Improved Bounds for Open Online Dial-a-Ride on the Line

Alexander Birx

Institute of Mathematics and Graduate School CE, TU Darmstadt, Germany
birx@gsc.tu-darmstadt.de

Yann Disser

Institute of Mathematics, TU Darmstadt, Germany
disser@mathematik.tu-darmstadt.de

Kevin Schewior

Institut für Informatik, Technische Universität München, Garching, Germany
kschewior@gmail.com

Abstract

We consider the open, non-preemptive online Dial-a-Ride problem on the real line, where transportation requests appear over time and need to be served by a single server. We give a lower bound of 2.0585 on the competitive ratio, which is the first bound that strictly separates online Dial-a-Ride on the line from online TSP on the line in terms of competitive analysis, and is the best currently known lower bound even for general metric spaces. On the other hand, we present an algorithm that improves the best known upper bound from 2.9377 to 2.6662. The analysis of our algorithm is tight.

2012 ACM Subject Classification Theory of computation → Online algorithms; Mathematics of computing → Combinatorial optimization

Keywords and phrases dial-a-ride on the line, elevator problem, online algorithms, competitive analysis, smartstart, competitive ratio

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.21

Category APPROX

Related Version A full version of the paper is available at `http://arxiv.org/abs/1907.02858`.

Funding *Alexander Birx*: This work was supported by the "Excellence Initiative" of the German Federal and State Governments and the Graduate School CE at TU Darmstadt.
Yann Disser: This work was supported by the "Excellence Initiative" of the German Federal and State Governments and the Graduate School CE at TU Darmstadt.
Kevin Schewior: Supported by the DAAD within the PRIME program using funds of BMBF and the EU Marie Curie Actions.

1 Introduction

We consider the online Dial-a-Ride problem on the line, where transportation requests appear over time and need to be transported to their respective destinations by a single server. More precisely, each request is of the form $\sigma_i = (a_i, b_i; r_i)$ and appears in position $a_i \in \mathbb{R}$ along the real line at time $r_i \geq 0$ and needs to be transported to position $b_i \in \mathbb{R}$. The server starts at the origin, can move at unit speed, and has a capacity $c \in \mathbb{N} \cup \{\infty\}$ that bounds the number of requests it can carry simultaneously. The objective is to minimize the completion time, i.e., the time until all requests have been served. In this paper, we focus on the *non-preemptive* and *open* setting, where the former means that requests can only be unloaded at their destinations, and the latter means that we do not require the server to return to the origin after serving all requests.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019).
Editors: Dimitris Achlioptas and László A. Végh; Article No. 21; pp. 21:1–21:22
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

We aim to bound the *competitive ratio* of the problem, i.e., the smallest ratio any online algorithm can guarantee between the completion time of its solution compared to an (offline) optimum solution that knows all requests ahead of time. To date, the best known lower bound of 2.0346 on this ratio was shown by Bjelde et al. [5], already for online TSP, where $a_i = b_i$ for all requests (i.e., requests only need to be visited). The best known upper bound of 2.9377 was achieved by the SMARTSTART algorithm [4].

Our results. Our first result is an improved lower bound for online DIAL-A-RIDE on the line. Importantly, since the bound of roughly 2.0346 was shown to be tight for online TSP [5], our new bound is the first time that DIAL-A-RIDE on the line can be strictly separated from online TSP in terms of competitive analysis. In addition, our bound is the currently best known lower bound even for general metric spaces. Specifically, we show the following.

▶ **Theorem 1.1.** *Let $\rho \approx 2.0585$ be the second largest root of the polynomial $4\rho^3 - 26\rho^2 + 39\rho - 5$. There is no $(\rho - \varepsilon)$-competitive algorithm for open, non-preemptive ($c < \infty$) online* DIAL-A-RIDE *on the line for any $\varepsilon > 0$.*

Our construction is a non-trivial variation of the construction achieving roughly 2.0346 for online TSP [5]. This construction is comprised of an initial request, a first stage consisting in turn of different iterations, and a second stage. We show that, by using a proper transportation requests as initial requests, we can adapt a single iteration of the first stage as well as the second stage to achieve the bound of roughly 2.0585 in the DIAL-A-RIDE setting.

Our second result is an improved algorithm SMARTERSTART for online DIAL-A-RIDE on the line. This algorithm improves the waiting strategy of the SMARTSTART algorithm, which was identified as a weakness in [4]. We show that this modification improves the competitive ratio of the algorithm and give a tight analysis. Specifically, we show the following.

▶ **Theorem 1.2.** *The competitive ratio of* SMARTERSTART *is (roughly)* 2.6662*.*

The general idea of SMARTERSTART is to improve the tradeoff between the case when the algorithm waits before starting its final schedule and the case when it starts the final schedule immediately. Our modification of SMARTSTART significantly improves the performance in the former case, while only moderately degrading the performance in the latter case. Overall, this results in an improved worst-case performance.

Related Work. The online DIAL-A-RIDE problem has received considerable attention in the past (e.g. [1, 4, 5, 6, 9, 13]). Table 1 gives an overview of the currently best known bounds on the line for open online DIAL-A-RIDE and its special case open online TSP.

The following results are known for *closed* online DIAL-A-RIDE: For general metric spaces, the competitive ratio is exactly 2, both for online DIAL-A-RIDE as well as online TSP [1, 3, 9]. On the line, a better upper bound is known only for online TSP, where the competitive ratio is exactly $(9 + \sqrt{17})/8 \approx 1.6404$ [3, 5]. The best known lower bound for closed, non-preemptive DIAL-A-RIDE on the line is 1.75 [5].

When the objective is to minimize the maximum flow time, on many metric spaces no online algorithm can be competitive [15, 16]. Hauptmeier et al. [12] showed that a competitive algorithm is possible if we restrict ourselves to instances with "reasonable" load. Yi and Tian [18] considered online DIAL-A-RIDE with deadlines, where the objective is to maximize the number of requests that are served in time. Other interesting variants of online DIAL-A-RIDE where destinations of requests are only revealed upon their collection were studied by Lipmann et al. [17] as well as Yi and Tian [19].

Table 1 Overview of the best known bounds for online Dial-a-Ride on the line (top), and online Dial-a-Ride on general metric spaces (bottom). Results are split into the non-preemptive case (with $c < \infty$), the preemptive case, and the TSP-case, where source and destination of each request coincide. Bold results are original, all other results follow immediately.

		open		closed	
		lower bound	upper bound	lower bound	upper bound
line	non-preemptive	**2.0585** (Thm 1.1)	**2.6662** (Thm 1.2)	**1.75** [5]	2
	preemptive	2.04	**2.41** [5]	1.64	2
	TSP	**2.04** [5]	**2.04** [5]	**1.64** [3]	**1.64** [5]
general	non-preemptive	2.0585 (Thm 1.1)	**3.41** [14]	2	**2** [1, 9]
	preemptive	2.04	3.41	2	2
	TSP	2.04	**2.5** [3]	**2** [3]	**2** [3]

For an overview of results for the offline version of Dial-a-Ride on the line, see [8]. Without release times, Gilmore and Gomory [10] and Atallah and Kosaraju [2] gave a polynomial time algorithm for closed, non-preemptive Dial-a-Ride on the line with capacity $c = 1$. Guan [11] showed that the closed, non-preemptive problem is hard for $c = 2$, and Bjelde et al. [5] extended this result for any finite capacity $c \geq 2$ in both the open and the closed variant. Bjelde et al. [5] also showed that the problem with release times is already hard for finite $c \geq 1$ in both variants, and Krumke [14] gave a 3-approximation algorithm for the closed variant. The complexity for the case $c = \infty$ remains open. For closed, preemptive Dial-a-Ride on the line without release times, Atallah and Kosaraju [2] gave a polynomial time algorithm for $c = 1$ and Guan [11] for $c \geq 2$. Charikar and Raghavachari [7] presented approximation algorithms for the closed case without release times on general metric spaces.

2 General Lower Bound

In this section, we prove Theorem 1.1. Let $c < \infty$ and Alg be a deterministic online algorithm for open online Dial-a-Ride. Let $\rho \approx 2.0585$, be the second largest root of the polynomial $4\rho^3 - 26\rho^2 + 39\rho - 5$. We describe a request sequence σ_ρ such that $\text{Alg}(\sigma_\rho) \geq \rho\text{Opt}(\sigma_\rho)$.

We first give a high-level description of our construction disregarding many technical details. Our construction is based on that in [5] for the TSP version of the problem. That construction consists of two *stages*: After an initial request $(1, 1; 1)$ (assuming w.l.o.g. Alg's position at time 1 is at most 0), the first stage starts. This stage consists of a loop, which ends as soon as two so-called critical requests are established. The second stage consists of augmenting the critical requests by suitable additional ones to show the desired competitive ratio. A single iteration of the loop only yields a lower bound of roughly 2.0298, but as the number of iterations approaches infinity one can show the tight bound of roughly 2.0346 in the limit.

In the Dial-a-Ride setting, we show a lower bound of roughly 2.0585 using the same general structure but only a single iteration. Our additional leeway stems from replacing the initial request $(1, 1; 1)$ with c initial requests of the form $(1, \delta; 1)$ where $\delta > 1$: At the time when an initial request is loaded, we show that w.l.o.g. all c requests are loaded and then proceed as we did when $(1, 1; 1)$ was served. In the new situation, the algorithm has to first deliver the c initial requests to be able to serve additional requests. For the optimum, the two situations however do not differ, because in the new situation there will be an additional request to the right of δ later anyway. Interestingly, this leeway turns out to be sufficient not only to create critical requests (w.r.t. a slightly varied notion of criticality) for a competitive ratio of larger than 2.0298 but even strictly larger than 2.0346. The second stage has to be slightly adapted to match the new notion of criticality. It remains unclear how to use multiple iterations in our setting.

We start by making observations that will simplify the exposition. Consider a situation in which the server is fully loaded. First note that it is essentially irrelevant whether we assume that the server, without delivering any of the loaded requests, can still serve requests $(a_i, b_i; t_i)$ for which $a_i = b_i$: If it can, we simply move a_i and b_i by $\varepsilon > 0$ apart, forbidding the server to serve it before delivering one of the loaded requests first. Therefore, we assume for simplicity that, when fully loaded, the server has to first deliver a request before it can serve any other one. We note that, in our construction, the above idea can be implemented without loss, not even in terms of ε.

The latter discussion also motivates restricting the space of considered algorithms: We call Alg *eager* if it, when fully loaded with requests with identical destinations, immediately delivers these requests without detour. It is clear that we can transform every algorithm Alg′ into an eager algorithm $\text{Alg}'_{\text{eager}}$ by letting it deliver the requests right away, waiting until Alg′ would have delivered them, and then letting it continue like Alg′. Since Alg′ cannot collect or serve other requests while being fully loaded, we have $\text{Alg}'_{\text{eager}}(\sigma) \leq \text{Alg}'(\sigma)$ for every request sequence σ.

▶ **Observation 2.1.** *Every algorithm for online* Dial-a-Ride *can be turned into an eager algorithm with the same competitive ratio.*

Thus, we may assume that Alg is eager. We now consider the second stage and then design a first stage to match the second stage. Suppose we have two requests $\sigma^R = (t^R, t^R; t^R)$ and $\sigma^L = (-t^L, -t^L, t^L)$ with $t^L \leq t^R$ to the right and to the left of the origin, respectively. We assume that Alg serves σ^R first at some time $t^* \geq (2\rho - 2)t^L + (\rho - 2)t^R$. Now suppose we could force Alg to serve σ^L directly after σ^R, even if additional requests are released. Then we could just release the request $\sigma^R_* = (t^R, t^R, 2t^L + t^R)$ and we would have

$$\text{Alg}(\sigma_\rho) = t^* + 2t^L + 2t^R \geq 2\rho t^L + \rho t^R = \rho \text{Opt}(\sigma_\rho),$$

since Opt can serve the three requests in time $2t^L + t^R$ by serving σ^L first. In fact, we will show that we can force Alg into this situation (or a worse situation) if the requests $\sigma^R = (t^R, t^R; t^R)$ and $\sigma^L = (-t^L, -t^L, t^L)$ satisfy the following properties. To describe the trajectory of a server, we use the notation "move(a)" for the tour that moves the server from its current position with unit speed to the point $a \in \mathbb{R}$.

▶ **Definition 2.2.** *We call the last two requests $\sigma^R = (t^R, t^R; t^R)$ and $\sigma^L = (-t^L, -t^L, t^L)$ of a request sequence with $0 < t^L \leq t^R$* critical *for* Alg *if the following conditions hold:*

(i) *Both tours move($-t^L$)$\oplus$move(t^R) and move(t^R)$\oplus$move($-t^L$) serve all requests presented until time t^R.*

(ii) Alg *serves both σ^R and σ^L after time t^R and* Alg*'s position at time t^R lies between t^R and $-t^L$.*

(iii) *If* Alg *serves σ^R before σ^L, it does so no earlier than $t^R_* := (2\rho - 2)t^L + (\rho - 2)t^R$.*

(iv) *If* Alg *serves σ^L before σ^R, it does so no earlier than $t^L_* := (2\rho - 2)t^R + (\rho - 2)t^L$.*

(v) *It holds that $\frac{t^R}{t^L} \leq \frac{4\rho^2 - 30\rho + 50}{-8\rho^2 + 50\rho - 66}$.*

▶ **Lemma 2.3.** *If there is a request sequence with two critical requests for* Alg*, we can release additional requests such that* Alg *is not* $(\rho - \varepsilon)$*-competitive on the resulting instance for any* $\varepsilon > 0$.

Definition 2.2 differs from [5, Definition 5] only in property (v), which is $\frac{t^R}{t^L} \leq 2$ in the original paper. Lemma 2.3 has been proved in [5, Lemma 6] for request sequences that satisfy the properties of [5, Definition 5], however, a careful inspection of the proof of [5, Lemma 6] shows that the statement of Lemma 2.3 also holds for request sequences that only satisfy (v) instead of $\frac{t^R}{t^L} \leq 2$. For a detailed proof, see Appendix A. Thus, our goal is to construct a request sequence σ_ρ that satisfies all properties of Definition 2.2.

The remaining part of this section focusses on establishing critical requests. There are no requests released until time 1. Without loss of generality, we assume that Alg's position at time 1 is $\text{pos}\,(1) \leq 0$ (the other case is symmetric). Here and throughout, we let $\text{pos}\,(t)$ denote the position of Alg's server at time t. Now, let

$$\delta := \frac{3\rho^2 - 11}{-3\rho^3 + 15\rho - 4}$$

and let c initial requests $\sigma^R_{(j)} = (1, \delta; 1)$ with $j \in \{1, \ldots, c\}$ appear. These are the only requests appearing in the entire construction with a starting point differing from the destination. We make a basic observation on how Alg has to serve these requests.[1]

▶ **Lemma 2.4.** Alg *cannot collect any of the requests* $\sigma^R_{(j)}$ *before time* 2*. If* Alg *collects the requests after time* $\rho\delta - (\delta - 1)$ *or serves* $c' < c$ *requests before loading the remaining* $c - c'$*, it is not* $(\rho - \varepsilon)$*-competitive.*

We hence may assume that Alg loads all c requests $\sigma^R_{(j)}$ at the same time. Let $t^L \in [2, \rho\delta - (\delta - 1))$ be the time Alg loads the c requests $\sigma^R_{(j)}$. We start the first stage and present a variant of a single iteration of the construction in [5]: We let the request $\sigma^L = (-t^L, -t^L; t^L)$ appear and define the function

$$\ell(t) = (4 - \rho) \cdot t - (2\rho - 2) \cdot t^L,$$

which can be viewed as a line in the path-time diagram. Because of $\rho > 2$, we have $\ell(t^L) = (6 - 3\rho)t^L < 0 < \text{pos}\left(t^L\right)$, i.e., Alg's position at time t^L is to the right of the line ℓ. Thus, Alg crosses the line ℓ before it serves σ^L. Let t^R be the time Alg crosses ℓ for the first time and let the request $\sigma^R = (t^R, t^R; t^R)$ appear. Assume Alg crosses the line ℓ and serves σ^R before σ^L. Then it does not serve σ^R before time

$$t^R + |\ell(t^R) - t^R| = (2\rho - 2)t^L + (\rho - 2)t^R = t^R_*. \tag{1}$$

Now assume Alg crosses ℓ at time $t^R \geq \frac{3\rho-5}{7-3\rho}t^L$ and serves σ^L before σ^R. Then it does not serve serve σ^L before time

$$\begin{aligned} t^R + |\ell(t^R) - (-t^L)| &= (5 - \rho)t^R - (2\rho - 3)t^L \\ &\geq (2\rho - 2)t^R + (7 - 3\rho)\frac{3\rho - 5}{7 - 3\rho}t^L - (2\rho - 3)t^L \\ &= (2\rho - 2)t^R + (\rho - 2)t^L = t^L_*. \end{aligned} \tag{2}$$

The following lemma shows that the two requests cannot be served before these respective times by establishing that indeed $t^R \geq \frac{3\rho-5}{7-3\rho}t^L$.

[1] The full proof and other omitted proofs can be found at `http://arxiv.org/abs/1907.02858`.

▶ **Lemma 2.5.** Alg *can neither serve* σ^L *before time* t^L_* *nor can it serve* σ^R *before time* t^R_*.

Proof. Since Alg is eager, it delivers the c requests $\sigma^R_{(j)}$ without waiting or detour, i.e., we have $\text{pos}\left(t^L + (\delta - 1)\right) = \delta$. Furthermore, we have

$$\begin{aligned} \ell(t^L + (\delta - 1)) &= (4-\rho)(t^L + (\delta-1)) - (2\rho-2)t^L \\ &= (6-3\rho)t^L + (4-\rho)(\delta-1) \\ &\leq (6-3\rho)(\rho\delta - (\delta-1)) + (4-\rho)(\delta-1) \\ &= \frac{3\rho^4 - 18\rho^3 + 3\rho^2 + 50\rho - 14}{3\rho^3 - 15\rho + 4} \\ &\overset{\rho < 2.06}{<} \delta = \text{pos}\left(t^L + (\delta-1)\right), \end{aligned}$$

i.e., Alg's position at time $t^L + (\delta - 1)$ is to the right of ℓ. The earliest possible time Alg crosses ℓ is the solution of

$$\ell(t^R) = (4-\rho)t^R - (2\rho-2)t^L = \text{pos}\left(t^L + (\delta-1)\right) + t^L + (\delta-1) - t^R,$$

which is $t^R = \frac{2\rho-1}{5-\rho}t^L + \frac{2\delta-1}{5-\rho}$. The inequality

$$\begin{aligned} \left(\frac{3\rho-5}{7-3\rho} - \frac{2\rho-1}{5-\rho}\right)t^L &= \frac{3\rho^2+3\rho-18}{3\rho^2-22\rho+35}t^L \\ &\leq \frac{3\rho^2+3\rho-18}{3\rho^2-22\rho+35}(\rho\delta - (\delta-1))) \\ &= \frac{3\rho^3+6\rho^2-15\rho-18}{3\rho^4-15\rho^3-15\rho^2+79\rho-20} \\ &= \frac{2\delta-1}{5-\rho}, \end{aligned}$$

implies that we have

$$t^R \geq \frac{3\rho-5}{7-3\rho}t^L. \tag{3}$$

Because of inequality (1) Alg does not serve σ^R before t^R_* and because of the inequalities (3) and (2) it does not serve σ^L before time t^L_*. ◀

In fact, also the other properties of critical requests are satisfied.

▶ **Lemma 2.6.** *The requests* σ^R *and* σ^L *of the request sequence* σ_ρ *are critical.*

Proof. We have to show that the requests σ^R and σ^L of the request sequence σ_ρ satisfy the properties (i) to (v) of Definition 2.2. The release time of every request is equal to its starting position, thus every request can be served/loaded immediately once its starting position is visited and (i) of Definition 2.2 is satisfied. At time t^R Alg has not served σ^R, because for that it would have needed to go right from time 0 on; it has not served σ^L either, because during the period of time $[t_L, t_R]$ Alg and σ^L were on different sides of ℓ. This establishes the first part of (ii) of Definition 2.2. Furthermore at time t^R Alg is at position $\text{pos}\left(t^R\right) = (4-\rho)t^R - (2\rho-2)t^L$ with

$$-t^L \leq (4-\rho)t^R - (2\rho-2)t^L \leq t^R$$

Therefore, the second part of (ii) of Definition 2.2 is satisfied as well.

Lemma 2.5 shows that (iii) and (iv) of Definition 2.2 are satisfied. It remains to show that property (v) is satisfied. For this we need to examine the release time t^R of σ^R. The time t^R is largest if ALG tries to avoid crossing the line ℓ for as long as possible, i.e., it continues to move right after serving the requests $\sigma^R_{(j)}$. Then, we have $\text{pos}\,(t) = 1 - t^L + t$ for $t \in [t^L, t^R]$ and t^R is the solution of

$$1 - t^L + t^R = (4 - \rho)t^R - (2\rho - 2)t^L.$$

Thus, in general, we have $t^R \leq \frac{2\rho-3}{3-\rho}t^L + \frac{1}{3-\rho}$, i.e.,

$$\frac{t^R}{t^L} \leq \frac{2\rho - 3}{3 - \rho} + \frac{1}{(3-\rho)t^L} \overset{t^L \geq 2}{\leq} \frac{4\rho - 5}{6 - 2\rho}. \tag{4}$$

For property (v), we need $\frac{t^R}{t^L} \leq \frac{4\rho^2-30\rho+50}{-8\rho^2+50\rho-66}$. This is satisfied if

$$\frac{4\rho - 5}{6 - 2\rho} \leq \frac{4\rho^2 - 30\rho + 50}{-8\rho^2 + 50\rho - 66},$$

which is equivalent to

$$4\rho^3 - 26\rho^2 + 39\rho - 5 \geq 0,$$

which is true by definition of ρ. ◀

Together with Lemma 2.3, this completes the proof of Theorem 1.1.

3 An Improved Algorithm

One of the simplest approaches for an online algorithm to solve DIAL-A-RIDE is the following: Always serve the set of currently unserved requests in an optimum offline schedule and ignore all new incoming request while doing so. Afterwards, repeat this procedure with all ignored unserved requests until no new requests arrive. This simple algorithm that is often called IGNORE [1] has a competitive ratio of exactly 4 [4, 14]. The main weakness of IGNORE is that it always starts its schedule immediately. Ascheuer et al. showed that it is beneficial if the server waits sometimes before starting a schedule and introduced the SMARTSTART algorithm [1], which has a competitive ratio of roughly 2.94 [4].

We define $L(t, p, R)$ to be the smallest makespan of a schedule that starts at position p at time t and serves all requests in $R \subseteq \sigma$ after they appeared (i.e., the schedule must respect release times). For the description of online algorithms, we denote by t the current time and by R_t the set of requests that have appeared until time t but have not been served yet.

The algorithm SMARTSTART is given in Algorithm 1. Essentially, at time t, SMARTSTART waits before starting an optimal schedule to serve all available requests at time

$$\min_{t' \geq t} \left\{ t' \geq \frac{L(t', p, R_{t'})}{\Theta - 1} \right\}, \tag{5}$$

where p is the current position of the server and $\Theta > 1$ is a parameter of the algorithm that scales the waiting time. Importantly, like IGNORE, SMARTSTART ignores incoming requests while executing a schedule.

Birx and Disser identified that SMARTSTART's waiting routine defined by inequality (5) has a critical weakness [4, Lemma 4.1]. It is possible to lure the server to any position q in time $q + \varepsilon$ for every $\varepsilon > 0$. Roughly speaking, a request $\sigma_1 = ((\Theta - 1)\varepsilon, (\Theta - 1)\varepsilon; (\Theta - 1)\varepsilon)$ is

Algorithm 1 SMARTSTART.

```
p_1 ← 0
for j = 1, 2, ... do
    while current time t < L(t, p_j, R_t)/(Θ − 1) do
        wait
    t_j ← t
    S_j ← optimal offline schedule serving R_t starting from p_j
    execute S_j
    p_{j+1} ← current position
```

released first and then for every $i \in \{2, \dots, \frac{q}{\varepsilon}\}$ a request $\sigma_i = (i\varepsilon, i\varepsilon; i\varepsilon)$ follows. The schedule to serve the request σ_1 is started at time ε and finished at time 2ε. The schedule to serve the request at position $i\varepsilon$ is not started earlier than time

$$\frac{L(i\varepsilon, (i-1)\varepsilon, \{\sigma_i\})}{\Theta - 1} = \frac{|(i-1)\varepsilon - i\varepsilon|}{\Theta - 1} = \frac{\varepsilon}{\Theta - 1}. \tag{6}$$

This time is (depending on the choice of Θ) later than the current time $i\varepsilon$ for every $i \geq 2$. Thus there is no waiting time for any schedule except the first one and the server reaches position q at time $q+\varepsilon$. We see that the request sequence to lure the server away heavily uses that inequality (5) relies on SMARTSTART's current position p, when computing the waiting time. Thus, we modify the waiting routine of SMARTSTART to avoid luring accordingly. Denote by $\sigma_{\leq t}$ the set of requests that have been released until time t.

Algorithm 2 SMARTERSTART.

```
p_1 ← 0
for j = 1, 2, ... do
    while current time t < L(t, 0, σ_{≤t})/(Θ − 1) do
        wait
    t_j ← t
    S_j ← optimal offline schedule serving R_t starting from p_j
    execute S_j
    p_{j+1} ← current position
```

The improved algorithm SMARTERSTART is given in Algorithm 2. At time t, it waits before starting an optimal schedule to serve all available requests at time

$$\min_{t' \geq t} \left\{ t' \geq \frac{L(t', 0, \sigma_{\leq t'})}{\Theta - 1} \right\}. \tag{7}$$

Again, $\Theta > 1$ is a parameter of the algorithm that scales the waiting time. In contrast to SMARTSTART, the waiting time is dependent on the length of the optimum offline schedule serving all requests appeared until the current time and starting from the origin. This guarantees that the server cannot be forced to reach any position q before time $q/(\Theta - 1)$ since we always have $L(t, 0, \sigma_{\leq t}) > q$ if $\sigma_{\leq t}$ contains a request with destination in position q.

Whenever we need to distinguish the behavior of SMARTERSTART for different values of $\Theta > 1$, we write SMARTERSTART$_\Theta$ to make the choice of Θ explicit. The length of SMARTERSTART's trajectory is denoted by SMARTERSTART(σ). Note that the schedules used by IGNORE, SMARTSTART and SMARTERSTART are NP-hard to compute for $1 < c < \infty$, see [5].

We let $N \in \mathbb{N}$ be the number of schedules needed by SmarterStart to serve σ. The j-th schedule is denoted by S_j, its starting time by t_j, its starting point by p_j, its ending point by p_{j+1}, and the set of requests served in S_j by σ_{S_j}. For convenience, we set $t_0 = p_0 = 0$.

3.1 Upper Bound for SmarterStart

We show the upper bound of Theorem 1.2. The completion time of SmarterStart is

$$\textsc{SmarterStart}(\sigma) = t_N + L(t_N, p_N, \sigma_{S_N}). \tag{8}$$

First, observe that, for all $0 \leq t \leq t'$, $p, p' \in \mathbb{R}$, and $R \subseteq \sigma$, we have

$$L(t, p, R) \geq L(t', p, R), \tag{9}$$

$$L(t, p, R) \leq |p - p'| + L(t, p', R), \tag{10}$$

$$L(t, 0, \sigma_{\leq t}) \leq L(t, 0, \sigma) \leq L(0, 0, \sigma) \leq \textsc{Opt}(\sigma). \tag{11}$$

Similar to [4], we distinguish between two cases, depending on whether or not SmarterStart waits after finishing schedule S_{N-1} and before starting the final schedule S_N. If the algorithm SmarterStart waits, the starting time of schedule S_N is given by

$$t_N = \frac{1}{\Theta - 1} L(t_N, 0, \sigma_{\leq t_N}), \tag{12}$$

otherwise, we have

$$t_N = t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}). \tag{13}$$

We start by giving a lower bound on the starting time of a schedule. It was shown in [4] that the schedule S_j of SmartStart is never started earlier than time $\frac{|p_{j+1}|}{\Theta}$. This changes slightly for SmarterStart.[1]

▶ **Lemma 3.1.** *Algorithm* SmarterStart *does not start schedule S_j earlier than time $\frac{|p_{j+1}|}{\Theta - 1}$, i.e., we have $t_j \geq \frac{|p_{j+1}|}{\Theta - 1}$.*

Using Lemma 3.1, we can give an upper bound on the length of SmarterStart's schedules, which is an essential ingredient in our upper bounds. The following lemma is proved similarly to [4, Lemma 3.2], which yields an upper bound of $(1 + \frac{\Theta}{\Theta+2})\textsc{Opt}(\sigma)$ for the length of every schedule S_j of SmartStart.[1]

▶ **Lemma 3.2.** *For every schedule S_j of* SmarterStart, *we have*

$$L(t_j, p_j, \sigma_{S_j}) \leq \left(1 + \frac{\Theta - 1}{\Theta + 1}\right) \textsc{Opt}(\sigma).$$

Proof sketch. To proof the claim we have to show the two inequalities

$$L(t_j, p_j, \sigma_{S_j}) \leq \textsc{Opt}(\sigma) + |p_j| \quad \text{and} \quad L(t_j, p_j, \sigma_{S_j}) \leq 2\textsc{Opt}(\sigma) - 2\frac{|p_j|}{\Theta - 1}. \tag{14}$$

This implies

$$\begin{aligned} L(t_j, p_j, \sigma_{S_j}) &\overset{(14)}{\leq} \min\left\{\textsc{Opt}(\sigma) + |p_j|, 2\textsc{Opt}(\sigma) - \frac{2}{\Theta - 1}|p_j|\right\} \\ &\leq \left(1 + \frac{\Theta - 1}{\Theta + 1}\right) \textsc{Opt}(\sigma), \end{aligned}$$

since the minimum above is largest for $|p_j| = \frac{\Theta - 1}{\Theta + 1}\textsc{Opt}(\sigma)$. ◀

The following proposition uses Lemma 3.2 to provide an upper bound for the competitive ratio of SmarterStart, in the case that SmarterStart does have a waiting period before starting the final schedule.

▶ **Proposition 3.3.** *In case* SmarterStart *waits before executing S_N, we have*

$$\frac{\text{SmarterStart}(\sigma)}{\text{Opt}(\sigma)} \leq f_1(\Theta) := \frac{2\Theta^2 - \Theta + 1}{\Theta^2 - 1}.$$

Proof. Assume SmarterStart waits before starting the final schedule. Lemma 3.2 yields the claimed bound:

$$\begin{aligned}
\text{SmarterStart}(\sigma) &\overset{(8)}{=} t_N + L(t_N, p_N, \sigma_{S_N}) \\
&\overset{(12)}{=} \frac{1}{\Theta - 1} L(t_N, 0, \sigma_{\leq t_N}) + L(t_N, p_N, \sigma_{S_N}) \\
&\overset{(11)}{\leq} \frac{1}{\Theta - 1} \text{Opt}(\sigma) + L(t_N, p_N, \sigma_{S_N}) \\
&\overset{\text{Lem. 3.2}}{\leq} \left(\frac{1}{\Theta - 1} + 1 + \frac{\Theta - 1}{\Theta + 1}\right) \text{Opt}(\sigma) \\
&= \frac{2\Theta^2 - \Theta + 1}{\Theta^2 - 1} \text{Opt}(\sigma).
\end{aligned}$$ ◀

In comparison, the upper bound for the competitive ratio of Smartstart, in case Smartstart has a waiting period before starting the final schedule is $\frac{2\Theta^2+2\Theta}{\Theta^2+\Theta-2}\text{Opt}(\sigma)$ [4, Proposition 3.2]. Note that SmarterStart's bound is better than Smartstart's bound for $\Theta > 1$.

It remains to examine the case that the algorithm SmarterStart has no waiting period before starting the final schedule. For this we use two lemmas from [4] originally proved for Smartstart, which are still valid for SmarterStart since they give bounds on the optimum offline schedules independently of the waiting routine.

By $x_- := \min\{0, \min_{i=1,\dots,n}\{a_i\}, \min_{i=1,\dots,n}\{b_i\}\}$ we denote the leftmost position that needs to be visited by the server and by $x_+ := \max\{0, \max_{i=1,\dots,n}\{a_i\}, \max_{i=1,\dots,n}\{b_i\}\}$ the rightmost. We denote by $y_-^{S_j}$ the leftmost and by $y_+^{S_j}$ the rightmost position that occurs in the requests σ_{S_j}. Note that $y_-^{S_j}$ and $y_+^{S_j}$ need not lie on different sides of the origin, in contrast to $x_{-/+}$.

▶ **Lemma 3.4** (Lemma 3.4, Full Version of [4]). *Let S_j with $j \in \{1, \dots, N\}$ be a schedule of* SmarterStart. *Moreover, let $\text{Opt}(\sigma) = |x_-| + x_+ + y$ for some $y \geq 0$. Then, we have*

$$L(t_j, 0, \sigma_{S_j}) \leq |\min\{0, y_-^{S_j}\}| + \max\{0, y_+^{S_j}\} + y.$$

▶ **Lemma 3.5** (Lemma 3.6, Full Version of [4]). *Let S_j with $j \in \{1, \dots, N\}$ be a schedule of* SmarterStart. *Moreover, let $|x_-| \leq x_+$ and $\text{Opt}(\sigma) = |x_-| + x_+ + y$ for some $y \geq 0$. Then, for every point p that is visited by S_j we have*

$$p \leq |p_j| + |p_j - p_{j+1}| + y - |\min\{0, y_-^{S_j}\}|.$$

Using the bounds established by Lemma 3.4 and Lemma 3.5, we can give an upper bound for the competitive ratio of SmarterStart if the server is not waiting before starting the final schedule.

▶ **Proposition 3.6.** *If* SmarterStart *does not wait before executing S_N, we have*

$$\frac{\text{SmarterStart}(\sigma)}{\text{Opt}(\sigma)} \leq f_2(\Theta) := \frac{3\Theta^2 + 3}{2\Theta + 1}.$$

Proof. Assume algorithm SmarterStart does not have a waiting period before the last schedule, i.e., SmarterStart starts the final schedule S_N immediately after finishing S_{N-1}. Without loss of generality, we assume $|x_-| \leq x_+$ throughout the entire proof by symmetry.

First of all, we notice that we may assume that SmarterStart executes at least two schedules in this case. Otherwise either the only schedule has length 0, which would imply $\text{Opt}(\sigma) = \text{SmarterStart}(\sigma) = 0$, or the only schedule would have a positive length, implying a waiting period. Let $\sigma_{S_N}^{\text{Opt}}$ be the first request of σ_{S_N} that is served by Opt and let a_N^{Opt} be its starting point and r_N^{Opt} be its release time. We have

$$\begin{aligned}\text{SmarterStart}(\sigma) &\overset{(8)}{=} t_N + L(t_N, p_N, \sigma_{S_N})\\ &\overset{(13)}{=} t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + L(t_N, p_N, \sigma_{S_N})\\ &\overset{t_N \geq r_N^{\text{Opt}}}{\leq} t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + L(r_N^{\text{Opt}}, p_N, \sigma_{S_N}). \end{aligned} \tag{15}$$

Since Opt serves all requests of σ_{S_N} after time r_N^{Opt}, starting with a request with starting point a_N^{Opt}, we also have

$$\text{Opt}(\sigma) \geq r_N^{\text{Opt}} + L(r_N^{\text{Opt}}, a_N^{\text{Opt}}, \sigma_{S_N}). \tag{16}$$

Furthermore, we have

$$r_N^{\text{Opt}} > t_{N-1} \tag{17}$$

since otherwise $\sigma_{S_N}^{\text{Opt}} \in \sigma_{S_{N-1}}$ would hold. This gives us

$$\begin{aligned}\text{SmarterStart}(\sigma) &\overset{(15)}{\leq} t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + L(r_N^{\text{Opt}}, p_N, \sigma_{S_N})\\ &\overset{(10)}{\leq} t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + |a_N^{\text{Opt}} - p_N|\\ &\quad + L(r_N^{\text{Opt}}, a_N^{\text{Opt}}, \sigma_{S_N})\\ &\overset{(16)}{\leq} t_{N-1} + L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + |a_N^{\text{Opt}} - p_N|\\ &\quad + \text{Opt}(\sigma) - r_N^{\text{Opt}}\\ &\overset{(17)}{<} L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + |a_N^{\text{Opt}} - p_N| + \text{Opt}(\sigma) &(18)\\ &\overset{(10)}{\leq} |p_{N-1}| + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + |a_N^{\text{Opt}} - p_N| + \text{Opt}(\sigma)\\ &\overset{\text{Lem. 3.1}}{\leq} (\Theta - 1)t_{N-2} + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + |a_N^{\text{Opt}} - p_N| + \text{Opt}(\sigma). &(19)\end{aligned}$$

We have

$$\text{Opt}(\sigma) \geq t_{N-2} + |a_N^{\text{Opt}} - p_N|, \tag{20}$$

because Opt has to visit both a_N^{Opt} and p_N after time t_{N-2}: It has to visit a_N^{Opt} to collect $\sigma_{S_N}^{\text{Opt}}$ and it has to visit p_N to deliver some request of $\sigma_{S_{N-1}}$. Using the above inequalitiy, we get

$$\begin{aligned}\text{SmarterStart}(\sigma) &\overset{(19)}{<} (\Theta - 1)t_{N-2} + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + |a_N^{\text{Opt}} - p_N| + \text{Opt}(\sigma)\\ &\overset{(20)}{\leq} 2\text{Opt}(\sigma) + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + (\Theta - 2)t_{N-2}. \end{aligned} \tag{21}$$

In the case $\Theta \geq 2$, we have

$$\begin{aligned}
\textsc{SmarterStart}(\sigma) &\overset{(21)}{<} 2\textsc{Opt}(\sigma) + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + (\Theta - 2)t_{N-2} \\
&\overset{(11)}{\leq} (\Theta + 1)\textsc{Opt}(\sigma) \\
&\overset{\Theta \geq 2}{\leq} \frac{3\Theta^2 + 3}{2\Theta + 1}\textsc{Opt}(\sigma).
\end{aligned}$$

Thus, we may assume $\Theta < 2$. Similarly as in inequality (21), we get

$$\begin{aligned}
\textsc{SmarterStart}(\sigma) &\overset{(19)}{<} (\Theta - 1)t_{N-2} + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + |a_N^{\textsc{Opt}} - p_N| + \textsc{Opt}(\sigma) \\
&\overset{(20)}{\leq} \Theta\textsc{Opt}(\sigma) + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + (2 - \Theta)|a_N^{\textsc{Opt}} - p_N| \\
&\overset{(7)}{\leq} \Theta\textsc{Opt}(\sigma) + (\Theta - 1)t_{N-1} + (2 - \Theta)|a_N^{\textsc{Opt}} - p_N| \\
&\leq (2\Theta - 1)\textsc{Opt}(\sigma) + (2 - \Theta)|a_N^{\textsc{Opt}} - p_N|, \qquad (22)
\end{aligned}$$

where the last inequality follows, because there exists a request in σ with release date later than t_{N-1}. This means the claim is shown if we have

$$|p_N - a_N^{\textsc{Opt}}| \leq \textsc{Opt}(\sigma) - \frac{\Theta - 1}{2\Theta + 1}\textsc{Opt}(\sigma) \qquad (23)$$

since then we have

$$\begin{aligned}
\textsc{SmarterStart}(\sigma) &\overset{(22)}{<} (2\Theta - 1)\textsc{Opt}(\sigma) + (2 - \Theta)|a_N^{\textsc{Opt}} - p_N| \\
&\overset{(23)}{\leq} (2\Theta - 1)\textsc{Opt}(\sigma) + (2 - \Theta)\left(1 - \frac{\Theta - 1}{2\Theta + 1}\right)\textsc{Opt}(\sigma) \\
&= \frac{3\Theta^2 + 3}{2\Theta + 1}\textsc{Opt}(\sigma).
\end{aligned}$$

Therefore, we may assume in the following that

$$|p_N - a_N^{\textsc{Opt}}| > \textsc{Opt}(\sigma) - \frac{\Theta - 1}{2\Theta + 1}\textsc{Opt}(\sigma). \qquad (24)$$

Let $\textsc{Opt}(\sigma) = |x_-| + x_+ + y$ for some $y \geq 0$. By definition of x_- and x_+ we have

$$|p_N - a_N^{\textsc{Opt}}| + y \leq \textsc{Opt}(\sigma). \qquad (25)$$

In the case that $\textsc{Opt}$ visits position p_N before it collects $\sigma_{S_N}^{\textsc{Opt}}$, we have

$$|a_N^{\textsc{Opt}} - p_N| + |p_N| \leq \textsc{Opt}(\sigma). \qquad (26)$$

Similarly, if $\textsc{Opt}$ collects $\sigma_{S_N}^{\textsc{Opt}}$ before it visits position p_N for the first time, we have

$$\begin{aligned}
\textsc{Opt}(\sigma) &\geq r_N^{\textsc{Opt}} + |a_N^{\textsc{Opt}} - p_N| \\
&\overset{(17)}{>} t_{N-1} + |a_N^{\textsc{Opt}} - p_N| \\
&\overset{\text{Lem. 3.1}}{\geq} \frac{|p_N|}{\Theta - 1} + |a_N^{\textsc{Opt}} - p_N| \\
&\overset{\Theta < 2}{\geq} |p_N| + |a_N^{\textsc{Opt}} - p_N|.
\end{aligned}$$

Thus, inequality (26) holds in general. To sum it up, we may assume that

$$\max\{y, |p_N|, t_{N-2}\} \overset{(24),(25),(26),(20)}{<} \frac{\Theta - 1}{2\Theta + 1}\textsc{Opt}(\sigma) \tag{27}$$

holds. In the following, denote by $y_-^{S_{N-1}}$ the leftmost starting or ending point and by $y_+^{S_{N-1}}$ the rightmost starting or ending point of the requests in $\sigma_{S_{N-1}}$. We compute

$$\begin{aligned}\textsc{SmarterStart}(\sigma) &\overset{(18)}{<} L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + |p_N - a_N^{\textsc{Opt}}| + \textsc{Opt}(\sigma)\\ &\overset{(26)}{<} L(t_{N-1}, p_{N-1}, \sigma_{S_{N-1}}) + 2\textsc{Opt}(\sigma) - |p_N|\\ &\overset{(9)}{\le} |p_{N-1}| + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + 2\textsc{Opt}(\sigma) - |p_N|\\ &\overset{\text{Lem. 3.1}}{\le} (\Theta - 1)t_{N-2} + L(t_{N-1}, 0, \sigma_{S_{N-1}}) + 2\textsc{Opt}(\sigma) - |p_N|\\ &\overset{\text{Lem. 3.4}}{\le} (\Theta - 1)t_{N-2} + \max\{0, |y_-^{S_{N-1}}|\} + \max\{0, y_+^{S_{N-1}}\} + y\\ &\quad +2\textsc{Opt}(\sigma) - |p_N|.\end{aligned} \tag{28}$$

Obviously, position $y_+^{S_{N-1}}$ is visited by SmarterStart in schedule S_{N-1}. Therefore, $y_+^{S_{N-1}}$ is smaller than or equal to the rightmost point that is visited by SmarterStart during schedule S_{N-1}, which gives us

$$y_+^{S_{N-1}} \overset{\text{Lem. 3.5}}{\le} |p_{N-1}| + |p_{N-1} - p_N| + y - \max\{0, |y_-^{S_{N-1}}|\}. \tag{29}$$

On the other hand, because of $|x_-| \le x_+$, we have $\textsc{Opt}(\sigma) \ge 2|x_-| + x_+$, which implies $y \ge |x_-|$. By definition of x_- and $y_-^{S_{N-1}}$, we have $|x_-| \ge \max\{0, |y_-^{S_{N-1}}|\}$. This gives us $y \ge \max\{0, |y_-^{S_{N-1}}|\}$ and

$$0 \le |p_{N-1}| + |p_{N-1} - p_N| + y - \max\{0, |y_-^{S_{N-1}}|\}. \tag{30}$$

To sum it up, we have

$$\max\{0, y_+^{S_{N-1}}\} \overset{(29),(30)}{\le} |p_{N-1}| + |p_{N-1} - p_N| + y - \max\{0, |y_-^{S_{N-1}}|\}. \tag{31}$$

The inequality above gives us

$$\begin{aligned}\textsc{SmarterStart}(\sigma) &\overset{(28)}{<} (\Theta - 1)t_{N-2} + \max\{0, |y_-^{S_{N-1}}|\} + \max\{0, y_+^{S_{N-1}}\}\\ &\quad +y + 2\textsc{Opt}(\sigma) - |p_N|\\ &\overset{(31)}{\le} (\Theta - 1)t_{N-2} + |p_{N-1}| + |p_{N-1} - p_N| + 2y + 2\textsc{Opt}(\sigma) - |p_N|\\ &\le (\Theta - 1)t_{N-2} + |p_{N-1}| + |p_{N-1}| + |p_N| + 2y + 2\textsc{Opt}(\sigma) - |p_N|\\ &\overset{\text{Lem. 3.1}}{\le} (\Theta - 1)t_{N-2} + 2(\Theta - 1)t_{N-2} + 2y + 2\textsc{Opt}(\sigma)\\ &\overset{(27)}{\le} (3\Theta - 3)\frac{\Theta - 1}{2\Theta + 1}\textsc{Opt}(\sigma) + 2\frac{\Theta - 1}{2\Theta + 1}\textsc{Opt}(\sigma) + 2\textsc{Opt}(\sigma)\\ &= \frac{3\Theta^2 + 3}{2\Theta + 1}\textsc{Opt}(\sigma).\end{aligned}$$

◀

In comparison, the upper bound for the competitive ratio of SmartStart in case it does not have a waiting period before starting the final schedule is $\Theta + 1 - \frac{\Theta - 1}{3\Theta + 3}\textsc{Opt}(\sigma)$ [4, Proposition 3.4]. Note that SmarterStart's bound is slightly worse than SmartStart's bound for $\Theta > 1.47$. However, in combination with the bound of Proposition 3.3, SmarterStart has a better worst-case than SmartStart.

▶ **Theorem 3.7.** *Let Θ^* be the largest solution of $f_1(\Theta) = f_2(\Theta)$, i.e.,*

$$\frac{3\Theta^{*2}+3}{2\Theta^*+1} = \frac{2\Theta^{*2}-\Theta^*+1}{\Theta^{*2}-1}.$$

Then, SmarterStart$_{\Theta^*}$ *is ρ^*-competitive with $\rho^* := f_1(\Theta^*) = f_2(\Theta^*) \approx 2.6662$.*

Proof. According to Proposition 3.3 and Proposition 3.6, if it exists,

$$\Theta^* = \operatorname*{argmin}_{\Theta>1} \{\max\{f_1(\Theta), f_2(\Theta)\}\}$$

is the parameter for SmarterStart with the smallest upper bound. We note that f_1 is strictly decreasing for $\Theta > 1$ and that f_2 is strictly increasing for $\Theta > 1$. Therefore, if an intersection point of f_1 and f_2 that is larger than 1 exists, then this is at Θ^*. Indeed, the intersection point exists, which is the largest solution of

$$\frac{3\Theta^2+3}{2\Theta+1} = \frac{2\Theta^2-\Theta+1}{\Theta^2-1}.$$

The resulting upper bound for the competitive ratio is

$$\rho^* = f_1(\Theta^*) = f_2(\Theta^*) \approx 2.6662.$$ ◀

3.2 Lower Bound for SmarterStart

We show the lower bound of Theorem 1.2. In this section, we explicitly construct instances that demonstrate that the upper bounds given in the previous section are tight for certain ranges of $\Theta > 1$, in particular for $\Theta = \Theta^*$ (as in Theorem 3.7). Further, we show that choices of $\Theta > 1$ different from Θ^* yield competitive ratios worse than $\rho^* \approx 2.67$. Together, this implies that ρ^* is exactly the best possible competitive ratio for SmarterStart.[1]

▶ **Proposition 3.8.** *Let $1 < \Theta < 2$. For every sufficiently small $\varepsilon > 0$, there is a set of requests σ such that* SmarterStart *waits before starting the final schedule and such that the inequality*

$$\frac{\text{SmarterStart}(\sigma)}{\text{Opt}(\sigma)} \geq \frac{2\Theta^2-\Theta+1}{\Theta^2-1} - \varepsilon$$

holds, i.e., the upper bound established in Proposition 3.3 is tight for $\Theta \in (1, 2)$.

Proof sketch. Let $\varepsilon > 0$ with $\varepsilon < \frac{\Theta}{\Theta+1}$ and $\varepsilon' = \frac{\Theta+1}{2\Theta}\varepsilon$. The request sequence $\sigma = \{\sigma_1, \sigma_2\}$ with

$$\sigma_1 = (1, 1; 0) \qquad \text{and} \qquad \sigma_2 = (-\frac{1}{\Theta-1} + \varepsilon', 1; \frac{1}{\Theta-1} + \varepsilon')$$

achieves the desired result. ◀

▶ **Proposition 3.9.** *Let $\frac{1}{2}(1+\sqrt{5}) \leq \Theta \leq 2$. For every sufficiently small $\varepsilon > 0$ there is a set of requests σ such that* SmarterStart *immediately starts S_N after S_{N-1} and such that*

$$\frac{\text{SmarterStart}(\sigma)}{\text{Opt}(\sigma)} \geq \frac{3\Theta^2+3}{2\Theta+1} - \varepsilon,$$

i.e., the upper bound established in Proposition 3.6 is tight for $\Theta \in [\frac{1}{2}(1+\sqrt{5}), 2] \approx [1.6180, 2]$.

Proof sketch. Let $\varepsilon > 0$ with $\varepsilon < \frac{1}{4}(\frac{5\Theta^2-9\Theta+4}{2\Theta+1})$ and $\varepsilon' = \frac{2\Theta+1}{5\Theta^2-9\Theta+4}\varepsilon$. The request sequence $\sigma = \{\sigma_1, \sigma_2\}$ with

$$\sigma_1 = (1, 1; 0),$$
$$\sigma_2^{(1)} = \left(2 + \frac{1}{\Theta-1} - 2\varepsilon', 2 + \frac{1}{\Theta-1} - 2\varepsilon'; \frac{1}{\Theta-1} + \varepsilon'\right),$$
$$\sigma_2^{(2)} = \left(-\frac{1}{\Theta-1}, -\frac{1}{\Theta-1}; \frac{1}{\Theta-1} + \varepsilon'\right),$$
$$\sigma_3 = \left(\frac{3}{(\Theta-1)^2} - \varepsilon', \frac{3}{(\Theta-1)^2} - \varepsilon'; \frac{3}{(\Theta-1)^2} + \frac{2}{\Theta-1}\right)$$

achieves the desired result. ◀

Recall that the optimal parameter Θ^* established in Theorem 3.7 is the only positive, real solution of the equation

$$\frac{3\Theta^2+3}{2\Theta+1} = \frac{2\Theta^2-\Theta+1}{\Theta^2-1},$$

which is $\Theta^* \approx 1.7125$. Therefore, according to Proposition 3.8 and Proposition 3.9 the parameter Θ^* lies in the range where the upper bounds of Propositions 3.3 and 3.6 are both tight. It remains to make sure that for all Θ that lie outside of this range the competitive ratio of SmarterStart$_\Theta$ is larger than $\rho^* \approx 2.6662$.[1]

▶ **Lemma 3.10.** *Let $\Theta > 2$. There is a set of requests $\sigma_{\Theta>2}$ such that*

$$\frac{\text{SmarterStart}(\sigma_{\Theta>2})}{\text{Opt}(\sigma_{\Theta>2})} > \rho^* \approx 2.6662.$$

Figure 1 shows the upper and lower bounds that we have established. Theorem 1.2 now follows from Theorem 3.7 combined with Propositions 3.8 and 3.9, as well as Lemma 3.10.

Proof of Theorem 1.2. We have shown in Proposition 3.8 that the upper bound

$$\frac{\text{SmarterStart}(\sigma)}{\text{Opt}(\sigma)} \leq f_1(\Theta) = \frac{2\Theta^2-\Theta+1}{\Theta^2-1}$$

established in Proposition 3.3 for the case, where SmarterStart waits before starting the final schedule, is tight for all $\Theta \in (1, 2)$. Furthermore, we have shown in Proposition 3.9 that the upper bound

$$\frac{\text{SmarterStart}(\sigma)}{\text{Opt}(\sigma)} \leq f_2(\Theta) = \frac{3\Theta^2+3}{2\Theta+1}$$

established in Proposition 3.6 for the case, where SmarterStart does not wait before starting the final schedule, is tight for all $\Theta \in (\frac{1}{2}(1+\sqrt{5}), 2]$. Since $\Theta^* \approx 1.71249$ lies in those ranges, the competitive ratio of SmarterStart$_{\Theta^*}$ is indeed exactly ρ^*.

It remains to show that for every $\Theta > 1$ with $\Theta \neq \Theta^*$ the competitive ratio is larger. First, according to Lemma 3.10, the competitive ratio of SmarterStart with parameter $\Theta \in (2, \infty)$ is larger than ρ^*. By monotonicity of f_1, every function value in $(1, \Theta^*)$ is larger than $f_1(\Theta^*) = \rho^*$. Thus, the competitive ratio of SmarterStart with parameter $\Theta \in (1, \Theta^*)$ is larger than ρ^*, since f_1 is tight on $(1, \Theta^*)$ by Proposition 3.8. Similarly, by monotonicity of f_2, every function value in $(\Theta^*, 2]$ is larger than $f_2(\Theta^*) = \rho^*$. Thus, the competitive ratio of SmarterStart with parameter $\Theta \in (\Theta^*, 2]$ is larger than ρ^*, since f_2 is tight on $(\Theta^*, 2]$ by Proposition 3.9. ◀

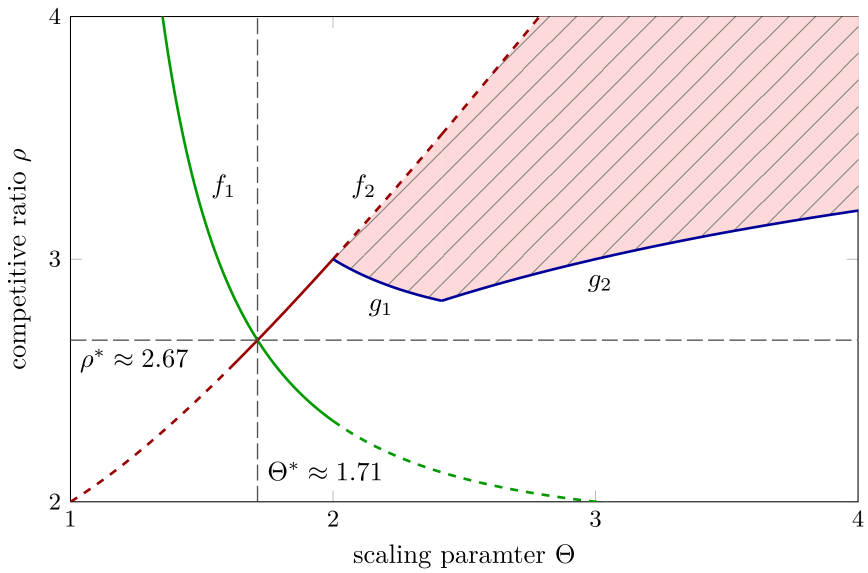

Figure 1 Overview of our bounds for SMARTERSTART. The functions f_1 (green) / f_2 (red) are upper bounds for the cases where SMARTERSTART waits / does not wait before starting the final schedule, respectively. The upper bounds are drawn solid in the domains where they are tight for their corresponding case. The functions g_1 and g_2 (blue) are general lower bounds.

References

1 Norbert Ascheuer, Sven Oliver Krumke, and Jörg Rambau. Online Dial-a-Ride Problems: Minimizing the Completion Time. In *Proceedings of the 17th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 639–650, 2000.

2 Mikhail J. Atallah and S. Rao Kosaraju. Efficient Solutions to Some Transportation Problems with Applications to Minimizing Robot Arm Travel. *SIAM Journal on Computing*, 17(5), 1988.

3 G. Ausiello, E. Feuerstein, S. Leonardi, L. Stougie, and M. Talamo. Algorithms for the On-Line Travelling Salesman. *Algorithmica*, 29(4):560–581, 2001.

4 A. Birx and Y. Disser. Tight analysis of the Smartstart algorithm for online Dial-a-Ride on the line. In *Proceedings of the 36th International Symposium on Theoretical Aspects of Computer Science (STACS)*, 2019. Full version: https://arxiv.org/abs/1901.04272.

5 Antje Bjelde, Yann Disser, Jan Hackfeld, Christoph Hansknecht, Maarten Lipmann, Julie Meißner, Kevin Schewior, Miriam Schlöter, and Leen Stougie. Tight Bounds for Online TSP on the Line. In *Proceedings of the 28th Annual Symposium on Discrete Algorithms (SODA)*, pages 994–1005, 2017.

6 Michiel Blom, Sven O. Krumke, Willem E. de Paepe, and Leen Stougie. The Online TSP Against Fair Adversaries. *INFORMS Journal on Computing*, 13(2):138–148, 2001.

7 Moses Charikar and Balaji Raghavachari. The Finite Capacity Dial-A-Ride Problem. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 458–467, 1998.

8 Willem E. de Paepe, Jan Karel Lenstra, Jiri Sgall, René A. Sitters, and Leen Stougie. Computer-Aided Complexity Classification of Dial-a-Ride Problems. *INFORMS Journal on Computing*, 16(2):120–132, 2004.

9 Esteban Feuerstein and Leen Stougie. On-line Single-server Dial-a-ride Problems. *Theoretical Computer Science*, 268(1):91–105, 2001.

10 Paul C. Gilmore and Ralph E. Gomory. Sequencing a One State-Variable Machine: A Solvable Case of the Traveling Salesman Problem. *Operations Research*, 12(5):655–679, 1964.
11 D. J. Guan. Routing a Vehicle of Capacity Greater Than One. *Discrete Applied Mathematics*, 81(1-3):41–57, 1998.
12 Dietrich Hauptmeier, Sven Oliver Krumke, and Jörg Rambau. The Online Dial-a-Ride Problem Under Reasonable Load. In *Proceedings of the 4th Italian Conference on Algorithms and Complexity (CIAC)*, pages 125–136, 2000.
13 Patrick Jaillet and Michael R. Wagner. Generalized Online Routing: New Competitive Ratios, Resource Augmentation, and Asymptotic Analyses. *Operations Research*, 56(3):745–757, 2008.
14 Sven O. Krumke. Online Optimization Competitive Analysis and Beyond, 2001. Habilitation thesis.
15 Sven O. Krumke, Willem E. de Paepe, Diana Poensgen, Maarten Lipmann, Alberto Marchetti-Spaccamela, and Leen Stougie. On Minimizing the Maximum Flow Time in the Online Dial-a-ride Problem. In *Proceedings of the Third International Conference on Approximation and Online Algorithms (WAOA)*, pages 258–269, 2006.
16 Sven O. Krumke, Luigi Laura, Maarten Lipmann, Alberto Marchetti-Spaccamela, Willem de Paepe, Diana Poensgen, and Leen Stougie. Non-abusiveness Helps: An O(1)-Competitive Algorithm for Minimizing the Maximum Flow Time in the Online Traveling Salesman Problem. In *Proceedings of the 5th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, pages 200–214, 2002.
17 Maarten Lipmann, Xiwen Lu, Willem E. de Paepe, Rene A. Sitters, and Leen Stougie. On-Line Dial-a-Ride Problems Under a Restricted Information Model. *Algorithmica*, 40(4):319–329, 2004.
18 Fanglei Yi and Lei Tian. On the Online Dial-a-ride Problem with Time-windows. In *Proceedings of the 1st International Conference on Algorithmic Applications in Management (AAIM)*, pages 85–94, 2005.
19 Fanglei Yi, Yinfeng Xu, and Chunlin Xin. Online Dial-a-ride Problem with Time-windows Under a Restricted Information Model. In *Proceedings of the 2nd International Conference on Algorithmic Aspects in Information and Management (AAIM)*, pages 22–31, 2006.

A Proof of Lemma 2.3

In this section we prove Lemma 2.3. The proof is almost identical to the proof of [5, Lemma 6]. Since there are however several parts where inequalities change slightly, we decided to present the full proof here.

▶ **Lemma 2.3.** *If there is a request sequence with two critical requests for* Alg*, we can release additional requests such that* Alg *is not* $(\rho - \varepsilon)$*-competitive on the resulting instance for any* $\varepsilon > 0$*.*

Let the requests σ^L and σ^R be critical. Furthermore, let $p_0 \in \{t^L, t^R\}$ be the starting position of the request $\sigma_0 \in \{\sigma^L, \sigma^R\}$ that is served first by Alg and let $p_1 \in \{t^L, t^R\}$ be the starting position of the request $\sigma_1 \in \{\sigma^L, \sigma^R\}$ that is not served first by Alg. By properties (iii) and (iv) of Definition 2.2, Alg cannot serve σ_0 before time $(2\rho-2)|p_1| + (\rho-2)|p_0|$. Thus, we have

$$\textsc{Alg}(\sigma_\rho) \geq (2\rho-2)p_1 + (\rho-2)p_0 + |p_0 - p_1| = (2\rho-1)|p_1| + (\rho-1)|p_0|. \tag{32}$$

We have equality in inequality (32) if Alg serves σ_0 the earliest possible time and then moves directly to position p_1. However, in general Alg does not need to do this and instead can wait. At time $t \geq \max\{|p_0|, |p_1|\}$, we have $\textsc{Alg}(\sigma_\rho) \geq t + |\mathrm{pos}\,(t) - p_0| + |p_0 - p_1|$ if Alg still has to serve σ_0 and $\textsc{Alg}(\sigma_\rho) \geq t + |\mathrm{pos}\,(t) - p_1|$ if σ_0 is served and only σ_1 is left to be served. We want to measure the delay of ALG at a time $t \geq \max\{|p_0|, |p_1|\}$, i.e. the difference between the time ALG needs at least to serve both requests σ_0 and σ_1 and the time $(2\rho - 1)|p_1| + (\rho - 1)|p_0|$. We define for $t \geq \max\{|p_0|, |p_1|\}$ the function

$$\text{delay}(t) := \begin{cases} t + |\text{pos}\,(t) - p_0| - (\rho - 2)|p_0| - (2\rho - 2)|p_1| & \text{if } \sigma_0 \text{ is not served at } t, \\ t + |\text{pos}\,(t) - p_1| - (\rho - 1)|p_0| - (2\rho - 1)|p_1| & \text{if } \sigma_0 \text{ is served at } t, \text{ but } \sigma_1 \text{ not}, \\ \text{undefined} & \text{otherwise.} \end{cases}$$

We make the following observation about delay.

▶ **Observation A.1.** *Let $t \geq \max\{|p_0|, |p_1|\}$ be a time at which σ_1 is not served yet. The earliest time* ALG *can serve σ_1 is $(2\rho - 1)|p_1| + (\rho - 1)|p_0| + delay(t)$.*

▶ **Lemma A.2.** *There is a $W \geq 0$ with*

$$delay\left(2|p_1| + |p_0| + \frac{W}{\rho - 1}\right) = W$$

Proof. Because of property (ii) of Definition 2.2, at time $\max\{|p_0|, |p_1|\}$ neither σ_0 nor σ_1 has been served by ALG yet. Since ALG serves σ_1 after σ_0, the request σ_1 is not served before time $\max\{|p_0|, |p_1|\} + |p_0| + |p_1| \geq 2|p_1| + |p_0|$, i.e, $\text{delay}(2p_1 + p_0)$ is defined. Because of properties (iii) and (iv) of Definition 2.2, σ_0 is not served before time $(2\rho - 2)|p_1| + (\rho - 2)|p_0|$. Thus, for $t \geq (2\rho - 2)p_1 + (\rho - 2)p_0$, we have $\text{delay}(t) \geq 0$. We have

$$\begin{aligned} 2p_1 + p_0 &\overset{\text{Def 2.2 (v)}}{\geq} 2p_1 + (3 - \rho)\frac{-8\rho^2 + 50\rho - 66}{4\rho^2 - 30\rho + 50}|p_1| + (\rho - 2)|p_0| \\ &\overset{2 < \rho < 2.5}{>} (2\rho - 2)|p_1| + (\rho - 2)|p_0|, \end{aligned} \tag{33}$$

i.e. $\text{delay}(2p_1 + p_0) \geq 0$. If $\text{delay}(2p_1 + p_0) = 0$, we have $W = 0$ and are done. Otherwise, by inequality (33), we have $\text{delay}(2p_1 + p_0) > 0$. Note that ALG needs to serve σ_1 at some point to be $(\rho - \varepsilon)$-competitive. Let W^* be chosen such that ALG serves σ_1 at time $2|p_1| + |p_0| + \frac{W^*}{\rho - 1}$. Therefore $\text{delay}(2|p_1| + |p_0| + \frac{W^*}{\rho - 1} - \varepsilon')$ is defined for some sufficiently small $\varepsilon' \leq |p_1|$. We define the function

$$f(W) := \text{delay}\left(2|p_1| + |p_0| + \frac{W}{\rho - 1}\right) - W.$$

Note that f is continuous and we have $f(0) > 0$. If

$$\text{delay}\left(2|p_1| + |p_0| + \frac{W^*}{\rho - 1} - \varepsilon'\right) \leq \frac{W^*}{\rho - 1} - \varepsilon' \overset{\rho > 1}{<} W^* - (\rho - 1)\varepsilon',$$

we have $f(W^* - (\rho - 1)\varepsilon') < 0$ and we find W in the interval $(0, W^* - (\rho - 1)\varepsilon']$. Otherwise, we have

$$\text{delay}\left(2|p_1| + |p_0| + \frac{W^*}{\rho - 1} - \varepsilon'\right) > \frac{W^*}{\rho - 1} - \varepsilon'.$$

By Observation A.1 ALG has not served σ_1 at time

$$(2\rho - 1)|p_1| + (\rho - 1)|p_0| + \frac{W^*}{\rho - 1} - \varepsilon' \overset{\rho > 2, \varepsilon' \leq |p_1|}{>} 2|p_1| + |p_0| + \frac{W^*}{\rho - 1}.$$

This is a contradiction to the fact, that W^* was chosen such that ALG serves σ_1 at time $2|p_1| + |p_0| + \frac{W^*}{\rho - 1}$. ◀

▶ **Lemma A.3.** *Let* $W \geq 0$ *with*

$$\textit{delay}\left(2|p_1| + |p_0| + \frac{W}{\rho - 1}\right) = W.$$

Alg *serves* σ_0 *no later than time* $2|p_1| + |p_0| + \frac{W}{\rho-1}$.

Proof. Assume we have

$$2|p_1| + |p_0| + \frac{W}{\rho - 1} \geq (2\rho - 2)|p_1| + (\rho - 2)|p_0| + W. \tag{34}$$

Then, by definition of W and Observation A.1, Alg can serve σ_1 at time

$$(2\rho-1)|p_1|+(\rho-1)|p_0|+\text{delay}\left(2|p_1| + |p_0| + \frac{W}{\rho - 1}\right) = (2\rho-1)|p_1|+(\rho-1)|p_0|+W. \tag{35}$$

Because of inequality (34), this can only be the case if Alg serves σ_0 no later than time

$$(2\rho-1)|p_1|+(\rho-1)|p_0|+W-|p_1|-|p_0| = (2\rho-2)|p_1|+(\rho-2)|p_0|+W \overset{(34)}{\leq} 2|p_1|+|p_0|+\frac{W}{\rho - 1}.$$

Thus, it remains to show inequality (34). Because of property (i) of Definition 2.2 all requests can be served the tours $\text{move}(p_0) \oplus \text{move}(p_1)$ and $\text{move}(p_1) \oplus \text{move}(p_0)$. By inequality 35, we have $\text{Alg}(\sigma_\rho) \geq (2\rho - 1)|p_1| + (\rho - 1)|p_0| + W$. Thus, if we have

$$\text{Alg}(\sigma_\rho) \geq (2\rho - 1)|p_1| + (\rho - 1)|p_0| + W > (\rho - \varepsilon)(2|p_1| + |p_0|) \geq (\rho - \varepsilon)\text{Opt}(\sigma_\rho),$$

Alg is not $(\rho - \varepsilon)$-competitive. Therefore, we may assume

$$(2\rho - 1)|p_1| + (\rho - 1)|p_0| + W \leq (\rho - \varepsilon)(2|p_1| + |p_0|),$$

and thus

$$\begin{aligned} W &\leq (\rho - \varepsilon)(2|p_1| + |p_0|) - (2\rho - 1)|p_1| - (\rho - 1)|p_0| \\ &= (1 - 2\varepsilon)|p_1| + (1 - \varepsilon)|p_0| \\ &< |p_1| + |p_0|. \end{aligned} \tag{36}$$

Inequality (34) now is equivalent to the inequality

$$\begin{aligned} \frac{2|p_1| + |p_0| - ((2\rho - 2)|p_1| + (\rho - 2)|p_0|)}{1 - \frac{1}{\rho-1}} &= \frac{(\rho-1)(4-2\rho)}{\rho - 2}|p_1| + \frac{(\rho-1)(3-\rho)}{\rho-2}|p_0| \\ &\overset{\text{Def 2.2 (v)}}{\geq} |p_0| + (2 - 2\rho)|p_1| \\ &\quad + \frac{(-\rho^2 + 3\rho - 1)(-8\rho^2 + 50\rho - 66)}{(\rho - 2)(4\rho^2 - 30\rho + 50)}|p_1| \\ &\geq |p_0| + \frac{5\rho^3 - 36\rho^2 + 86\rho - 67}{2\rho^3 - 19\rho^2 + 55\rho - 50}|p_1| \\ &\overset{2<\rho<2.5}{>} |p_0| + |p_1| \\ &\overset{(36)}{>} W \end{aligned}$$

if we solve inequality (34) for W. ◀

Now we have all ingredients to proof Lemma 2.3.

Proof of Lemma 2.3. Let $W \geq 0$ with delay$(2|p_1| + |p_0| + \frac{W}{\rho-1}) = W$. We present the request

$$\sigma_0^+ = (p_0^+, p_0^+; t_0^+) := \left(p_0 + \operatorname{sgn}(p_0)\frac{W}{\rho-1}, p_0 + \operatorname{sgn}(p_0)\frac{W}{\rho-1}; 2|p_1| + |p_0| + \frac{W}{\rho-1}\right)$$

and distinguish two cases.

Case 1: At time t_0^+, ALG is at least as close to p_1 as to p_0^+ or it serves σ_1 before σ_0^+. In this case, we do not present additional requests. By Lemma A.3, ALG has served σ_0 at time t_0^+ or before and by Observation A.1 it does not serve σ_1 earlier than time $(2\rho-1)|p_1| + (\rho-1)|p_0| + W$. Thus, we have

$$\begin{aligned}\text{ALG}(\sigma_\rho) &\geq (2\rho-1)|p_1| + (\rho-1)|p_0| + W + |p_1| + |p_0| + \frac{W}{\rho-1}\\ &\geq \rho\left(2|p_1| + |p_0| + \frac{W}{\rho-1}\right)\\ &= \rho\text{OPT}(\sigma_\rho).\end{aligned}$$

Case 2: At time t_0^+, ALG is closer to p_0^+ than to p_1 and it serves σ_0^+ first. We assume that the offline server continues moving away from the origin after serving σ_0^+ at time p_0^+. Then, the position of the offline serve at time $t \geq |p_1|$ is $\operatorname{sgn}(p_0)t + 2p_1$. We denote by

$$M(t) := \frac{\operatorname{sgn}(p_0)t + 3p_1}{2}$$

the midpoint between the current position of the offline server and the position p_1. Note that the time $M^{-1}(p)$, when the midpoint is at position p is given by

$$M^{-1}(p) := |2p - 3p_1|.$$

We again distinguish two cases

Case 2.1: ALG does not serve σ_0^+ until time $M^{-1}(p_0^+)$. In this case, we do not present additional requests. Since we are in Case 2, neither σ_0^+ nor σ_1 is served at time $M^{-1}(p_0^+)$. Thus, we have

$$\begin{aligned}\text{ALG}(\sigma_\rho) &\geq M^{-1}(p_0^+) + |p_0^+| + |p_1|\\ &= |2p_0^+ - 3p_1| + |p_0^+| + |p_1|\\ &= |2p_0 + 2\operatorname{sgn}(p_0)\frac{W}{\rho-1} - 3p_1| + |p_0| + \frac{W}{\rho-1} + |p_1|\\ &= 3|p_0| + 4|p_1| + 3\frac{W}{\rho-1}\\ &\overset{2<\rho<2.5}{>} \rho|p_0| + 2\rho|p_1| + 3\frac{W}{\rho-1}\\ &> \rho\left(|p_0| + 2|p_1| + \frac{W}{\rho-1}\right)\\ &= \rho\text{OPT}(\sigma_\rho).\end{aligned}$$

Case 2.2: $\textsc{Alg}$ serves σ_0^+ before time $M^{-1}(p_0^+)$. By definition of W, the delay function is defined for time p_0^+, hence $\textsc{Alg}$ has not served σ_1 before time p_0^+. Since $\textsc{Alg}$ is to the right of the midpoint $M(p_0^+)$ at time p_0^+, there is a first time t_{mid} at which $M(t_{\text{mid}}) = \text{pos}\,(t_{\text{mid}})$. We present the request

$$\sigma_0^{++} = (p_0^{++}, p_0^{++}; t_0^{++}) := (\text{sgn}(p_0)t_{\text{mid}} + 2p_1, \text{sgn}(p_0)t_{\text{mid}} + 2p_1; t_{\text{mid}}).$$

Note that $\textsc{Alg}$ is at the midpoint between p_0^{++} and p_1 and thus, both tours $\text{move}(p_0^{++}) \oplus \text{move}(p_1)$ and $\text{move}(p_1) \oplus \text{move}(p_0^{++})$ incur identical costs for $\textsc{Alg}$. We have

$$\textsc{Alg}(\sigma_\rho) \geq t_{\text{mid}} + 3\left(\frac{|\text{sgn}(p_0)t_{\text{mid}} + 2p_1 - p_1|}{2}\right) = \frac{5t_{\text{mid}} + 3|p_1|}{2}$$

We have $\textsc{Opt}(\sigma_\rho) = t_{\text{mid}}$, i.e., if we want to show

$$\textsc{Alg}(\sigma_\rho) \geq \frac{5t_{\text{mid}} + 3|p_1|}{2} \geq \rho t_{\text{mid}} = \rho\textsc{Opt}(\sigma_\rho) \tag{37}$$

Inequality (37) is equivalent to

$$(5 - 2\rho)t_{\text{mid}} \geq 3|p_1|. \tag{38}$$

Since $2\rho < 2.5$, the coefficient $(5 - 2\rho)$ of t_{mid} is positive. Thus we may assume t_{mid} is minimal to show the inequality (38). By assumption, σ_0^+ is already served at time t_{mid}. Hence, t_{mid} is minimum if, starting at time t_0^+ at position $\text{pos}\left(t_0^+\right)$, $\textsc{Alg}$ serves σ_0^+ and then moves towards the origin. Then, t_{mid} is the solution of the equation

$$\text{sgn}(p_0)t_0^+ + |\text{pos}\left(t_0^+\right) - p_0^+| + p_0^+ - \text{sgn}(p_0)t_{\text{mid}} = \frac{\text{sgn}(p_0)t_{\text{mid}} + 3p_1}{2}. \tag{39}$$

Because of Lemma A.3, the request σ_0 is already served at time t_0^+. Furthermore, since the position of σ_1 has not been visited yet at time t_0^+, we have $\text{sgn}(p_0)\text{pos}\left(t_0^+\right) > \text{sgn}(p_0)p_1$, i.e.,

$$|\text{pos}\left(t_0^+\right) - p_1| = \text{sgn}(p_0)(\text{pos}\left(t_0^+\right) - p_1) > 0$$

and thus, because of $-\text{sgn}(p_0)p_1 = |p_1|$, we get

$$\begin{aligned}
\text{delay}(t_0^+) &= t_0^+ + |\text{pos}\left(t_0^+\right) - p_1| - (\rho - 1)|p_0| - (2\rho - 1)|p_1| \\
&= t_0^+ + \text{sgn}(p_0)\text{pos}\left(t_0^+\right) - \text{sgn}(p_0)p_1 - (\rho - 1)|p_0| - (2\rho - 1)|p_1| \\
&= t_0^+ + \text{sgn}(p_0)\text{pos}\left(t_0^+\right) + |p_1| - (\rho - 1)|p_0| - (2\rho - 1)|p_1|. \qquad (40)
\end{aligned}$$

Solving equation (40) for $\text{sgn}(p_0)\text{pos}\left(t_0^+\right)$ gives

$$\begin{aligned}
\text{sgn}(p_0)\text{pos}\left(t_0^+\right) \quad &= \quad \text{delay}\left(2|p_1| + |p_0| + \frac{W}{\rho - 1}\right) - \frac{W}{\rho - 1} \\
&\quad\quad +(\rho - 2)|p_0| + (2\rho - 4)|p_1| \\
&= \quad W - \frac{W}{\rho - 1} + (\rho - 2)|p_0| + (2\rho - 4)|p_1| \\
&= \quad \frac{\rho - 2}{\rho - 1}W + (\rho - 2)|p_0| + (2\rho - 4)|p_1| \qquad (41) \\
&\overset{\rho < 3}{<} \quad \frac{W}{\rho - 1} + (\rho - 2)|p_0| + (2\rho - 4)|p_1| \\
&\overset{\text{Def 2.2 (v)}}{\leq} \quad \frac{W}{\rho - 1} + \left((\rho - 2) + (2\rho - 4)\frac{4\rho^2 - 30\rho + 50}{-8\rho^2 + 50\rho - 66}\right)|p_0| \\
&\overset{1.9 < \rho < 4.3}{<} \quad \frac{W}{\rho - 1} + |p_0| \\
&\overset{\text{sgn}(p_0) = \text{sgn}(p_0^+)}{=} \quad \text{sgn}(p_0)p_0^+.
\end{aligned}$$

Thus, we have

$$|\mathrm{pos}\left(t_0^+\right) - p_0^+| = \mathrm{sgn}(p_0)(p_0^+ - \mathrm{pos}\left(t_0^+\right)) > 0 \tag{42}$$

Using inequality (42) and plugging inequality (41) into inequality (39) gives us

$$\begin{aligned}
\mathrm{sgn}(p_0)t_{\mathrm{mid}} &= \frac{1}{3}(2\mathrm{sgn}(p_0)t_0^+ + 2|\mathrm{pos}\left(t_0^+\right) - 2p_0^+| + 2p_0^+ - 3p_1) \\
&\overset{(42)}{=} \frac{1}{3}(2\mathrm{sgn}(p_0)t_0^+ + 2\mathrm{sgn}(p_0)p_0^+ - 2\mathrm{sgn}(p_0)\mathrm{pos}\left(t_0^+\right) + 2p_0^+ - 3p_1) \\
&= \frac{1}{3}\left(-7p_1 + 6p_0 + \frac{(6\mathrm{sgn}(p_0))W}{\rho - 1} - 2\mathrm{sgn}(p_0)\mathrm{pos}\left(t_0^+\right)\right) \\
&\overset{(41)}{=} \frac{1}{3}\left(-(15 - 4\rho)p_1 + (10 - 2\rho)p_0 + \frac{(10 - 2\rho)\mathrm{sgn}(p_0)W}{\rho - 1}\right)
\end{aligned} \tag{43}$$

Note that we also used $\mathrm{sgn}(p_0) = \mathrm{sgn}(p_0^+) = -\mathrm{sgn}(p_1)$. Multiplying equality (43) with $\mathrm{sgn}(p_0)$ gives us

$$t_{\mathrm{mid}} = \frac{1}{3}\left((15 - 4\rho)|p_1| + (10 - 2\rho)|p_0| + \frac{(10 - 2\rho)W}{\rho - 1}\right). \tag{44}$$

By substituting (44) into (38) and noting that it is hardest to satisfy, when $W = 0$, we get

$$\frac{|p_0|}{|p_1|} \leq \frac{4\rho^2 - 30\rho + 50}{-8\rho^2 + 50\rho - 66},$$

which is true due to Definition 2.2 (v). ◀