On the Power of Temporal Locality on Online Routing Problems

Swapnil Guragain
Kent State University
Kent, Ohio, USA
sguragai@kent.edu

Gokarna Sharma
Kent State University
Kent, Ohio, USA
gsharma2@kent.edu

ABSTRACT

We consider the online variants of two fundamental routing problems, traveling salesman (TSP) and dial-a-ride (DRP), which have a variety of relevant applications in logistics and robotics. These problems concern with efficiently serving a sequence of requests presented in an on-line fashion located at points of a metric space by servers (salesmen/repairmen/vehicles/robots). In this paper, we propose the *temporal locality* model that provides in advance the time interval between the release of subsequent request(s). We study the usefulness of this advanced information on achieving the improved competitive ratios for both the problems with $k \geq 1$ servers. We show the surprising impact: shorter locality is useful for arbitrary metric but for line metric larger locality.

KEYWORDS

Online algorithms; traveling salesman; dial-a-ride; makespan; arbitrary and line metric; competitive analysis; temporal locality

ACM Reference Format:
Swapnil Guragain and Gokarna Sharma. 2025. On the Power of Temporal Locality on Online Routing Problems. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 11 pages.

1 INTRODUCTION

We consider the *online* variants of two fundamental problems, *traveling salesman* (Tsp) and *dial-a-ride* (Darp), which we denote as oTsp and oDarp. We solve these problems in line and arbitrary metric that satisfy triangle inequality. These problems concern with efficiently serving a sequence of requests presented in an on-line fashion located at points of a metric space. These problems have a variety of relevant applications in logistics and robotics. Consider for an example a set of salesmen/repairmen/vehicles/robots that have to serve locations on its workspace (e.g., Euclidean plane) and of many other routing and scheduling problems on a transportation network modeled with a graph. In this paper, we refer to as *servers* the salesmen/repairmen/vehicles/robots. As the input to the servers is communicated in an online fashion, the scheduled routes will have to be updated also in an online fashion during the trips of the servers. Since online execution is unaware of future requests, designing an optimal schedule is generally not possible.

In the literature, the *offline* variants were studied heavily for both the problems. Since complete input is known beforehand, the offline schedules are relatively easier to design. However, the

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), A. El Fallah Seghrouchni, Y. Vorobeychik, S. Das, A. Nowe (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

Model	Characteristic
Original	Requests arriving at time t_i are known at t_i
Lookahead [1]	Requests arriving upto t_i are known at $t_i - a$
Disclosure [11]	Requests arriving upto $t_i + a$ are known at t_i
Temporal locality (this paper)	Subsequent request arrival interval Δ

Table 1: Comparing online routing models.

offline variants cannot capture the real-world situation in which the complete input may not be available to the algorithm a priori. An online algorithm makes a decision on what to do based on the input (or requests) known so far without knowledge on future requests. A standard technique to evaluate the quality of a solution provided by an online algorithm is *competitive analysis*: An online algorithm OL is c-competitive if for any request sequence σ it holds that the cost of the online algorithm $OL(\sigma) \leq c \cdot OPT(\sigma)$, where OPT is an optimal *offline* algorithm for σ knowing σ completely a priori.

In oDarp, there are m ride requests between points of the metric space arriving over time. Each ride request consists of the corresponding source and destination points. A ride request is *served* if a serverfi rst reaches to the source and then to the destination. There are $k \geq 1$ servers initially positioned at a *distinguished origin*, a point in the metric space. Each server travels with speed at most one, meaning that it traverses unit distance in unit time if on maximum speed. We consider the *uncapacitated* variant of oDarp meaning that a server can serve simultaneously all the requests (source and destination pairs) enroute. This is in contrast to the *capacitated* variant in which there is a limit on how many requests a server can serve simultaneously at any time.

oTsp is similar to oDarp with the only difference that the source and destination points of requests coincide, i.e., reaching to source point serves the request. In certain situations, the server(s) need to return to the distinguished origin after serving all the requests. If such a requirement, we refer as *homing* (or closed), *nomadic* (or open) otherwise.

The execution starts at time $t = 0$. Suppose a request r_i arrives (releases) at time $t_i \geq 0$ and served by a server at time $t'_i \geq t_i$. Time t'_i denotes the *completion time* of r_i. The goal is to minimize the *maximum* completion time defined as $\max_{i=1}^{m} t'_i$ for all m requests in σ (note that m is not known a priori).

In the *original* (online) model, it is assumed that a request r_i arrived (released) at time t_i is only known at t_i. Therefore, the completion time t'_i for r_i cannot be smaller than t_i, i.e., $t'_i \geq t_i$. Notice that if a request arrived at t_i can be served at $t'_i = t_i$, then that algorithm would be best possible (i.e., 1-competitive). For nomadic oTsp, the best previously known results obtained lower bound of 2.04 [7] and upper bound of $1 + \sqrt{2} = 2.41$ [13] in arbitrary metric. For nomadic oDarp, the lower bound is 2.0585 and upper bound is 2.457 [4]. The following question naturally arises: *Can providing an algorithm with limited clairvoyance, i.e., the capability to foresee some limited future, help in achieving a better competitive ratio?* In other words, is limited clairvoyance helpful in reducing the gap between competitive ratio upper and lower bounds?

The clairvoyance can come in many forms and they might have exhibit different power on how much they could help in obtaining the better competitive ratios. In the literature, Jaillet and Wagner [11] proposed the *disclosure* model in which request r_i with release time t_i is known to the online algorithm OL at time $t_{ia} = t_i - a$ for some constant $a > 0$. That is, r_i becomes known to OL before its release time t_i, that is, at t_{ia} OL knows r_i will be released at time t_i. Knowing at $t_{ia} < t_i$ may help OL to plan such that r_i can be served in time $t'_i \geq t_i$ that is not much larger than t_i, possibly improving the competitive ratios. Allulli *et al.* [1] proposed the *lookahead* model in which OL knows at any time $t \geq 0$ all the requests with release time $t + \Theta$ (with Θ the lookahead time). Despite naming differently, these models resemble the same concept of making OL aware of some of the future requests before their release times.

In this paper, we focus on limited clairvoyance as in the line of disclosure and lookahead models but propose another form that is sufficiently different from these models (Table 1). In particular, we propose the *temporal locality* model in which the online algorithm OL has the advanced information about the time interval $\Delta \geq 0$ between the arrival of consecutive requests, i.e., if a (set of) request(s) is released at time t, the next (set of) request(s) will be released at time $t + \Delta$. The temporal locality Δ is defined with respect to the diameter D of the metric, meaning that $\Delta = \delta D$ with $\delta \geq 0$. In many applications, such as sensor data collection, there is afi xed time interval on when the (set of) sensors have data ready to be collected. Providing this time interval information to, say the data collecting robot, may help in improving the maximum data completion time.

We propose new online algorithms and derive improved competitive ratios, which are functions of the advance information (that is, temporal locality Δ), beating the existing algorithms in the original model (with no such advance information). We consider explicitly the case of "fixed amount" of temporal locality, i.e., the temporal locality between requests is the same amount $\Delta \geq 0$.

Note that, following the literature, we do not focus on runtime complexity of our algorithms as they may need non-polynomial time to compute a solution. Instead, we focus on the quality of the online algorithms (the total time to serve all the requests) on achieving competitive ratios (without complete knowledge on the requests) close to the competitive ratios achieved by offline algorithms with complete knowledge on the requests.

Contributions. Table 2 lists our contributions and compares them with the previous results. We have following four contributions.

i. Wefi rst prove a lower bound, i.e., no deterministic algorithm can achieve better than 2-competitive ratio for both nomadic and homing oTsp (which applies directly to both nomadic and homing oDarp) on arbitrary metric, independently of the amount of temporal locality $\Delta = \delta D$. (**Section 3**)

ii. We prove $\min\{1 + \sqrt{2}, 2 + \delta)$-competitive ratio for nomadic oTsp on arbitrary metric. The best previously known bound is $1 + \sqrt{2}$ in the original model and the temporal locality $\Delta = \delta D$ with $\delta < \sqrt{2} - 1$ provides the improved competitive ratio. With $\Delta << D$, the competitive ratio almost matches the lower bound of 2. For homing oTsp, there exists a 2-competitive algorithm in the original model which directly provides 2-competitive ratio in the temporal locality model. We then consider the line metric. Let $\beta = \min\{1, \frac{t_{max}}{\Delta}\}$, where t_{max} is the maximum arrival time among requests in σ. For nomadic oTsp, we prove $\min\{2.04, 1 + \frac{3}{2\beta\delta}\}$-competitive ratio. The significance of this result is that the ratio never exceeds 2.04 and gets arbitrarily closer to 1 (which is optimal) whenever $t_{max} > \Delta$ and $\Delta > 1.442D$. For homing oTsp, we prove $\min\{2, 1 + \frac{2}{\max\{2,\beta\delta\}}\}$-competitive ratio. (**Section 4**)

iii. For nomadic oDarp on arbitrary metric, we prove the competitive ratio of $\min\{2.457, 2 + \delta\}$. For homing oDarp, there exists a 2-competitive algorithm in the original model which directly provides 2-competitive ratio in the temporal locality model. On line metric, for nomadic oDarp, we prove $\min\{4, 1 + \frac{3}{\beta\delta}\}$-competitive ratio, and for homing oDarp, the competitive ratio of $\min\{3, 1 + \frac{4}{\max\{2,\beta\delta\}}\}$. (**Section 5**)

iv. Finally, we consider $k > 1$ servers (notice the lower bound of 2 also applies to $k > 1$ servers) and establish competitive ratios for both nomadic oTsp and oDarp on arbitrary metric. We then consider line metric and establish competitive bounds for both nomadic and homing versions of oTsp and oDarp. (**Section 6**)

The results exhibit surprising impact of temporal locality on the competitive ratios for oTsp and oDarp. On arbitrary metric, shorter temporal locality is beneficial ($\delta < \sqrt{2} - 1 = 0.414$ for oTsp and $\delta < 0.457$ for oDarp), whereas on line metric, longer temporal locality is beneficial (e.g., $\beta\delta > 2$ for nomadic oTsp). In cases of incorrect Δ, δ in our bounds will be replaced by $\delta + \epsilon$, where ϵ is the maximum error on δ. In other words, if subsequent request(s) arrives in the interval of $\delta' \neq \delta$, then $\epsilon' = \delta' - \delta$ and ϵ can be the maximum ϵ'. Finally, our upper bounds are in the form of $\min\{X, Y\}$ with X (Y) being the bound in the original (temporal locality) model. Therefore, even with incorrect δ, our bounds do not go beyond the original model bound X.

Previous Work. Wefi rst discuss literature on oTsp. oTsp was first considered by [3] in which they established tight (competitive ratio) of 2/2 (lower/upper) on arbitrary metric and 1.64/1.75 on line metric for homing oTsp. For nomadic oTsp, they provided the lower bound of 2 on line metric and upper bound of $\frac{5}{2}$ on arbitrary metric. Lipmann [13] improved the upper bound to $1+\sqrt{2}$ for nomadic oTsp on arbitrary metric. On line metric, Bjelde *et al.* [7] provided tight bound of 2.04/2.04 for nomadic oTsp and, for homing oTsp, they improved the upper bound to 1.64 matching the lower bound. In the lookahead model, Allulli *et al.* [1] provided an upper bound of $1 + \frac{2}{\alpha}$ for both nomadic and homing oTsp on line metric and lower/upper bounds of $2/\max\{2, 1 + \frac{1}{2}(\sqrt{\alpha^2 + 8} - \alpha)\}$ and 2/2 for nomadic and homing oTsp, respectively, on arbitrary metric. In the disclosure model, Jaillet and Wagner [11] provided an upper bound of $(2-\frac{\rho}{1+\rho})$ for homing oTsp on arbitrary metric. Considering multiple servers ($k > 1$), Bonifaci and Stougie [8] provided lower/upper bounds of $1 + \Omega(\frac{1}{k})/1 + O(\frac{\log k}{k})$ and $2/(1 + \sqrt{2})$ for nomadic oTsp on line and arbitrary metric, respectively. In this paper, we provide, for both nomadic and homing otsp on both line and arbitrary

Algorithm	NoTspHoT (lower/upper)	spNoD (lower/upper)	arpHoDarp (lower/upper)	(lower/upper)	Metric
			Single server		
Original	2.04/2.04 [7]	1.64/1.64 [3, 7]	2.0585/2.457 [4, 6]	2/2 [2, 9]	line
Original	2.04/2.41 [7, 13]	2/2 [3]	2.0585/2.457 [4, 6]	2/2 [2, 9]	arbitrary
Lookahead [1]	-/$(1+\frac{2}{\alpha})$	-/$(1+\frac{2}{\alpha})$	-/-	-/-	line
Lookahead [1]	2/$\max\{2, 1+\frac{1}{2}(\sqrt{\alpha^2+8}-\alpha)\}$	2/2	-/-	-/-	arbitrary
Disclosure [11]	-/-	-/$(2-\frac{\rho}{1+\rho})$	-/-	-/-	arbitrary
Temporal locality	-/$\min\{2.04, 1+\frac{3}{2\beta\delta}\}$	-/$\min\{2, 1+\frac{2}{\max\{2,\beta\delta\}}\}$	-/$\min\{4, 1+\frac{3}{\beta\delta}\}$	-/$\min\{3, 1+\frac{4}{\max\{2,\beta\delta\}}\}$	**line**
Temporal locality	2/$\min\{2.41, 2+\delta\}$	2/2	2/$\min\{2.457, 2+\delta\}$	2/2	**arbitrary**
			Multiple servers		
Original [8]	$1+\Omega(\frac{1}{k})/1+O(\frac{\log k}{k})$	-/-	-/-	-/-	line
Original [8]	2/2.41	-/-	-/-	-/-	arbitrary
Temporal locality	-/$\min\{2.04, 1+\frac{1}{\beta\delta}\}$	-/$\min\{2, 1+\frac{2}{\max\{2,\beta\delta\}}\}$	-/$\min\{4, 1+\frac{3}{2\beta\delta}\}$	-/$\min\{3, 1+\frac{2}{\beta\delta}\}$	**line**
Temporal locality	2/$\min\{2.41, 2+\delta\min\{\gamma,1\}\}$	2/2	2/$\min\{2.457, 2+\delta\min\{\gamma,1\}\}$	2/2	**arbitrary**

Table 2: A summary of previous and proposed results for both nomadic and homing versions of oTsp and oDarp (uncapacitated) for $k \geq 1$ servers. The notion 'X/Y' denotes X as a lower bound and Y as an upper bound in the competitive ratio. We have $\rho = \frac{a}{|\mathcal{T}|}$, $\alpha = \frac{\Theta}{D}$, $\delta = \frac{\Delta}{D}$, $\beta = \min\{1, \frac{t_{max}}{\Delta}\}$, and $\gamma = \frac{\max_{1\leq j\leq k}|\mathcal{T}_j|}{D}$ with a being the disclosure time, $\mathcal{T}$ being the TSP tour of the requests in σ, $\mathcal{T}_j$ being the TSP tour of the j-th server for the requests in σ, Θ being the lookahead time, D being the diameter of the metric space, and t_{max} being the maximum release time among the requests in σ, and Δ being the temporal locality between the consecutive requests in σ. '-' denotes non-existence of the respective lower/upper bound for the respective problem.

metric, lower/upper bound results in the temporal locality model. In arbitrary metric, for the homing oTsp, the 2-competitive bound on the original model applies directly to the temporal locality model.

We now discuss literature of oDarp. For homing oDarp lower/upper bounds of 2/2 exist on both line and arbitrary metric. For nomadic oDarp, the best previously known lower/upper bounds are 2.0585/2.457 [4]. oDarp was not studied before in the k-server setting. It was also not studied in the lookahead and disclosure models. In this paper, we provide, for both nomadic and homing oDarp on both line and arbitrary metric, lower/upper bound results in the temporal locality model for single and multiple servers. In arbitrary metric, for the homing oDarp, the 2-competitive bound on the original model applies directly to the temporal locality model.

A distantly related model is of *prediction* [10] which provides the online algorithm with the predicted locations (which may be erroneous) of requests beforehand. It is a different model since temporal locality (also lookahead and disclosure) focus on time not request location.

2 MODEL

We consider the online model where time is divided into discrete steps. Multiple requests may arrive at a time step and a new request may arrive before the previously released request(s) has been served. We consider a sequence $\sigma = r_1, \ldots, r_m$ of m requests; m is not known beforehand. Every request $r_i = (t_i, e_i, d_i)$ is a triple, where $t_i \geq 0$ is the *release time*, e_i is the source location (point), and d_i is the destination location. In oTsp, e_i and d_i coincide and hence they can be considered as a single point e_i. All the information about r_i: t_i, e_i, d_i, and its existence is revealed only at time t_i. The lookahead and disclosure models [1, 11] extend this model and assume that the request releasing at time t_i is known to server at time $t_i - a$, with $a \geq 0$. Our temporal locality model assumes only the advanced knowledge of how far in time subsequent requests arrive, which we formally define in the following.

Definition 1 (temporal locality). An online algorithm OL has *temporal locality* $\Delta \in \mathbb{N}$, if for any two consecutive requests r_i, r_j with release times t_i, t_j, $t_i \neq t_j$, $|t_i - t_j| = \Delta$.

In our temporal locality model, the additional prior knowledge the server has compared to the original model is the value of Δ, our temporal locality parameter.

We assume that the execution starts at time 0. We consider $k \geq 1$ servers $s_1, \ldots, s_k$, initially positioned at origin o. The servers can move with maximum speed one unit so that in one time step they can travel one unit distance. To *serve* r_i, the server has to visit both the locations e_i, d_i, but not earlier than t_i, and e_i has to be visited before d_i. In other words, visitingfirst d_i and then coming to e_i does not serve r_i. After visiting e_i, it has to visit d_i to serve r_i. In oTsp, since e_i and d_i coincide, visiting e_i serves r_i.

Let $dist(A,B)$ denotes the length of the shortest path between two points A and B. Following the literature [5, 9, 12], we consider metric space $\mathcal{M}$ which satisfies the following properties: (i) *definiteness:* for any point $x \in \mathcal{M}$, $dist(x,x) = 0$, (ii) *symmetry:* for any two points $x,y \in \mathcal{M}$, $dist(x,y) = dist(y,x)$, and (iii) *triangle inequality:* for any three points $x,y,z \in \mathcal{M}$, $dist(x,z) + dist(z,y) \geq dist(x,y)$. $\mathcal{M}$ becomes line metric $\mathcal{L}$ when for any three points $x,y,z \in \mathcal{M}$, $dist(x,z) + dist(z,y) = dist(x,y)$. The server(s), origin o, and requests in σ are all on $\mathcal{M}$.

The *completion time* of request $r_i = (t_i, e_i, d_i) \in \sigma$ is the time $t'_i \geq t_i$ at which r_i has been served (notice that r_i cannot be served before t_i). After r_i is released at time t_i and before it has been served at time t'_i, it remains *outstanding* for the duration $t'_i - t_i$. Given σ, a *feasible schedule* for σ is a sequence of moves of the server(s) such that all the requests in σ are served. $OL(\sigma)$ denotes the maximum completion time of an online algorithm OL for serving the requests in σ i.e., $OL(\sigma) = \max_{1\leq i\leq m} t'_i$. $OPT(\sigma)$ is the completion time of an optimal algorithm for serving the requests in σ. $OPT(\sigma)$ has two components depending on the release times t_i of the requests. If $t_i > 0$ for at least a request in σ, $OPT(\sigma) \geq \max_{1\leq i\leq m} t_i$. If $t_i = 0$ for each request in σ, $OPT(\sigma) \geq |\mathcal{T}|$, where $\mathcal{T}$ is the optimal TSP tour length that connects origin o with the m source (and destination for oDarp) points of the requests in σ. Combining these two bounds, we have $OPT(\sigma) \geq \max\{\max_{1\leq i\leq m} t_i, |\mathcal{T}|\}$.

3 LOWER BOUND

Thefi rst natural direction is to study the impact of temporal locality on solutions to oTsp and oDarp. We study this impact through a lower bound, which holds for both nomadic and homing versions of oTsp (this oTsp lower bound applies directly to oDarp). We prove that no online algorithm for oTsp can be better than 2-competitive in arbitrary metric.

Although we are able to establish the same lower bound for both nomadic and homing versions of oTsp, their implication is substantially different. For homing oTsp, an optimal 2-competitive algorithm exists in the original model [3], i.e., temporal locality has no impact. Instead, for the nomadic oTsp, there exists a lower bound of 2.04 in the original model for line metric [7] (which directly applies to arbitrary metric), whereas in the temporal locality model we could establish the lower bound of 2. Additionally, the best known online algorithm achieves the competitive ratio of $1+\sqrt{2}$ [13]. We will show later that temporal locality is indeed useful for nomadic oTsp; the lower bound of 2 is matched for a sufficiently small value of Δ. Our lower bound is interesting since it holds for any value of Δ, which implies that the larger temporal locality does not help to improve the competitive ratio in arbitrary metric.

Theorem2. *No deterministic algorithm for homing* oTsp *or nomadic* oTsp *can be better than 2-competitive in arbitrary metric, irrespective of temporal locality* Δ.

Proof. Consider a star graph $G = (V, E)$ with $N + 1$ nodes; a central node v_0 and N peripheral nodes $v_1, \ldots, v_N$. Each peripheral node v_i is connected to the central node v_0 by an edge $e_i = (v_0, v_i)$ of length $\frac{1}{2}$ (see Fig. 1). Let OL be any algorithm for homing oTsp or nomadic oTsp on G with temporal locality $N > \Delta > 1$. Consider Δ such that $N \mod \Delta = 0$.

At time $t_0 = \Delta - 1$, N requests are released, one on each peripheral node. At time $t_0 + \Delta$, Δ requests have been served by OL and $N - \Delta$ requests are still waiting to be served. At time $t_0 + \Delta$, Δ new requests are released on same vertices on which the requests were served in the last Δ time steps. Therefore, at time $t_0 + \Delta$, there are again N requests, one on each vertex v_i of G. Continue releasing Δ requests with temporal locality Δ until time $t_0 + N$, i.e., after Δ requests have been served by OL.

Let $t_f = \Delta + N - 1$. At t_f, there are exactly N requests on N peripheral nodes of G waiting to be served. Suppose after t_f, no new request is presented, i.e., in total $2N$ requests have been presented in G. At time t_f, since OL still needs to serve N outstanding requests, it cannotfi nish serving them before time $t_f + N - 1 = \Delta + 2N - 2$.

Consider an offline adversary OPT. We show that OPT can complete serving all $2N$ requests in no later than time $2\Delta + N - 1$.

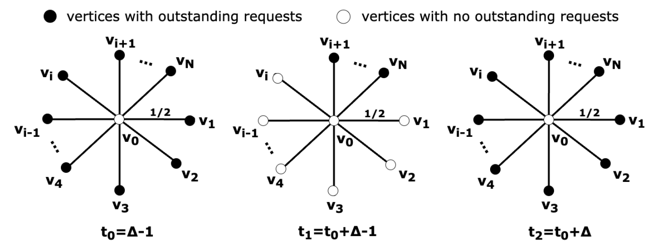

Figure 1: An illustration of the lower bound construction: (*left*) At time $t_0 = \Delta - 1$, N requests are released on N peripheral nodes of G, (*middle*) At $t_0 + \Delta - 1$, Δ requests have been served by the online algorithm OL, (*right*) At $t_0 + \Delta$, Δ new requests have been released on the Δ empty nodes at $t_0 + \Delta - 1$. This process continues until $t_0 + N$ such that in total $2N$ requests have been released.

Consider the vertices of the Δ requests released at time $t_0 + \Delta$. OPT waits until $t_0 + \Delta - 1/2$ at v_0, then reaches thefi rst request at time $t_0 + \Delta$ and serve the 2Δ requests on Δ nodes by time $t_0 + \Delta + \Delta - 1 = t_0 + 2\Delta - 1$. Continuing this way, OPT finishes no later than $t_0 + \Delta + \frac{N}{\Delta}\Delta - 1 = 2\Delta + N - 2$ all $2N$ requests. For homing oTsp, OPT needs additional $\frac{1}{2}$ time to return to v_0 after serving the last request. Therefore, the lower bound on the competitive ratio becomes $\frac{OL(\sigma)}{OPT(\sigma)} \geq \frac{\Delta+2N-2}{2\Delta+N-2}$. The ratio $\frac{OL(\sigma)}{OPT(\sigma)}$ becomes arbitrarily close to 2 for $N \gg \Delta$. □

4 SINGLE-SERVER ONLINE TSP

Wefi rst present and analyze an algorithm for oTsp that achieves competitive ratio $\min\{2.04, 1+\frac{3}{2\beta\delta}\}$ for the nomadic version and $\min\{2, 1+\frac{2}{\max\{2,\beta\delta\}}\}$ for the homing version on line metric $\mathcal{L}$. This is interesting since the competitive ratio tends to 1 as $\delta = \Delta/D$ increases. Notice that $\beta = \min\{1, \frac{t_{max}}{\Delta}\}$, where t_{max} is the maximum arrival time among requests in σ. We will then present and analyze an algorithm that achieves competitive ratio of $\min\{1+\sqrt{2}, 2+\delta\}$ on arbitrary metric $\mathcal{M}$ for nomadic oTsp. For homing oTsp, there is a 2-competitive algorithm in the original model [3], which gives the same 2 ratio in the temporal locality model.

4.1 Algorithm on Line Metric

In the highlevel, the server initially needs to pick a direction (right or left). It traverse in the direction until all the requests are served enroute. Once there are no outstanding request in that direction, switch direction. This switching of direction continues until there is no outstanding request, in which case, server stays to its current position or goes to origin depending on whether TSP is nomadic or homing.

The pseudocode is in Algorithm 1. Server s is initially at origin o. Let $pos_{OL}(t)$ denote the position of server s at any time $t \geq 0$. At $t = 0$, $pos_{OL}(0) = o$. If $\delta \leq 1.442$, we use the algorithm of Bjelde *et al.* [7] of the original model. The value 1.442 is picked as a threshold through the calculation of δ for the equation $2.04 = 1 + \frac{3}{2\delta}$ keeping $\beta = 1$ (its maximum value). For $\delta > 1.442$, we use the following approach. Let S denote the set of outstanding requests. When $S = \emptyset$, the server s stays at its current position $pos_{OL}(t)$. Whenever $S \neq \emptyset$,

Algorithm 1: Single-server algorithm for oTsp on line metric with temporal locality $\Delta = \delta D$

$o \leftarrow$ origin where server resides initially
$pos_{OL}(t) \leftarrow$ the current position of server s on line $\mathcal{L}$ at time $t \geq 0$; $pos_{OL}(0) = o$
if $\delta \leq 1.442$ **then**
  run the algorithm by Bjelde *et al.* [7]
else
  if new request(s) arrives **then**
    $S \leftarrow$ the set of outstanding requests including the new request(s)
    $L \leftarrow$ the farthest position e_L among the requests in S on $\mathcal{L}$
    $R \leftarrow$ the farthest position e_R among the requests in S on $\mathcal{L}$ on the opposite side of L
    if both L and R are on the same side on L from $pos_{OL}(t)$ **then**
      $T \leftarrow$ a tour connecting $pos_{OL}(t)$ with L or R whichever is farthest
    else
      if $dist(pos_{OL}(t), L) \leq dist(pos_{OL}(t), R)$ **then**
        $T \leftarrow$ a tour connecting $pos_{OL}(t)$ with L and then L with R
      else
        $T \leftarrow$ a tour connecting $pos_{OL}(t)$ with R and then R with L
      end if
    end if
    server s traverses T until a new request arrives or T is traversed; for the nomadic version, after T is traversed stay at $pos_{OL}(t)$ but for homing version, return to origin o
  end if
end if

Algorithm 1 does the following. Out of the requests in S, Algorithm 1fi nds the two extreme positions L and R among the positions of the requests in S. In some cases there is no L or R (i.e., $pos_{OL}(t)$ coincides with L or R) and it does not hamper Algorithm 1 in any way. We have two cases: (i) both L and R are on the same side on $\mathcal{L}$ from $pos_{OL}(t)$, (ii) L is on one side and R is on another side on $\mathcal{L}$ from $pos_{OL}(t)$. For Case (i), TSP tour T is constructed connecting $pos_{OL}(t)$ with L or R whichever is farthest. For Case (ii), TSP tour T is constructed connecting $pos_{OL}(t)$ withfirst L then R if $dist(pos_{OL}(t), L) \leq dist(pos_{OL}(t), R)$, otherwise, withfirst R then L. Server s traverses T until either a new request arrives or T is completely traversed. After T completely is traversed and there is no new request, server s stays at its current position $pos_{OL}(t)$ for the nomadic version; for the homing version, it starts to return to origin o in full speed. Fig. 2 illustrates these ideas.

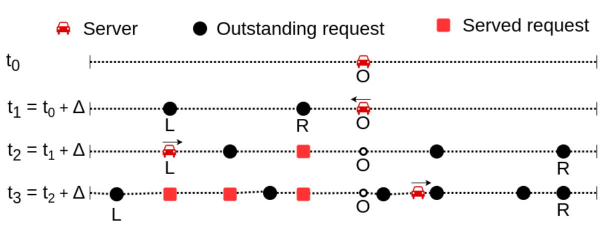

Figure 2: An illustration of Algorithm 1 for oTsp on line metric. At t_0, server s is on o, origin. At t_1, s picks the left direction to serve requests. At t_2, since s finds new outstanding request in right direction and there is no request on left of $pos_{ALG}(t_2)$, it changes direction and starts to traverse right direction. At t_3, it continues in the right direction since there are outstanding requests enroute.

Analysis of the Algorithm. Let L', R' be the two extreme positions among the positions of the requests in $\sigma \cup \{o\}$, meaning that all m requests and origin o have positions between L' and R' on $\mathcal{L}$ (inclusive). Let $D = |R' - L'|$. Recall that we define temporal locality Δ with respect to D, i.e., $\Delta = \delta D$. Let t_{max} be the arrival time of the request $r_{max} = (t_{max}, e_{max})$ in σ released last, i.e., there is no other request $r' = (t', e')$ with $t' > t_{max}$. Let t_{min} be the arrival time of the request $r_{min} = (t_{min}, e_{min})$ in σ releasedfi rst, i.e., there is no other request $r'' = (t'', e'')$ with $t'' < t_{min}$.

Wefi rst establish correctness.

Lemma1. *Algorithm 1 serves all the requests in σ.*

Proof. We prove this by contradiction. Support a request $r_i = (t_i, e_i)$ is not served. It must be the case that server s has not reached e_i. By construction, we have that e_i is either on the left or right of the current position $pos_{OL}(t)$ of server s. As soon as a request arrives, s moves towards that request. For two or more requests arriving at the same time, server s moves toward the one with shorter distance from $pos_{OL}(t)$. The direction is changed as soon as there is no outstanding request enroute, which happens at the extreme end of $\mathcal{L}$. After changing direction, the server s does not stop until all outstanding requests enroute are served. Therefore, e_i must be visited in the server s's traversal during left or right direction before server stops, hence a contradiction. □

We now establish the competitive ratio bound for nomadic oTsp.

Theorem3. *Algorithm 1 with temporal locality $\Delta = \delta D$ is $\min\{2.04, 1 + \frac{3}{2\beta\delta}\}$-competitive for nomadic oTsp defined on an interval of length D, where $\beta = \min\{1, \frac{t_{max}}{\Delta}\}$.*

Proof. Consider the input instance σ. Suppose all m requests are released at $t = 0$ (i.e., $t_{min} = t_{max} = 0$), then Algorithm 1finishes serving requests in σ in $|\mathcal{T}|$ time, i.e. $OL(\sigma) \leq |\mathcal{T}|$. Any optimal algorithm OPT also needs at least $|\mathcal{T}|$ time, i.e., $OPT(\sigma) \geq |\mathcal{T}|$. Therefore, Algorithm 1 is 1-competitive. Suppose not all requests are released at $t = 0$ (i.e., $t_{min} = 0$ but $t_{max} > 0$) , there must be at least a request released at time $t = \Delta = \delta D$ since temporal locality is Δ and hence $OPT(\sigma) \geq \delta D$.

Consider $r_{max} = (t_{max}, e_{max})$, the request in σ released last. Since r_{max} cannot be served before t_{max} by any algorithm, $OPT(\sigma) \geq t_{max}$. At t_{max}, the tour T computed by Algorithm 1 cannot be larger than $\frac{3}{2}D$. This is because either $dist(pos_{OL}(t_{max}), L) \leq \frac{1}{2}D$ or $dist(pos_{OL}(t_{max}), R) \leq \frac{1}{2}D$ and server s picks L or R depending on whichever is of smaller distance and $dist(L,R) \leq D$. After t_{max}, each outstanding request waits for at most $\frac{3}{2}D$ time units before being served. Therefore, $OL(\sigma) \leq t_{max} + \frac{3}{2}D$. Combining these results,

$$\frac{OL(\sigma)}{OPT(\sigma)} \leq \frac{t_{max} + \frac{3}{2}D}{\max\{t_{max}, \delta D\}}.$$

We have two cases: (a) $t_{max} \geq \delta D$ or (b) $t_{max} < \delta D$. For Case (a), $D \leq t_{max}/\delta$ and hence

$$OL(\sigma) \leq \left(\frac{t_{max}}{t_{max}} + \frac{\frac{3}{2}\frac{t_{max}}{\delta}}{t_{max}}\right) OPT(\sigma) = (1 + \frac{3}{2\delta}) OPT(\sigma).$$

For Case (b), replacing t_{max} with δD, we obtain

$$OL(\sigma) \leq \left(\frac{\delta D}{\delta D} + \frac{\frac{3}{2}D}{\delta D}\right) OPT(\sigma) = (1 + \frac{3}{2\delta}) OPT(\sigma).$$

Now suppose $t_{min} > 0$, i.e., no request is released at time $t = 0$. We know that $t_{min} < \delta D$. Even in this case, if $t_{max} \geq t_{min} + \delta D \geq \delta D$, we obtain the competitive ratio of $(1 + \frac{3}{2\delta})$ as above. Therefore, for the case of $0 < t_{min}, t_{max} < \delta D$, we have that $OPT(\sigma) \geq t_{max}$ and also $OPT(\sigma) \geq \beta\delta D$ since $\beta = \min\{1, \frac{t_{max}}{\Delta}\}$. Therefore,

$$\frac{OL(\sigma)}{OPT(\sigma)} \leq \frac{t_{max} + \frac{3}{2}D}{\max\{t_{max}, \beta\delta D\}} = 1 + \frac{3}{2\beta\delta}.$$

Finally, we now analyze the competitive ratio that does not depend on Δ. This is directly obtained from Bjelde *et al.* [7] where they established 2.04 competitive ratio for their algorithm. □

We now establish the following theorem for homing oTsp.

Theorem4. *Algorithm 1 with temporal locality $\Delta = \delta D$ is $\min\{2, 1 + \frac{2}{\max\{2,\beta\delta\}}\}$-competitive for homing* oTsp *defined on an interval of length D.*

Proof. Consider the input instance σ. Suppose all m requests are released at $t = 0$ (i.e., $t_{min} = t_{max} = 0$). Starting from o running Algorithm 1, server s finishes serving requests in σ and return to o in $2D$ time. Therefore, $OL(\sigma) \leq 2D$. Any optimal algorithm OPT also needs at least $2D$ time to serve the requests, starting from o and returning to o after serving all the requests, i.e., $OPT(\sigma) \geq 2D$. Therefore, Algorithm 1 is 1-competitive.

If not all requests are released at $t = 0$ (i.e., $t_{min} = 0$ but $t_{max} \neq 0$), there must be at least a request released at time $t = \Delta = \delta D$ and hence $OPT(\sigma) \geq \delta D$.

Consider $r_{max} = (t_{max}, e_{max})$. Since r_{max} cannot be served before t_{max}, $OPT(\sigma) \geq t_{max} + dist(o, e_{max})$. At t_{max}, the tour T computed by Algorithm 1 cannot be longer than $2D$, to serve all the outstanding requests and return to origin o. Therefore, $OL(\sigma) \leq t_{max} + 2D$. Combining the above results,

$$\frac{OL(\sigma)}{OPT(\sigma)} \leq \frac{t_{max} + 2D}{\max\{t_{max}, \max\{2, \delta\}D\}}.$$

We have two cases: (a) $t_{max} \geq \max\{2, \delta\}D$ or (b) $t_{max} < \max\{2, \delta\}D$. For Case (a), $D \leq t_{max}/\max\{2, \delta\}$ and hence $OL(\sigma) \leq (1 + \frac{2}{\max\{2,\delta\}})OPT(\sigma)$. For Case (b), replacing t_{max} with $\max\{2, \delta\}D$, we obtain $OL(\sigma) \leq (1 + \frac{2}{\max\{2,\delta\}})OPT(\sigma)$.

We now consider the case where thefi rst (set of) request(s) is released at time $0 < t_{min} < \Delta$. Even in this case, if $t_{max} \geq t_{min} + \delta D > \delta D$, we obtain the competitive ratio of $(1 + \frac{2}{\max\{2,\delta\}})$. If $t_{max} < \delta D$, then it must be the case that $t_{min} = t_{max}$. In this case we have that

$$\frac{OL(\sigma)}{OPT(\sigma)} \leq \frac{t_{max} + 2D}{\max\{t_{max}, \max\{2, \beta\delta\}D\}} = 1 + \frac{2}{\max\{2, \beta\delta\}}.$$

Finally, we now analyze the competitive ratio without dependence on Δ. $OL(\sigma) \leq t_{max} + 2D$. It is obvious that $OPT(\sigma) \geq t_{max}$ and $OPT(\sigma) \geq 2D$. Therefore, $OL(\sigma) \leq 2\ OPT(\sigma)$. □

4.2 Algorithm on Arbitrary Metric

In the highlevel, the server either runs the *Return Home* algorithm from Lipmann [13] or our approach which asks it to return to the origin as soon as new request arrives. At origin, it computes a TSP tour of all outstanding requests and starts traversing the tour. Doing so, the only extra distance traversed is δD every time a new request arrives. When $\delta < \sqrt{2} - 1$, the competitive ratio becomes $(2 + \delta)$, better than original model result of 2.41 $(= 1 + \sqrt{2})$.

Algorithm 2: Single-server algorithm for nomadic oTsp on arbitrary metric with temporal locality $\Delta = \delta D$

$o \leftarrow$ origin where server resides initially
$pos_{OL}(t) \leftarrow$ the current position of server s on metric $\mathcal{M}$ at time $t \geq 0$; $pos_{OL}(0) = o$
if $\delta \geq \sqrt{2} - 1$ **then**
  run the ReturnHome algorithm by Lipmann [13]
else
  if new request(s) arrives **then**
    $S \leftarrow$ the set of outstanding including the new request(s)
    $\mathcal{T} \leftarrow$ the minimum cost TSP tour that connects the positions of the requests in $S \cup \{o\}$ with o being the one endpoint of the tour
    $T' \leftarrow$ the tour that connects the current position $pos_{OL}(t)$ of server s with o
    $T \leftarrow T' \cup \mathcal{T}$
    server s traverses tour T until either itfi nishes traversing T or new request(s) arrives
  end if
end if

The pseudocode of the algorithm is given in Algorithm 2. Server s is initially on o, the origin. In Algorithm 2, server s serves the requests as follows. Let $pos_{OL}(t)$ be the current position of s at time t; $pos_{OL}(0) = o$. Let S the set of outstanding requests at time t. Whenever a new request arrives at time t, s constructs a tour T as follows. Itfi nds a minimum length TSP tour $\mathcal{T}$ that connects o with each position e_i on the requests in S. It also finds a tour T' that connects $pos_{OL}(t)$ with o. Therefore, $T = T' \cup \mathcal{T}$. The server s then start traversing the tour T starting from $pos_{OL}(t)$ until a new request arrives or until T is completely traversed. If T is completely traversed before any request arrives, server s stops at the endpoint of T until a new request arrives. If a new request arrives at time $t' > t$ beforefi nish traversing T, s again computes T considering the set of outstanding requests S at time t' as discussed above and start traversing the tour T. Fig. 3 illustrates these ideas.

Wefi rst prove correctness of Algorithm 2.

Lemma2. *Algorithm 2 serves all the requests in σ.*

Proof. We prove this by contradiction. Suppose a request $r_i = (t_i, e_i)$ is not served. When r_i arrives at t_i, it must be outstanding. At t_i, the server s computes the TSP tour $\mathcal{T}$ that visits the locations of all outstanding requests with origin as an one endpoint of the tour. Server s does not stop until $\mathcal{T}$ is fully traversed serving all the outstanding requests during its traversal. Since r_i was outstanding at t_i and after, it must have been served by server s running Algorithm 2, hence a contraction. □

We now establish the competitive ratio bound for nomadic oTsp.

Theorem5. *Algorithm 2 with temporal locality $\Delta = \delta D$ is $\min\{1 + \sqrt{2}, 2 + \delta\}$-competitive for nomadic* oTsp *on arbitrary metric of diameter D.*

Proof. Consider the input instance σ. If $\delta \geq \sqrt{2} - 1$, Algorithm 2 runs the *ReturnHome* algorithm by Lipmann [13]. Since *ReturnHome* is $(1 + \sqrt{2})$-competitive, Algorithm 2 is $(1 + \sqrt{2})$-competitive for nomadic oTsp.

For the case of $\delta < \sqrt{2} - 1$, we prove $(2 + \delta)$-competitive ratio for Algorithm 2. Let $r_{max} = (t_{max}, e_{max})$ be the request in σ released last. We show that, at t_{max}, the length of the tour T computed by Algorithm 2 cannot be larger than $|\mathcal{T}| + \delta D$, i.e., $|T| \leq |\mathcal{T}| + \delta D$. Since temporal locality is $\Delta = \delta D$, the server cannot be more than δD distance away from the origin o. Therefore, after t_{max}, server can return to origin in Δ time and thenfi nish traversing the tour $\mathcal{T}$ in $|\mathcal{T}|$ time. In other words, after t_{max}, each outstanding request waits for at most $|\mathcal{T}| + \delta D$ time units before being served. Therefore, $OL(\sigma) \leq t_{max} + |\mathcal{T}| + \delta D$. We have that $OPT(\sigma) \geq t_{max}$. Irrespective of whether all the requests are released at time $t \geq 0$, since OPT must visit all the requests , it pays at least $|\mathcal{T}|$, i.e., $OPT(\sigma) \geq |\mathcal{T}|$. Combining the above results,

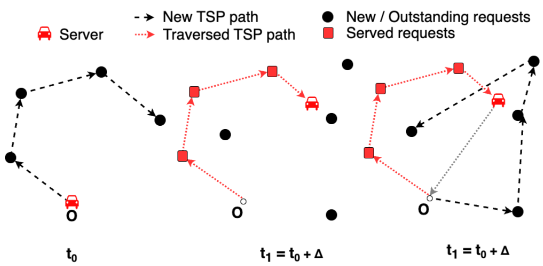

Figure 3: An illustration of Algorithm 2 for nomadic oTsp on arbitrary metric: (*left*) At t_0, server at o computes a TSP tour of 4 requests and starts to traverse the tour, (*middle*) When three new requests arrive at t_1, server computes a TSP tour T of outstanding requests with one end point being o, (*right*) server starts to traverse T first reaching to o from its current location $pos_{OL}(t_1)$.

$$\frac{OL(\sigma)}{OPT(\sigma)} \leq \frac{t_{max} + |\mathcal{T}| + \delta D}{\max\{t_{max}, |\mathcal{T}|\}}.$$

Since the diameter is D, we have that $|\mathcal{T}| \geq D$. We have two cases: (a) $t_{max} \geq |\mathcal{T}|$ or (b) $t_{max} < |\mathcal{T}|$. For Case (a), $OL(\sigma) \leq \left(\frac{t_{max}}{t_{max}} + \frac{t_{max}}{t_{max}} + \frac{\delta t_{max}}{t_{max}}\right) OPT(\sigma) = (2 + \delta) OPT(\sigma)$. For Case (b), $OL(\sigma) \leq \left(\frac{|\mathcal{T}|}{|\mathcal{T}|} + \frac{|\mathcal{T}|}{|\mathcal{T}|} + \frac{\delta|\mathcal{T}|}{|\mathcal{T}|}\right) OPT(\sigma) = (2 + \delta) OPT(\sigma)$. □

5 SINGLE SERVER ONLINE DIAL-A-RIDE

Wefi rst discuss an algorithm for both homing and nomadic oDarp on line metric. We then discuss an algorithm for nomadic oDarp on arbitrary metric. For homing oDarp, there is a 2-competitive algorithm in the original model [2, 9], which gives the same 2 ratio in the temporal locality model.

5.1 Algorithm on Line Metric

We modify Algorithm 1 to solve oDarp on line metric $\mathcal{L}$. We consider both nomadic and homing oDarp. The only modification is the server needs to visit e_i of each request r_i before d_i to consider r_i served. The tour computed takes into account this requirement.

Theorem6. *Algorithm 1 with temporal locality $\Delta = \delta D$ is $\min\{4, 1 + \frac{3}{\beta\delta}\}$-competitive for nomadic oDarp on an interval of length D, where $\beta = \min\{1, \frac{t_{max}}{\Delta}\}$.*

Proof. Consider the input sequence σ. Suppose all m requests are released at time $t = 0$. Algorithm 1 is clearly 1-competitive. Suppose not all requests are released at time $t = 0$, then there must be at least a request $r_i = (t_i, e_i, d_i)$ with $t_i \geq \delta D$ since the temporal locality is δD, i.e., $OPT(\sigma) \geq \delta D$.

Let $r_{max} = (t_{max}, e_{max}, d_{max})$ be the request released last. $OPT(\sigma) \geq t_{max}$. We have two upper bounds based on whether the requests in σ are all increasing requests or there is at least one request that is non-increasing. An increasing request means, the destination location is farther from the origin than its source location. If all requests are increasing, then $OL(\sigma) \leq t_{max} + 2D$. This is because, after t_{max}, traversing from originfi rst to left (or right) extreme point on line and then to right (or left) extreme point and finally to origin servers all the requests. However, if there is at least one non-increasing request, $OL(\sigma) \leq t_{max} + 3D$. This is because the extreme point visitedfi rst need to be visited again and then back to the origin to handle the non-increasing part. Therefore, for any combination of increasing and non-increasing requests in σ,

$$\frac{OL(\sigma)}{OPT(\sigma)} \leq \frac{t_{max} + 3D}{\max\{t_{max}, \delta D\}} \leq 1 + \frac{3}{\delta}.$$

Now suppose $< 0 < t_{max} < \delta D$. In this case,

$$\frac{OL(\sigma)}{OPT(\sigma)} \leq \frac{t_{max} + 3D}{\max\{t_{max}, \beta\delta D\}} \leq 1 + \frac{3}{\beta\delta}.$$

Finally, we analyze the competitive ratio independent of Δ. We have that $OL(\sigma) \leq t_{max} + 3D$. It is obvious that $OPT(\sigma) \geq t_{max}$ and $OPT(\sigma) \geq D$. Therefore, we obtain $OL(\sigma) \leq 4\, OPT(\sigma)$. □

Theorem7. *Algorithm 1 with temporal locality $\Delta = \delta D$ is $\min\{3, 1 + \frac{4}{\max\{2, \beta\delta\}}\}$-competitive for homing oDarp on an interval of length D.*

We omit the proof of this theorem due to space constraints.

5.2 Algorithm on Arbitrary Metric

We modify Algorithm 2 to solve oDarp on arbitrary metric $\mathcal{M}$. We consider only nomadic oDarp; homing oDarp is solved with 2-competitive ratio using the existing algorithm [2, 9] in the original model. The only modification is the server needs to visit e_i of each request r_i before d_i to consider r_i served. The tour computed takes into account this requirement. Fig. 4 illustrates these ideas.

Theorem8. *Algorithm 2 with temporal locality $\Delta = \delta D$ is $\min\{2.457, 2 + \delta\}$-competitive for nomadic oDarp on arbitrary metric with diameter D.*

Proof. Consider the input instance σ. If $\delta \geq 0.457$, Algorithm 2 runs the *Lazy* algorithm by [4] which provides a competitive ratio of 2.457 for nomadic oDarp.

For the case of $\delta < 0.457$, we prove $(2 + \delta)$-competitive ratio for Algorithm 2. Let $r_{max} = (t_{max}, e_{max})$ be the request in σ released last. Let $\mathcal{T}_{Darp}$ be the minimum length tour for the requests in σ such that one endpoint of $\mathcal{T}_{Darp}$ is origin o and for each request r_i, its e_i and d_i come consecutively in $\mathcal{T}_{Darp}$ exactly once.

At t_{max}, the length of the tour T computed by Algorithm 2 cannot be larger than $|\mathcal{T}_{Darp}| + \delta D$, i.e., $|T| \leq |\mathcal{T}_{Darp}| + \delta D$. Since temporal locality is $\Delta = \delta D$, the server cannot be more than δD distance away from the origin o at anytime t. After t_{max}, each outstanding request is served before time $|\mathcal{T}_{Darp}| + \delta D$. Therefore, $OL(\sigma) \leq t_{max} + |\mathcal{T}_{Darp}| + \delta D$. We have that $OPT(\sigma) \geq t_{max}$.

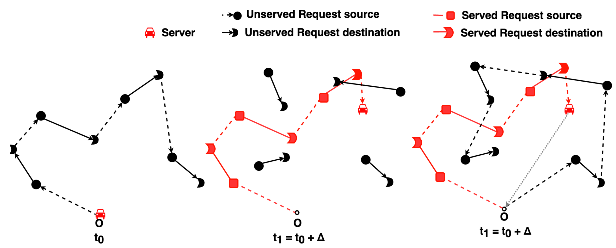

Figure 4: An illustration of Algorithm 2 for oDarp on arbitrary metric: (*left*) At t_0, server at o computes a TSP tour of 4 requests and starts to traverse the tour, (*middle*) When three new requests arrive at t_1, server computes a TSP tour T of outstanding requests with one end point of T being o, (*right*) server starts to traverse T first reaching to o from its current location $pos_{OL}(t_1)$.

Irrespective of whether all the requests are released at time $t \geq 0$, since OPT must visit all the requests , it pays at least $|\mathcal{T}_{Darp}|$, i.e., $OPT(\sigma) \geq |\mathcal{T}_{Darp}|$. Combining the above results,

$$\frac{OL(\sigma)}{OPT(\sigma)} \leq \frac{t_{max} + |\mathcal{T}_{Darp}| + \delta D}{\max\{t_{max}, |\mathcal{T}_{Darp}|\}}.$$

Since $|\mathcal{T}_{Darp}| \geq D$, we obtain $OL(\sigma) \leq (2+\delta)OPT(\sigma)$. □

6 $k > 1$ SERVER EXTENSIONS

We now discuss how the competitive ratios for $k = 1$ server extend to $k > 1$ servers for both oTsp and oDarp. We omit proofs of four theorems in this section due to space constraints.

Theorem9. *Parallelized Algorithm 1 with temporal locality $\Delta = \delta D$ is $\min\{2.04, 1 + \frac{1}{\beta\delta}\}$-competitive for nomadic* oTsp *on an interval of length D for $k > 1$ servers.*

Proof. Consider the input instance σ. Suppose all m requests are released at $t = 0$ (i.e., $t_{min} = t_{max} = 0$). Let $\mathcal{T}_1, \ldots \mathcal{T}_k$ be the minimum cost TSP tours for k servers starting from o to serve the requests in σ such that the length of each tour $\mathcal{T}_j, 1 \leq j \leq k$, is minimized. Parallelized Algorithm 1fi nishes serving requests in σ in $\max_{1\leq j\leq k} |\mathcal{T}_j|$ time, i.e. $OL(\sigma) \leq \max_{1\leq j\leq k} |\mathcal{T}_j|$. Any optimal algorithm OPT also needs at least $\max_{1\leq j\leq k} |\mathcal{T}_j|$ time, i.e., $OPT(\sigma) \geq \max_{1\leq j\leq k} |\mathcal{T}_j|$. Therefore, Algorithm 1 is 1-competitive. Suppose not all requests are released at $t = 0$ (i.e., $t_{min} = 0$ but $t_{max} > 0$), there must be at least a request released at time $t = \Delta = \delta D$ and hence $OPT(\sigma) \geq \delta D$.

Consider $r_{max} = (t_{max}, e_{max})$, the request in σ released last. Since r_{max} cannot be served before t_{max} by any online algorithm, $OPT(\sigma) \geq t_{max}$. At t_{max}, the tour T computed by Algorithm 1 for each server s_j cannot be larger than D. After t_{max}, each outstanding request waits for at most D time units before being served. Therefore, $OL(\sigma) \leq t_{max} + D$. Combining the above results, $\frac{OL(\sigma)}{OPT(\sigma)} \leq \frac{t_{max}+D}{\max\{t_{max},\delta D\}} = \left(1 + \frac{1}{\delta}\right)$.

Now suppose $t_{min} > 0$, i.e., no request is released at time $t = 0$. We know that $t_{min} < \delta D$. Even in this case, if $t_{max} \geq t_{min} + \delta D \geq \delta D$, we obtain the competitive ratio of $(1 + \frac{1}{\delta})$ as above. Therefore, for the case of $0 < t_{min}, t_{max} < \delta D$, we have that $OPT(\sigma) \geq t_{max}$ and also $OPT(\sigma) \geq \beta\delta D$. Therefore, $OL(\sigma) \leq \left(\frac{t_{max}+D}{\max\{t_{max},\beta\delta D\}}\right) OPT(\sigma) = \left(1 + \frac{1}{\beta\delta}\right) OPT(\sigma)$.

Finally, the 2.04 competitive ratio independent on Δ is immediate from the result of Bjelde *et al.* [7] in the original model. □

Theorem10. *Parallelized Algorithm 1 with temporal locality $\Delta = \delta D$ is $\min\{2, 1 + \frac{2}{\max\{2,\beta\delta\}}\}$-competitive for homing* oTsp *on an interval of length D for $k > 1$ servers.*

Theorem11. *Parallelized Algorithm 2 with temporal locality $\Delta = \delta D$ is $\min\{1 + \sqrt{2}, 2 + \delta \min\{\gamma, 1\}\}$-competitive for nomadic* oTsp *on arbitrary metric with diameter D for $k > 1$ servers.*

Proof. Consider the input instance σ. If $\delta \geq \sqrt{2} - 1$, Algorithm 2 runs the *GroupReturnHome* (GRH) algorithm by [8] which achieves $(1 + \sqrt{2})$ competitive ratio for nomadic oTsp for k servers.

For the case of $\delta < \sqrt{2}-1$, let $r_{max} = (t_{max}, e_{max})$ be the request in σ released last. At t_{max}, the length of the tour T_j computed by Algorithm 2 for server s_j cannot be larger than $|\mathcal{T}_j| + \delta D$, i.e., $|T_j| \leq |\mathcal{T}_j| + \delta D$. Since temporal locality $\Delta = \delta D$, the server cannot be more than δD distance away from the origin o. After t_{max}, each outstanding request served by s_j waits for at most $|\mathcal{T}_j| + \delta D$ time units before being served by s_j. Therefore, $OL(\sigma) \leq t_{max} + \max_{1\leq j\leq k} |\mathcal{T}_j| + \delta D$. We have that $OPT(\sigma) \geq t_{max}$. Irrespective of whether all the requests are released at time $t \geq 0$, since OPT must visit all the requests, it must pay at least $\max_{1\leq j\leq k} |\mathcal{T}_j|$, i.e., $OPT(\sigma) \geq \max_{1\leq j\leq k} |\mathcal{T}_j|$. Combining the above results,

$$\frac{OL(\sigma)}{OPT(\sigma)} \leq \frac{t_{max} + \max_{1\leq j\leq k} |\mathcal{T}_j| + \delta D}{\max\{t_{max}, \max_{1\leq j\leq k} |\mathcal{T}_j|\}}.$$

Since diameter is D, $\max_{1\leq j\leq k} |\mathcal{T}_j| \geq \min\{\gamma, 1\}D$ for some $\gamma > 0$. Either (a) $t_{max} \geq \max_{1\leq j\leq k} |\mathcal{T}_j|$ or (b) $t_{max} < \max_{1\leq j\leq k} |\mathcal{T}_j|$. For both cases, $OL(\sigma) \leq (2 + \delta \min\{\gamma, 1\})OPT(\sigma)$. □

Theorem12. *Parallelized Algorithm 1 with temporal locality $\Delta = \delta D$ is $\min\{4, 1 + \frac{3}{2\beta\delta}\}$-competitive for nomadic* oDarp *on an interval of length D for $k > 1$ servers.*

Theorem13. *Parallelized Algorithm 1 with temporal locality $\Delta = \delta D$ is $\min\{3, 1 + \frac{2}{\beta\delta}\}$-competitive for homing* oDarp *on an interval of length D for $k > 1$ servers.*

Theorem14. *Parallelized Algorithm 2 with temporal locality $\Delta = \delta D$ is $\min\{2.457, 2 + \delta \min\{\gamma, 1\}\}$-competitive for nomadic* oDarp *on arbitrary metric with diameter D.*

7 CONCLUDING REMARKS

In this paper, we have proposed the new clairvoyance model, called temporal locality, and studied its power on online routing. We first established a lower bound of 2-competitive ratio for oTsp and (uncapacitated) oDarp in arbitrary metric. We then showed that, in arbitrary metric, the competitive ratios better than the currently known can be obtained with smaller temporal locality (i.e., $\delta = \Delta/D < \sqrt{2}-1$ for nomadic oTsp and $\delta < 0.457$ for nomadic oDarp). For line metric, the competitive ratio gets closer to 1 with larger temporal locality for both homing and nomadic oTsp and oDarp. For future work, it will be interesting to consider other online routing problems and/or objective functions (e.g., sum of completion times) with temporal locality.

REFERENCES

[1] Luca Allulli, Giorgio Ausiello, Vincenzo Bonifaci, and Luigi Laura. 2008. On the power of lookahead in on-line server routing problems. *Theor. Comput. Sci.* 408, 2-3 (2008), 116–128. https://doi.org/10.1016/J.TCS.2008.08.003

[2] Norbert Ascheuer, Sven Oliver Krumke, and Jörg Rambau. 2000. Online Dial-a-Ride Problems: Minimizing the Completion Time. In *STACS*. Springer, 639–650.

[3] Giorgio Ausiello, Esteban Feuerstein, Stefano Leonardi, Leen Stougie, and Maurizio Talamo. 2001. Algorithms for the On-Line Travelling Salesman. *Algorithmica* 29, 4 (2001), 560–581.

[4] Júlia Baligács, Yann Disser, Farehe Soheil, and David Weckbecker. 2023. Tight Analysis of the Lazy Algorithm for Open Online Dial-a-Ride. In *WADS*. Springer, 43–64.

[5] Marcin Bienkowski, Artur Kraska, and Hsiang-Hsuan Liu. 2021. Traveling Repairperson, Unrelated Machines, and Other Stories About Average Completion Times. In *ICALP (LIPIcs, Vol. 198)*. 28:1–28:20.

[6] Alexander Birx, Yann Disser, and Kevin Schewior. 2023. Improved Bounds for Open Online Dial-a-Ride on the Line. *Algorithmica* 85, 5 (2023), 1372–1414.

[7] A. Bjelde, Y. Disser, J. Hackfeld, C. Hansknecht, M. Lipmann, J. Meißner, K. Schewior, M. Schlöter, and L. Stougie. 2021. Tight bounds for online TSP on the line. *ACM Transactions on Algorithms* 17, 1 (2021), 3:1–3:58.

[8] Vincenzo Bonifaci and Leen Stougie. 2009. Online k-server routing problems. *Theory of Computing Systems* 45, 3 (2009), 470–485.

[9] Esteban Feuerstein and Leen Stougie. 2001. On-line single-server dial-a-ride problems. *Theoretical Computer Science* 268, 1 (2001), 91–105.

[10] Themistoklis Gouleakis, Konstantinos Lakis, and Golnoosh Shahkarami. 2023. Learning-Augmented Algorithms for Online TSP on the Line. In *AAAI*, Brian Williams, Yiling Chen, and Jennifer Neville (Eds.). AAAI Press, 11989–11996.

[11] Patrick Jaillet and Michael R. Wagner. 2006. Online Routing Problems: Value of Advanced Information as Improved Competitive Ratios. *Transportation Science* 40, 2 (2006), 200–210.

[12] Sven Oliver Krumke, Willem de Paepe, Diana Poensgen, and Leen Stougie. 2003. News from the online traveling repairman. *Theor. Comput. Sci.* 295 (2003), 279–294.

[13] Maarten Lipmann. 2003. On-line routing. *Ph.D. Thesis, Technical University of Eindhoven* (2003).