T, Xu Y (2018a) Car-sharing between two locations: online scheduling with flexible advance bookings. In: Wang L, Zhu D (eds) Computing and combinatorics-24th international conference, COCOON 2018, Qing Dao, China, July 2–4, 2018, proceedings, lecture notes in computer science, vol 10976. Springer, pp 242–254. https://doi.org/10.1007/978-3-319-94776-1_21

Luo K, Erlebach T, Xu Y (2018b) Car-sharing between two locations: online scheduling with two servers. In: Potapov I, Spirakis PG, Worrell J (eds) 43rd International symposium on mathematical foundations of computer science, MFCS 2018, August 27–31, 2018, Liverpool, LIPIcs, vol 117. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp 50:1–50:14. https://doi.org/10.4230/LIPIcs.MFCS.2018.50

Luo K, Erlebach T, Xu Y (2018c) Online scheduling of car-sharing requests between two locations with many cars and flexible advance bookings. In: Hsu W, Lee D, Liao C (eds) 29th International symposium on algorithms and computation, ISAAC 2018, December 16–19, 2018, Jiaoxi, Yilan, Taiwan, LIPIcs, vol 123. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp 64:1–64:13. https://doi.org/10.4230/LIPIcs.ISAAC.2018.64

Luo K, Erlebach T, Xu Y (2019) Car-sharing on a star network: On-line scheduling with k servers. In: Niedermeier, R, Paul C (eds) 36th International symposium on theoretical aspects of computer science, STACS 2019, March 13–16, 2019, Berlin, Germany, LIPIcs, vol 126. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp 51:1–51:14. https://doi.org/10.4230/LIPIcs.STACS.2019.51

The future of driving: Seeing the back of the car (2012) https://www.economist.com/briefing/2012/09/22/seeing-the-back-of-the-car