Orienteering problem with time-windows and updating delay

Bertrand Jouve, Marc Demange, David Ellison

Orienteering problem with time-windows and updating delay

Marc Demange[a], David Ellison[a], Bertrand Jouve[b]

[a] *RMIT University, School of Science, Melbourne, Australia*
[b] *LISST UMR5193, Toulouse University, CNRS, France*

Abstract

The Orienteering Problem with Time Window and Delay (`OPTiWinD`) is a variant of the online orienteering problem. A series of requests appear in various locations while a vehicle moves within the territory to serve them. Each request has a time window during which it can be served and a weight which describes its importance. There is also a minimum delay T between successive requests. The objective is to find a path for the vehicles that maximises the sum of the weights of the requests served. We further assume that the length of each time window is equal to the diameter of the territory. We study the optimal performance and competitive ratio for the set of instances with n requests. We obtain complete resolution for T at least half of the diameter, small values of T or small values of n, as well as partial results in the remaining cases.

Keywords *— Online orienteering problem, Online Scheduling, Competitive analysis, Combinatorial optimisation.*

1. Introduction and related work

1.1. Defining the problem

This work is motivated by a wildfire emergency management problem. On very hot days and during the dry season, in regions exposed to wildfires, it is not rare to observe, in a single region, up to one hundred fire ignitions per day. Some of these fires spread very quickly and, if not extinguished quickly, they escape and become completely out of control. Although the risk factors, such as temperature, wind, fuel load, dryness, etc. and the main causes of ignition are well-understood, it is almost impossible to forecast, during a day, where and when a fire will occur. Wildfires are already an important threat to human lives, goods and the environment; and in the near future, this is expected to become worse due to climate change.

Email addresses: `marc.demange@rmit.edu.au` (Marc Demange), `david.ellison2@rmit.edu.au` (David Ellison), `jouve@univ-tlse2.fr` (Bertrand Jouve)

For emergency management commanders, one of the main challenges is to rely on efficient decision making. And when disaster strikes, especially when fatalities occur, they need to be able to justify their decisions once the crisis is over. Hence, in order to satisfy legal constraints, the decision chain needs to be very clear and well-defined. The most critical decision commanders need to make during the initial outbreak is the allocation of firefighting resources in cases where there are not enough resources to keep all the fires under control. Such a decision relies on the information available about the on-going fires and the threats they pose. Early warning systems are set-up to detect new fires, but once a smoke plume has been detected, an on-site reconnaissance is necessary to assess the importance of the fire, evaluate the possible induced risk and estimate how long until the fire can no longer be contained. Based on this information, the management team decides whether or not to send firefighting resources to each new fire. Given that making this decision requires having evaluated all the risks associated with the fire, each new fire is only taken into account after a delay corresponding to the time used by the reconnaissance team. We will assume that there is a single firefighter team which cannot be divided, thus two fires igniting simultaneously will be processed one at a time with a delay between them. Natural generalisations may induce several or splittable teams. The objective is to schedule the movements of the team by selecting which fires to extinguish and when, in order to maximise the total weight of the fires contained. We model this situation using a sequence of fires, each with a weight representing the expected loss if the fire goes out of control and a time window in which the fire has been assessed and remains controllable. We include a minimum delay between the release time of fires.

This model corresponds to the standard Orienteering Problem in a space X, with Time Windows and with the addition of time delays. This variant will be called *The Orienteering Problem with Time Window and Delay* (`OPTiWinD`). In this problem, time is continuous and new requests can appear at any moment and at any point in X. Each request has a location, a weight and a time window during which it can be served. There is a minimum delay between successive new requests. The player controls a unique vehicle which travels to serve the requests. A request is instantaneously satisfied when it is visited. The player's objective is to find a path in the space that maximises the sum of the weights of the requests served. A strategy of the player is described as an algorithm.

We will consider online algorithms for instances where each request is revealed at the start of its time window. This corresponds to the reality of our motivating problem, for which the difficulty to establish firefighting strategies is largely due to not knowing in advance when and where the fires will appear.

1.2. Related work

The underlying combinatorial optimisation problem is a natural generalisation of the Metric Travelling Salesman Problem (Metric TSP), which is also called the *Orienteering Problem* [6]: one is given a time limit and a graph with n vertices including departure and arrival points. Each vertex is associated with a score and each edge with a distance. The objective is to select some vertices

and an order to visit them while travelling at constant speed 1 from the departure point to the arrival point, so that the time limit is not exceeded and the total score of visited vertices is maximised. In Laporte and Martello [9], the orienteering problem is called *Selective TSP*. The same problem has been studied under other names, in particular *Bank Robber Problem* [3] and *Maximum Collection Problem*. The dual problem where the aim is to collect a fixed targeted total score in the minimum time is called *Quota TSP* [3] or also *Prize Collecting Problem*. These problems fall into the class of *Travelling Salesman Problems with Profits* [5]. Our application deals with the version of the orienteering problem with time windows [10]: instead of an overall time limit, each vertex is associated with a release time and a deadline and it must be visited between these two dates in order to collect the associated score.

Some of these problems have been considered in the online case, also referred as *dynamic case*. In most cases, the online setup consists in revealing requests (vertices) over time, so that the online algorithm needs to decide about the movements of the vehicle without knowing when and where the next request will be revealed. The Metric Travelling Salesman has been considered under this setup in [1, 2], for instance, and the Quota TSP in [1]. When the delay is zero, `OPTiWinD` is equivalent to the *Whack-a-Mole Problem*, which is defined in Gutiérrez et al. [7], or the *Dynamic Traveling Repair Problem* [8]. For a space which is a truncated line $[-L, L]$, Gutiérrez et al. [7] note that the only non trivial cases are those for $\theta/4 < L \leq \theta$, where θ denotes the size of the time windows. Thus, they study the case $\theta = L$ with moles that may only arrive at integer positions on $[-L, L]$. In the case of the segment, we study in this paper the case where $L = \theta/2$, which is not examined in Gutiérrez et al. [7]. To our knowledge, no previous paper deals with a delay between requests.

2. Preliminaries

2.1. Geodesic metric spaces

A metric space $E = (X, d)$ is said to be *geodesic* [4] if for any $a, b \in X$, there is a continuous path $\gamma : [0, 1] \to X$ from a to b such that the range of γ is isometric to the segment $[0, d(a, b)]$.

Defining the offline orienteering problem on graphs or on metric spaces is equivalent. The distance, or travel time, between requests is the only aspect of the underlying metric space relevant to the game. However, in the online case, the position of the vehicle must be defined at all time. Hence the online orienteering problem is defined either on graphs (e.g. [8]) or on geodesic metric spaces (e.g. [2]), producing two slightly different online games.

Having a geodesic metric space means that the distance between two points is equal to the length of the shortest path connecting them. The vehicle is allowed to change directions whenever a new request appears, even when the vehicle is travelling between requests. Alternatively, on graphs, the vehicle can only select a new direction after reaching a vertex.

2.2. Definitions and notations

For geodesic metric spaces with only one pair of points maximising the distance, it is natural to consider a worst case scenario when all the requests appear on these two points. As it happens, the case where E is a segment is critical to understanding the general case. Thus, except in Section 8, we will consider the online problem on the segment $[-1, 1]$ with the vehicle starting at 0. We will show in Section 8 that most of the results for the segment can be extended to bounded geodesic metric spaces of diameter 2, and we will give counter-examples to the others.

An instance of `OPTiWinD`is a set of requests, each characterised by a triple $(A, [r, r'], \delta)$ where $A \in X$ is the location of the request, r is the release time, r' is the time when the request is lost if it is not served and $\delta \in \mathbb{R}_+^*$ is the *weight* (or score) associated with the request.

A request $(A, [r, r'], \delta)$ is said to be *served* if the vehicle is on A at a time $t \in [r, r']$. A feasible solution is an itinerary for the vehicle starting at O at time $t = 0$ and serving some requests. The objective is to maximise the total weight of the requests served. We denote by $\mathcal{I}_{n,T}$ the set of instances of `OPTiWinD` on $[-1, 1]$ with at most n requests and delay at least T between requests.

We will make the following assumptions about the `OPTiWinD` Problem:

- We consider that X is the segment $[-1, 1]$, except in Section 8 where we extend the problem to the case of a bounded geodesic metric space X of diameter 2.
- The vehicle is able to move no faster than unit speed in X and a request is instantaneously satisfied when it is visited.
- We assume that each time window is of the form $[r, r + 2]$ where r is the release time. Thus, once a request appears, the vehicle always has a chance to reach its location in time from anywhere in E.
- The release times of any two successive requests differ by at least a delay $T \geq 0$, which is independent from the requests.
- Online algorithms for `OPTiWinD` learn about a new request (its location and weight) at the start of its associated time window. The vehicle can serve it at any time within that time window or it can choose not to serve it at all. The geodesic metric space is completely known from the beginning.

In the following, the sets of non-negative integers and real numbers are denoted by $\mathbb{N}$ and $\mathbb{R}$ respectively. The golden ratio is φ. The i-th request is denoted f_i and its release time is t_i. The weight of f_i is δ_i and $S_i = \sum_{k=1}^{i} \delta_k$. The earliest time when the vehicle may reach f_i, assuming it starts heading towards it at t_i, is τ_i.

2.3. Performance and competitive ratio

The *performance* λ^I_{ALG} of an algorithm ALG for an instance $I \in \mathcal{I}_{n,T}$ of `OPTiWinD` is the ratio of the sum of the weights of the served requests by the total sum of the weights of all requests. The performance of ALG for at most n requests and delay T is then defined as:

$$\lambda^{\mathcal{I}_{n,T}}_{ALG} = \inf_{I \in \mathcal{I}_{n,T}} \lambda^I_{ALG}.$$

By definition, any λ^I_{ALG} is in $[0,1]$; hence $\lambda^{\mathcal{I}_{n,T}}_{ALG} \in [0,1]$. An online algorithm ALG is γ-competitive for problem `OPTiWinD` with at most n requests and delay T if, given any instance $I \in \mathcal{I}_{n,T}$, the performance of the online algorithm is at least γ multiplied by the performance of OPT, the optimal offline algorithm: $\forall I \in \mathcal{I}_{n,T}$, $\lambda^I_{ALG} \geq \gamma \cdot \lambda^I_{OPT}$, with $\gamma > 0$. Note that this definition of competitive ratio is standard for maximisation problems. Hence, we have $0 \leq \gamma \leq 1$. The $c$ompetitive ratio of ALG for at most n requests and delay T is defined by:

$$\gamma^{\mathcal{I}_{n,T}}_{ALG} = \inf_{I \in \mathcal{I}_{n,T}} \frac{\lambda^I_{ALG}}{\lambda^I_{OPT}}.$$

For any algorithm ALG, we have $\lambda^{\mathcal{I}_{n,T}}_{ALG} \leq \gamma^{\mathcal{I}_{n,T}}_{ALG}$. In order to evaluate the performance and competitive ratio of an algorithm, we need to consider a worst case scenario. This situation corresponds to having a malicious adversary deciding when the requests are released and choosing their weights. While our problem is a one-player online game, as the adversary is not technically a player, it is standard practice to discuss strategies of the adversary as though in a two-player game.

The *optimal performance* and the *optimal competitive ratio* are respectively:

$$\lambda^{\mathcal{I}_{n,T}} = \sup_{ALG} \lambda^{\mathcal{I}_{n,T}}_{ALG} \quad \text{and} \quad \gamma^{\mathcal{I}_{n,T}} = \sup_{ALG} \gamma^{\mathcal{I}_{n,T}}_{ALG}$$

An online algorithm will be called *optimal* if its competitive ratio is equal to the optimal competitive ratio. Note that we have:

$$\lambda^{\mathcal{I}_{n,T}} \leq \gamma^{\mathcal{I}_{n,T}}.$$

Definition 1. *Given a function $f : \mathbb{N} \to \mathbb{R}^+$, we say that an online algorithm ALG is $f(n)$-performant (resp. $f(n)$-competitive) if it guarantees a performance (resp. competitive ratio) of $f(n)$ for instances with n requests. Hence:*

$$\lambda^{\mathcal{I}_{n,T}}_{ALG} \geq f(n) \quad (\text{resp. } \gamma^{\mathcal{I}_{n,T}}_{ALG} \geq f(n)).$$

Note that in Definition 1, the algorithm functions without knowing ahead of time the total number of requests that will appear.

Proposition 1. *The functions $\lambda^{\mathcal{I}_{n,T}}$ and $\gamma^{\mathcal{I}_{n,T}}$ are non-decreasing with respect to T and non-increasing with respect to n.*

Proof. Increasing T restricts the strategy of the adversary without affecting the player's algorithm. Similarly, increasing n increases the possibilities for the adversary. More precisely, if $T \leq T'$, then $\mathcal{I}_{n,T'} \subset \mathcal{I}_{n,T}$ and if $n \leq n'$, then $\mathcal{I}_{n,T} \subset \mathcal{I}_{n',T}$. □

Remark 1. *In order to find bounds to performances and competitive ratios, we may restrict our analysis by omitting inefficient algorithms. It is inefficient for the vehicle to slow down or to change directions when no request appears. Thus we will assume that the vehicle always moves at speed 1 and with a constant direction between the release times t_i and t_{i+1} of two successive requests.*

Remark 2. *Multiplying all the weights δ_i by a positive constant does not affect the game. Thus we will always suppose that the weight of the first request is $\delta_1 = 1$, except in the proof of Theorem 2.*

2.4. Summary of results

The optimal performance and competitive ratio are both decreasing with the number of requests n. For a fixed n, they are non-decreasing step functions with regards to the delay T. We distinguish three cases corresponding to small, large or medium delays. The following results were obtained for the case where X is the segment $[-1, 1]$.

- When there is no delay, there is a natural greedy algorithm for the vehicle which consists in always heading towards the request of greatest weight. This greedy algorithm is optimal, guaranteeing a performance and a competitive ratio of $\frac{1}{n}$. Small delays refer to values of T small enough that this greedy algorithm remains optimal. This means $T < T_0 = \frac{1}{2^{n-3}+1}$ in terms of performance and $T < T_1 = \frac{1}{2^{n-1}-2}$ in terms of competitive ratio.
- With a delay T greater or equal to 1, a more refined greedy algorithm, defined in Section 4, becomes optimal. The values of the optimal performances and optimal competitive ratios are then defined by a sequence (α_n) which is analysed in the appendix.
- The intermediate case is the most complex and remains mostly open. When the delay T reaches the critical value T_0 (resp. T_1), the performance (resp. competitive ratio) increases by at least $\epsilon = \frac{1}{n(n-1)(n+3)}$. For $\frac{1}{2} \leq T < 1$, the performance is characterised by a sequence (β_n), which is difficult to compute beyond the first few terms.

These general results are summarised in Table 1. In sections 3 to 6, we study successively the performances and competitive ratios for no delay, large delays, small delays and medium delays on the segment $[-1, 1]$. Section 7 deals with values of $n \leq 4$, whence we are able to calculate explicitly the performance and competitive ratios for all the possible values of the delay. Results for $n = 3$ and $n = 4$ are summarised in Tables 2 and 3. In Section 8, we generalise previous results to the case of centred geodesic metric spaces of diameter 2 (indicated

Delay	Performance	Competitive Ratio
$T < T_1$	$\frac{1}{n}$ (**)	$\frac{1}{n}$ (**)
$T_1 \leq T < T_0$	$\frac{1}{n}$ (**)	$> \frac{1}{n}$
$T_0 \leq T < \frac{1}{2}$	$\geq \frac{1}{n} + \epsilon$	$\geq \frac{1}{n} + \epsilon$
$\frac{1}{2} \leq T < 1$	β_n (*)	$\geq \beta_n$ (*)
$1 \leq T < 2 - \frac{1}{n-1}$	$\alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$ (*)	$\alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$ (*)
$2 - \frac{1}{n-1} \leq T$	1 (*)	1 (*)

Table 1: Performances and Competitive Ratios with $T_0 = \frac{1}{2^{n-3}+1}$, $T_1 = \frac{1}{2^{n-1}-2}$, $\epsilon = \frac{1}{n(n-1)(n+3)}$. The results marked with an asterisk (resp. a double asterisk) can be generalised to centred geodesic metric spaces (resp. geodesic metric spaces) of diameter 2.

with an asterisk in Table 1) or more generally to geodesic metric spaces of diameter 2 (indicated with a double asterisk). *Centred* geodesic metric spaces of diameter 2 are spaces included in $\overline{B}(O, 1)$, where $\overline{B}(O, 1)$ is the closed ball of radius 1 centred in O.

Delay	Performance	Competitive Ratio
$T < 1/2$	$1/3$	$1/3$
$1/2 \leq T < 1$	$1/\varphi^2$	$1/\varphi^2$
$1 \leq T < 1.5$	$\frac{1}{2}$	$\frac{1}{2}$
$1.5 \leq T$	1	1

Table 2: Performances and Competitive Ratios for 3 requests ($n = 3$)

Delay	Performance	Competitive Ratio
$T < 1/6$	$1/4$	$1/4$
$1/6 \leq T < 1/5$		0.2578
$1/5 \leq T < 1/4$		$2 - \sqrt{3}$
$1/4 \leq T < 1/3$		0.2803
$1/3 \leq T < 1/2$	$2 - \sqrt{3}$	$1 - \sqrt{2}/2$
$1/2 \leq T < 1$	$1 - \sqrt{2}/2$	0.3177
$1 \leq T < 1.5$	$1/\varphi^2$	$1/\varphi^2$
$1.5 \leq T < 5/3$	$1/2$	$1/2$
$5/3 \leq T$	1	1

Table 3: Performances and Competitive Ratios for 4 requests ($n = 4$)

3. `OPTiWinD` with no Delay

In this section, we will show that a greedy algorithm is optimal when there is no delay, and that no online algorithm guarantees a constant competitive ratio.

We consider a greedy algorithm for the vehicle, denoted by GR_0, in which the vehicle always moves towards the request of greatest weight. Thus, whenever a new request appears with a greater weight, the vehicle ignores his previous destination to go towards the new one.

Lemma 1. *The algorithm GR_0 is $\frac{1}{n}$-performant; i.e. $\lambda_{GR_0}^{\mathcal{I}_{n,0}} = \frac{1}{n}$.*

Proof. By applying GR_0, the vehicle is guaranteed to serve the request of greatest weight. This is sufficient to ensure a performance at least equal to $1/n$. □

Note that it follows from Lemma 1 that GR_0 is $\frac{1}{n}$-competitive.

Theorem 1. *For instances of $\mathcal{I}_{n,0}$, the algorithm GR_0 is optimal; i.e. $\gamma^{\mathcal{I}_{n,0}} = \frac{1}{n}$*

Proof. Let us consider an online algorithm ALG for the vehicle. We will describe a strategy of the adversary which limits the competitive ratio of ALG to $\frac{1}{n}$. The adversary first releases a request $f_1 = (-1, [1, 3], 1)$. If the vehicle does not go towards -1, the adversary will not release any more requests, and the performance will be 0. Thus, we may assume that the vehicle will start moving towards -1 at time 1. The strategy of the adversary is as follows: let τ_1 be the estimated time arrival (ETA) of the vehicle, i.e. $\tau_1 = 2$. While the vehicle maintains his course and fewer than n requests have been released, the adversary releases requests

$$f_{1,k} = (1, [\tau_1 - \frac{1}{4.3^k.n}, \tau_1 - \frac{1}{4.3^k.n} + 2], 1)\,,\ k \geq 0.$$

If at time $\tau_1 - \frac{1}{4.3^k.n}$, the vehicle changes directions to go towards 1, the adversary sets τ_2 as the new ETA in 1. He then repeats the process, with

$$f_{i,k} = ((-1)^{i+1}, [\tau_i - \frac{1}{4.3^k.n}, \tau_i - \frac{1}{4.3^k.n} + 2], 1)$$

where τ_i is the ETA in $(-1)^i$ after the vehicle's i-th change of directions, until n requests have been released (see Figure 1 for an illustration). In this manner, a total of exactly n requests will be released. Also, when request $f_{i,k}$ is released, if $k > 0$, it is already too late for the vehicle to reach $f_{i,k-1}$. Thus the vehicle will serve only one request. Note that $2i - \frac{i}{2n} \leq \tau_i \leq 2i$. Hence, $2i - \frac{1}{2} \leq \tau_i \leq 2i$. So $2i + 1 \in [\tau_i - \frac{1}{4.3^k.n}, \tau_i - \frac{1}{2.3^k.n} + 2]$. Hence, an optimal offline algorithm will serve all the requests by starting at $t = 0$ towards -1 and reaching $(-1)^{i+1}$ at time $2i + 1$. Thus, no algorithm can guarantee a competitive ratio greater than $\frac{1}{n}$. □

Remark 3. *When a new request f_i is released at time t_i, if τ_i denotes the ETA for reaching f_i, then, given that the vehicle moves at speed 1, the distance to f_i at t_i is $\tau_i - t_i$.*

Also, if the vehicle moves away from f_i towards f_k at time t_i, when the vehicle has passed half the distance to f_k, it is too late to serve f_i. Thus, if a request f_{i+1} is released at the same end as f_i, it is impossible to serve both f_i and f_{i+1} if and only if $\tau_k - t_{i+1} < \frac{1}{2}(\tau_k - t_i)$.

Remark 4. *Note that Theorem 1 implies that there is no algorithm with a constant competitive ratio for `OPTiWinD` with zero delay.*

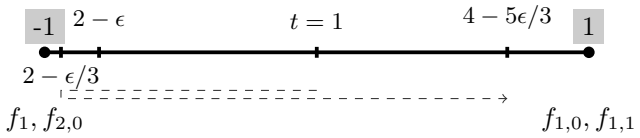

Figure 1: Initial movements of the vehicle in the case where it changes direction toward $f_{1,1}$ at time $2-\epsilon/3$. Using $\epsilon = 1/4n$, the figure displays the vehicle's position at each release time

.

4. OPTiWinD with Large Delays

In this section, we will show that introducing a large delay significantly improves the performance and competitive ratio. The result shown in Theorem 1 indicates that the previous case is extremely unfavorable to the vehicle. However, this changes when we impose a delay between the release times of successive requests. With a delay sufficiently large, the strategy of the adversary used in the proof of Theorem 1 is no longer possible. Indeed, this strategy required that requests be released in rapid succession in -1 and 1.

Let us note that if $T \geq 2$ then GR_0 will serve all the requests. So, the only interesting case is when $T < 2$. For $n \geq 1$, we define:

$$\alpha_n = \inf_{\delta \in (\mathbb{R}_+^*)^n} \max\{\frac{\delta_1}{S_2}, \cdots, \frac{\delta_i}{S_{i+1}}, \cdots, \frac{\delta_{n-1}}{S_n}, \frac{\delta_n}{S_n}\}$$

where $S_i = \sum_{k=1}^{i} \delta_k$. (See Appendix for the study of (α_n).)

Theorem 2. *For $1 \leq T \leq 2 - \frac{1}{n-1}$, we have $\lambda^{\mathcal{I}_{n,T}} = \gamma^{\mathcal{I}_{n,T}} = \alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$ and for $T \geq 2 - \frac{1}{n-1}$, $\lambda^{\mathcal{I}_{n,T}} = \gamma^{\mathcal{I}_{n,T}} = 1$.*

Proof. First, we will give a strategy of the adversary with a parameter $\epsilon > 0$ which limits the competitive ratio to $\alpha_{n-\lfloor \frac{1}{2-T} \rfloor} + O(\epsilon)$, when $\epsilon \to 0$. Then, we will give a greedy algorithm for the vehicle which guarantees a performance of $\alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$.

1. Let $\epsilon > 0$. The strategy of the adversary is defined as follows:
   (a) First, release a request $f_1 = (-1, [1, 3], \epsilon)$.
   (b) If the vehicle does not serve f_1, do not release any more requests.
   (c) Similarly, while the vehicle serves all the requests up to f_{i-1} and $i \leq i_0 = \lfloor \frac{1}{2-T} \rfloor$, release a request
   $$f_i = ((-1)^i, [1 + (i-1)T, 3 + (i-1)T], 4^{i-1}\epsilon).$$
   (d) If the vehicle serves f_{i_0}, release a request
   $$f_{i_0+1} = ((-1)^{i_0+1}, [1 + i_0 T, 3 + i_0 T], 1).$$

(e) Then, while the vehicle does not serve any more requests and $i \leq n$, release a request

$$f_i = ((-1)^i, [2i - 2 - \eta_i, 2i - \eta_i], \delta_{i-i_0})$$

where (δ_i) realises $\alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$ (see appendix) and $\eta_i = \frac{i-i_0}{n} - \frac{3}{2n}$.

(f) When the vehicle serves a request f_i, $i > i_0$, do not release any more requests.

Note that $\forall i \leq i_0 + 1$, $2i - 1 \in [1 + (i-1)T, 3 + (i-1)T]$. Since $i_0 \geq 1$, we have $\eta_i \leq \eta_n \leq 1 - \frac{5}{2n} < 1$. So, $\forall i \geq i_0 + 2$, $2i - 1 \in [2i - 2 - \eta_i, 2i - \eta_i]$. Therefore, the optimal offline algorithm will serve all the requests by being in $(-1)^i$ at time $2i - 1$. Also, the values of η_i have been chosen so that for $i \geq i_0 + 2$, $t_i = \tau_{i-1} - \frac{1}{2n}$.

Let us now consider an online algorithm, ALG. When the adversary uses the above strategy, if ALG does not allow the vehicle to serve f_i for $i \leq i_0$, then the performance is $\frac{\sum_{k=1}^{i-1} 4^{k-1}\epsilon}{\sum_{k=1}^{i} 4^{k-1}\epsilon} = \frac{4^{i-1}-1}{4^i-1} < \frac{1}{4}$. It is shown in Proposition 8 in the appendix that $\frac{1}{4} < \alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$.

If ALG allows the vehicle to serve the first i_0 requests, the vehicle will then reach at most one f_i with $i > i_0$. Its performance will then be

$$\begin{aligned} \frac{\delta_{i-i_0} + \sum_{k=1}^{i_0} 4^{k-1}\epsilon}{\sum_{k=1}^{i-i_0+1} \delta_k + \sum_{k=1}^{i_0} 4^{k-1}\epsilon} &= \frac{\delta_{i-i_0}}{S_{i-i_0+1}} + O(\epsilon) \\ &= \alpha_{n-\lfloor \frac{1}{2-T} \rfloor} + O(\epsilon). \end{aligned}$$

Therefore, $\gamma^{\mathcal{I}_{n,T}} \leq \alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$.

2. The greedy algorithm GR_1 is defined as follows:

(a) When there is no request to be served, or none that can be reached in time, head towards 0.

(b) While a single request that can be served is ongoing, head towards it.

(c) When a request $f_2 = (x_2, [t_2, t_2 + 2], \delta_2)$ is released while another request $f_1 = (x_1, [t_1, t_1+2], \delta_1)$ is still reachable, if $\frac{\delta_1}{\delta_1+\delta_2} \geq \alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$, head towards f_1; else head towards f_2. From this point on, until a request is served, number the requests f_1 to f_d and denote by δ_i the weight of f_i and $S_i = \sum_{k=1}^{i} \delta_k$. When f_{i+1} is released, if $\frac{\delta_i}{S_{i+1}} \geq \alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$, head towards f_i; else, head towards f_{i+1}.

We will divide the game into phases according to when the vehicle is either in cases a) and b) or in case c).

In the phases corresponding to a) and b), the vehicle will serve all the requests, and the performance over those phases of the game will be 1.

If at some point during the game, a sequence of requests $f_1, \ldots, f_d$ places the vehicle in case c), we will show that the performance will be greater or equal to $\alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$ over this phase. With a delay of T and the vehicle applying algorithm GR_1, the first time case c) may occur is for the (i_0+1)-th and (i_0+2)-th requests, where $i_0 = \lfloor \frac{1}{2-T} \rfloor$. Hence, $d \leq n - i_0$. Note that when a request $f_i = (x_i, [t_i, t_i + 2], \delta_i)$ is served at time t, since $T \geq 1$ and the time windows have length 2, at most one request may have been released between t_i and t. It follows that when f_i is reached, the performance over that phase of the game is $\frac{\delta_i}{S_i}$ or $\frac{\delta_i}{S_{i+1}}$. By definition of (α_n), this performance is at least α_{n-i_0}.

Hence, in all phases of the game, the performance is at least $\alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$. Therefore, $\lambda_{ALG}^{\mathcal{I}_{n,T}} \geq \alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$.

It follows from 1. and 2. that we have $\lambda^{\mathcal{I}_{n,T}} = \gamma^{\mathcal{I}_{n,T}} = \alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$ for $1 \leq T \leq 2 - \frac{1}{n-1}$.

Hence, $\lambda^{\mathcal{I}_{n,2-\frac{1}{n-1}}} = \gamma^{\mathcal{I}_{n,2-\frac{1}{n-1}}} = \alpha_1 = 1$. Therefore, for $T \geq 2 - \frac{1}{n-1}$, we have $\lambda^{\mathcal{I}_{n,T}} = \gamma^{\mathcal{I}_{n,T}} = 1$.

□

5. OPTiWinD with Small Delays

In this section, we will consider cases where T is small enough that the results from Section 3 still apply.

Theorem 3. *For $T < T_0 = \frac{1}{2^{n-3}+1}$, we have $\lambda^{\mathcal{I}_{n,T}} = \frac{1}{n}$.*

Proof. When the vehicle has to choose between two symmetrically placed requests of equal weights and cannot serve both, it is optimal to go towards the closest one. We will restrict our description of the strategy of the adversary by assuming that the vehicle follows this rule.

The strategy of the adversary is to release requests

$$f_i = (-1, [t_i, t_i + 2], 1)$$

where $t_1 = 1$, $t_i = 2 - \frac{1-T}{2^{i-2}} + \frac{1-T}{2^{n-2}} - \frac{T}{2}$ for $2 \leq i \leq n-1$, and $t_n = t_{n-1} + T$. Note that $T < \frac{1}{2^{n-3}+1}$ implies $T < \frac{1-T}{2^{n-3}}$. Hence, $t_{i+1} - t_i \geq T$ for all $1 \leq i \leq n-1$.

Let us show by induction that if the player follows the rule given at the beginning of this proof, starting from $t = 1$, the vehicle will always move towards -1. This is true when the request f_1 appears. Let us assume that it moved

towards -1 until f_{i+1} appears. Thus, when f_{i+1} is released, the vehicle is located at $1 - t_{i+1}$. Hence, the distance between its current position and 1 is t_{i+1}. Note that for $2 \leq i \leq n-1$, we have $\frac{1}{2}(2-t_i) = \frac{1-T}{2^{i-1}} - \frac{1-T}{2^{n-1}} - \frac{T}{4} > 2-t_{i+1}$. Hence, $2t_{i+1} > t_i + 2$. Therefore, when f_{i+1} is released, it is too late to serve f_i. Thus, the vehicle will continue towards -1.

So, this strategy is well defined for optimal play. Also, the vehicle will only be able to serve a single request, and n requests will be released. Hence, the performance is at most $\frac{1}{n}$. Since the algorithm GR_0 guarantees at least $\frac{1}{n}$, we have an equality. □

Theorem 4. *For $n \geq 3$ and $T < T_1 = \frac{1}{2^{n-1}-2}$, the algorithm GR_0 is optimal and $\gamma^{\mathcal{I}_{n,T}} = \frac{1}{n}$.*

Proof. Let us consider an online algorithm ALG for the vehicle. We will describe a strategy of the adversary which limits the competitive ratio of ALG to $\frac{1}{n}$. The adversary first releases a request $f_1 = (-1, [1, 3], 1)$. If the vehicle does not go towards -1, the adversary will not release any more requests, and the performance will be 0. Thus, we may assume that the vehicle will start moving towards -1 at time 1. The strategy of the adversary is as follows:

1. let τ_1 be the estimated time arrival (ETA) of the vehicle in -1, i.e. $\tau_1 = 2$.
2. While the vehicle maintains its course and fewer than $n-1$ requests have been released, the adversary releases requests

   $$f_{1,k} = (1, [\tau_1 - 2^{n-3-k}(T + 2\epsilon) + \epsilon, \tau_1 - 2^{n-3-k}(T + 2\epsilon) + \epsilon + 2], 1)$$

   for $0 \leq k \leq n-3$ and some $0 < \epsilon < T$.
3. If at time $\tau_1 - 2^{n-3-k}(T+2\epsilon)+\epsilon$, the vehicle chooses to change directions to go towards 1, the adversary sets τ_2 as the new ETA in 1.
4. He then repeats the process, with

   $$f_{i,k} = ((-1)^{i+1}, [\tau_i - 2^{n-2-i-k}(T+2\epsilon)+\epsilon, \tau_i - 2^{n-2-i-k}(T+2\epsilon)+\epsilon+2], 1)$$

   where τ_i is the ETA in $(-1)^i$ after the vehicle's i-th change of directions, until $n-1$ requests have been released.

In this manner, a total of exactly $n-1$ requests will be released. If the vehicle changed directions $j-1$ times in total, the n-th request will be

$$f_n = ((-1)^{j+1}, [\tau_j - \epsilon, \tau_j - \epsilon + 2], 1).$$

We will show that a) this strategy satisfies the condition regarding the delay T, b) the vehicle can only serve a single request and c) for ϵ small enough the optimal offline algorithm can serve all the requests.

a) Note that the delay between $f_{i,k}$ and $f_{i,k+1}$ is greater than T (even $T+2\epsilon$), and the delay between f_n and $f_{j,k}$, for the last value of k, is at least T.

b) When request $f_{i,k}$ is released, if $k > 0$, it is already too late for the vehicle to reach $f_{i,k-1}$. Similarly, when f_n is released, it is too late to serve $f_{j,k}$. Thus the vehicle will serve only one request.

c) If the vehicle changes direction when $f_{i,0}$ is released, the new ETA in $(-1)^{i+1}$ is $\tau_i + 2 - 2^{n-1-i}(T + 2\epsilon) + 2\epsilon$. Hence,

$$\tau_{i+1} \geq \tau_i + 2 - 2^{n-1-i}(T + 2\epsilon) + 2\epsilon = \tau_i + 2 - 2^{n-1-i}T + O(\epsilon).$$

Therefore, we have $\tau_i \geq 2i - (2^{n-1} - 2^{n-i})T + O(\epsilon)$. Thus, the time window of $f_{i,k}$ closes at

$$\tau_i - 2^{n-2-i-k}T + 2 + O(\epsilon) \geq 2i + 2 - (2^{n-1} - 2^{n-i} - 2^{n-2-i-k})T + O(\epsilon).$$

Since $2^{n-i} + 2^{n-2-i-k} > 2$, we have $(2^{n-1} - 2^{n-i} - 2^{n-2-i-k})T < 1$. Hence, for ϵ small enough, $2i + 1$ is in the time window of $f_{i,k}$. Similarly, the time window of f_n closes at

$$\tau_j - \epsilon + 2 + O(\epsilon) \geq 2j + 2 - (2^{n-1} - 2^{n-j})T + O(\epsilon) \geq 2j + 2 - (2^{n-1} - 2)T + O(\epsilon).$$

Hence, for ϵ small enough, $2j + 1$ is in the time window of f_n. Therefore, the optimal offline algorithm will serve all the requests by starting at $t = 0$ towards -1 and reaching $(-1)^{i+1}$ at time $2i + 1$.

Thus no algorithm can obtain a competitive ratio greater than $\frac{1}{n}$. □

6. `OPTiWinD` with Medium Delays

6.1. Tightness of the bounds T_0 and T_1

In this section, we will show that when the delay T is at least T_0 (resp. T_1), the optimal performance (resp. competitive ratio) is greater than $\frac{1}{n}$, meaning that the boundaries T_0 and T_1 are tight.

In order to show that the boundary T_0 is tight, we will use the following lemma, inspired by the proof of Theorem 3.

Lemma 2. *Let $T \geq \frac{1}{2^{n-3}+1}$. Starting at time 1, if the vehicle, initially at 0, always moves towards the request $f_1 = (-1, [1, 3], \delta_1)$, and if $n-1$ other requests are released before time 2, there will be a moment when the vehicle has the possibility of serving two requests.*

Proof. Since the vehicle is moving towards -1, we may assume that the other $n - 1$ requests are all located in 1; thus we will use the same notations as in the proof of Theorem 3. The earliest possible release time for f_2 is $1 + T$. For $2 \leq i \leq n - 1$,

- if $2 - t_{i+1} \geq \frac{1}{2}(2 - t_i)$, it follows from Remark 3 that it is possible for the vehicle to serve both f_i and f_{i+1}.

- if $2 - t_{i+1} < \frac{1}{2}(2 - t_i)$, then $t_{n-1} > 2 - T$. Thus, $t_n > 2$; and it is possible to serve both f_1 and f_n.

□

Theorem 5. *For $T_0 = \frac{1}{2^{n-3}+1}$, we have $\lambda^{\mathcal{I}_{n,T_0}} \geq \frac{1}{n} + \epsilon$, where $\epsilon = \frac{1}{n(n-1)(n+3)}$.*

Proof. We define the algorithm $Al1$ as follows:

(a) When the first request f_1 is released, head towards it. We may assume that $f_1 = (-1, [1, 3], 1)$.

(b) While no weight is greater than $\kappa = \frac{1+n\epsilon}{1-n(n-1)\epsilon}$ and the vehicle does not have the possibility of serving two requests, keep going towards f_1.

(c) If a request f_i appears with weight $\delta_i > \kappa$, head towards it and behave like GR_0 from this point onwards.

(d) If at some point the vehicle has the possibility of serving two requests, combine their weights and behave like GR_0.

We will show that $\lambda_{Al1}^{\mathcal{I}_{n,T_0}} \geq \frac{1}{n} + \epsilon$.

- Case (b): If no weight is greater than κ and the vehicle never has the possibility of serving two requests, it will reach f_1 at time 2. It follows from Lemma 2 that at most $n-1$ requests have been released at this point. Therefore, the vehicle's performance is at least $\frac{1}{\kappa(n-1)}$. Note that with $\epsilon = \frac{1}{n(n-1)(n+3)}$, we have

$$\frac{1}{\kappa(n-1)} = \frac{1 - n(n-1)\epsilon}{1 + n\epsilon} \frac{1}{n-1} = \frac{1}{n} \frac{n^2 + 2n}{n^2 + 2n - 2} > \frac{1}{n} \frac{n^2 + 2n - 2}{n^2 + 2n - 3} = \frac{1}{n} + \epsilon.$$

- Case (c): If $\delta_1 = 1$ and $\delta_i > \frac{1+n\epsilon}{1-n(n-1)\epsilon}$, we cannot have both $\max(\delta_i) \leq (\frac{1}{n} + \epsilon)S_n$ and $\min(\delta_i) \geq (\frac{1}{n} - (n-1)\epsilon)S_n$. And since $\max(\delta_i) \leq (\frac{1}{n} + \epsilon)S_n$ implies $\min(\delta_i) \geq (\frac{1}{n} - (n-1)\epsilon)S_n$, we have: $\frac{\max(\delta_i)}{S_n} > \frac{1}{n} + \epsilon$.

- Case (d): Let us assume that, at time t_j, the vehicle becomes able to serve two requests of weights δ_i and δ_j. Let $\Delta = \{1, \ldots, n+1\} - \{i, j\}$ and let $\delta_{n+1} = \delta_i + \delta_j$. Let $J = \{1 \leq k \leq j | f_k \text{ is no longer reachable at time } t_j\}$ and let $\delta = \max_{k \in \Delta - J} \delta_k$. We define $S_\Delta = \sum_{k \in \Delta} \delta_k$ and S_J similarly. Note that when the possibility of serving two requests appears at time t_j, the request f_1 is still reachable. Hence, $1 \notin J$ and $\delta \geq 1$. Also, since we are in case (d), $\forall k \in J$, $\delta_k \leq \kappa$. By behaving like GR_0, the vehicle is sure to serve a request of weight at least δ. As $\kappa > 1$, we have:

$$S_\Delta \leq (n - 1 - |J|)\delta \leq \kappa\delta(n - 1 - |J|) \text{ and } S_J \leq \kappa|J| \leq \kappa\delta|J|.$$

Hence, the vehicle's optimal performance is at least equal to:

$$\frac{\delta}{S_n} = \frac{\delta}{S_\Delta + S_J} \geq \frac{1}{\kappa(n-1)} > \frac{1}{n} + \epsilon.$$

□

Corollary 1. *We have* $\lambda^{\mathcal{I}_{n,T}} = \frac{1}{n}$ *if and only if* $T < \frac{1}{2^{n-3}+1}$.

Proof. This follows directly from Theorem 3, Theorem 5 and the fact that $\lambda^{\mathcal{I}_{n,T}}$ increases with T. □

In order to prove that the bound given in Theorem 4 is tight, we define the algorithm $Al2$ as follows:

(a) When f_1 is released at $t_1 = 1$ in -1, go towards -1.

(b) When f_2 is released at t_2 in 1, if $t_2 > 2 - 2^{n-3}T_1$, keep going towards -1 either until two requests can be served by changing direction, or until f_1 is reached; else go towards 1 and set τ_2 as the ETA in 1.

(c) Similarly, while the vehicle has changed direction with each new request, when f_i is released at t_i in $(-1)^i$, if $t_i > \tau_{i-1} - 2^{n-1-i}T_1$, keep going towards $(-1)^{i-1}$ either until two requests can be reached by changing direction, or until f_{i-1} is reached; else change direction towards $(-1)^i$ and set τ_i as the ETA in $(-1)^i$.

Lemma 3. *For* $n \geq 3$ *and given a delay* $T \geq T_1 = \frac{1}{2^{n-1}-2}$*, the algorithm* $Al2$ *guarantees that if* n *requests are released, either the vehicle can serve at least two requests or no optimal offline algorithm can serve all requests.*

Proof. We may assume that the first request released by the adversary is $f_1 = (-1, [1, 3], 1)$ and the vehicle applies $Al2$. Assuming it exists, let i denote the smallest index for which $t_i > \tau_{i-1} - 2^{n-1-i}T$. The vehicle will keep going towards $(-1)^{i-1}$. For $i \leq k \leq n-1$, if $\tau_{i-1} - t_{k+1} \geq \frac{1}{2}(\tau_{i-1} - t_k)$, then it follows from Remark 3 that the vehicle can serve both f_k and f_{k+1}. However, if $\tau_{i-1} - t_{k+1} < \frac{1}{2}(\tau_{i-1} - t_k)$, for all $i \leq k \leq n-1$, then $\tau_{i-1} - t_{n-1} < T$ so the n-th request cannot be released before f_{i-1} is reached.

Since $\forall i \geq 2$, $\tau_i = 2t_i - \tau_{i-1} + 2$, if $\forall 2 \leq i \leq n-1$, $t_i \leq \tau_{i-1} - 2^{n-1-i}T$, then

$$\tau_{n-1} \leq \tau_1 + 2(n-2) - \sum_{i=1}^{n-2} 2^i T \leq \tau_1 + 2(n-2) - \sum_{i=1}^{n-2} 2^i T_1 = 2n - 3.$$

So if $t_n < \tau_{n-1}$, then $2n - 1 \notin [t_n, t_n + 2]$. Hence, either the vehicle can serve two requests, or the optimal offline algorithm cannot serve all the requests. □

Theorem 6. *For* $T_1 = \frac{1}{2^{n-1}-2}$*, we have* $\gamma^{\mathcal{I}_{n,T_1}} > \frac{1}{n}$.

Proof. Let $0 < \epsilon < \frac{1}{n^2}$, $\kappa = \frac{1+n\epsilon}{1-n(n-1)\epsilon}$ and let the sequence (ω_i) be defined by $\omega_1 = 1$ and $\omega_{i+1} = \omega_i(1 - n^2\epsilon) - (i-2)(\kappa - \omega_i)$. We define the algorithm $Al3$ as follows:

(a) When $f_1 = (-1, [1, 3], 1)$ is released and while each request f_i has a weight satisfying $\omega_i \leq \delta_i \leq \kappa$ and no two requests can be served, apply $Al2$.

(b) If the request f_i has a weight $\delta_i > \kappa$ or $\delta_i < \omega_i$, apply GR_0.

(c) If two requests can be served, combine them and apply GR_0.

We will show that for ϵ small enough, $Al3$ guarantees a competitive-ratio of $\frac{1}{n}+\epsilon$.

We can show by induction that $\omega_i \leq 1$ and $\omega_i = 1+O(\epsilon)$. Since $\kappa > 1 \geq \omega_i$, we have (ω_i) is decreasing. Note also that $\kappa = 1+O(\epsilon)$, so $\frac{\omega_i}{\kappa} = 1+O(\epsilon)$.

- Case (a): Assume that for all i, $\omega_i \leq \delta_i \leq \kappa$ and no two requests can ever be served. Then, the vehicle applies $Al2$ until the end. It follows from Lemma 3 that the optimal offline algorithm will not be able to serve n requests. Thus the competitive ratio in this case is at least $\frac{\omega_n}{(n-1)\kappa} = \frac{1}{n-1}+O(\epsilon)$. Hence, for ϵ small enough, this competitive ratio is greater than $\frac{1}{n}+\epsilon$.

- Case (b.1): Assume now that the vehicle switches from $Al2$ to GR_0 when a request f_i is released with $\delta_i > \kappa$. This case is identical to Case (c) in the proof of Theorem 5. Thus, the performance is at least $\frac{1}{n}+\epsilon$.

- Case (b.2): Assume now that the vehicle switches from $Al2$ to GR_0 when a request f_i has a weight $\delta_i < \omega_i$. Let j denote the index of the request towards which the vehicle was heading when f_i was released. Let $\delta = \max\{\delta_k, k > i\} \cup \{\delta_j\}$. The vehicle will at least serve a request of weight δ, with $\delta \geq \omega_{i-1}$ and $\delta_i < \omega_i$. The sum of the weights is at most $(n-i+1)\delta + \delta_i + (i-2)\kappa$. So the competitive ratio is at least

$$\begin{aligned}
\frac{\delta}{(n-i+1)\delta+\delta_i+(i-2)\kappa} &\geq \frac{\omega_{i-1}}{(n-i+1)\omega_{i-1}+\omega_i+(i-2)\kappa} \\
&\geq \frac{\omega_{i-1}}{(n-i+1)\omega_{i-1}+\omega_{i-1}(1-n^2\epsilon)+(i-2)\omega_{i-1}} \\
&\geq \frac{1}{n-n^2\epsilon} \\
&\geq \frac{1}{n}+\epsilon.
\end{aligned}$$

- Case (c): Assume now that the vehicle switches from $Al2$ to GR_0 when two requests can be served. This case is similar to Case (d) of the proof of Theorem 5. The only difference is that we no longer have $\delta \geq 1$. Using the same notations, we have instead $\delta \geq \omega_{j-1}$. Thus the performance is at least $\frac{\omega_{j-1}}{\kappa(n-1)}$. For ϵ small enough, this is greater than $\frac{1}{n}+\epsilon$.

Hence, for ϵ small enough, $Al3$ guarantees a competitive ratio of $\frac{1}{n}+\epsilon$. □

We close this section by showing that the optimal performance for $\frac{1}{2} \leq T < 1$ can be expressed using an induction formula. In order to solve the general case for $\frac{1}{2} \leq T < 1$, we introduce a variant problem in which the initial state is modified. Thus we define the *optimal* weighted performance $\lambda^{\mathcal{I}_{n,T},\delta_0}$ as the optimal performance obtained by an algorithm in the case where the vehicle's starting position at $t=0$ is somewhere in $]T-1, 1-T[$, requests with a total weight of $\delta_0 \geq 0$ have already been missed and the next request to appear is of weight 1. We will show that this is well defined as the optimal weighted performance does not depend on where in $]T-1, 1-T[$ the vehicle starts.

Theorem 7. *For $\frac{1}{2} \leq T < 1$, the optimal weighted performance $\lambda^{\mathcal{I}_{n,T},\delta_0}$ satisfies the following:*

- $\lambda^{\mathcal{I}_{1,T},\delta_0} = \frac{1}{1+\delta_0}$ *and* $\lambda^{\mathcal{I}_{2,T},\delta_0} = \frac{1}{2+\delta_0}$,
- $\forall n \geq 3$,

$$\begin{aligned}\lambda^{\mathcal{I}_{n,T},\delta_0} &= \inf_{\delta_2 \geq 0} \max\{\frac{1}{\delta_0+1+\delta_2}, \min\{\lambda^{\mathcal{I}_{n-2,T},\frac{\delta_0+1+\delta_2}{\delta_2}}, \\ &\quad \lambda^{\mathcal{I}_{n-1,T},\frac{\delta_0+1}{\delta_2}}\}\}.\end{aligned}$$

Proof. If $n = 1$, the vehicle will reach the request of weight 1, so its performance will be $\frac{1}{1+\delta_0}$. If $n = 2$, the vehicle will always be able to reach one of the two requests, and the worst case is when both requests are released with delay T at opposite ends and with equal weights. In that case, the vehicle's performance is $\frac{1}{2+\delta_0}$. Let us now consider $n \geq 3$. We may assume that the first request is $f_1 = (-1, [0, 2], 1)$ and that the vehicle will move towards it. After a delay of at least T and before f_1 is served, the adversary will release a request $f_2 = (1, [t_2, t_2+2], \delta_2)$. The vehicle will thus have two options. If it continues towards the first request, it will serve f_1 before a third request can be released and its performance will be $\frac{1}{\delta_0+1+\delta_2}$. If it changes direction towards f_2, it will have returned to the interval $]T-1, 1-T[$ at time t_2+T. While the vehicle remains in this interval, the adversary may release a request $f_3 = (-1, [t_3, t_3+2], \delta_3)$, and the vehicle will have to choose between f_2 and f_3. In this case, it follows from the symmetry of that situation that it is optimal for the adversary to select $\delta_3 = \delta_2$. If we divide all the weights by δ_2, the situation is identical to that of the problem defined above, with n replaced by $n-2$ and δ_0 replaced by $\frac{\delta_0+1+\delta_2}{\delta_2}$. While the vehicle is in the interval $]T-1, 1-T[$, the adversary may choose not to release a request, and similarly, in this case, n is replaced by $n-1$ and δ_0 is replaced by $\frac{\delta_0+1}{\delta_2}$. Hence,

$$\begin{aligned}\lambda^{\mathcal{I}_{n,T},\delta_0} &= \inf_{\delta_2 \geq 0} \max\{\frac{1}{\delta_0+1+\delta_2}, \min\{\lambda^{\mathcal{I}_{n-2,T},\frac{\delta_0+1+\delta_2}{\delta_2}}, \\ &\quad \lambda^{\mathcal{I}_{n-1,T},\frac{\delta_0+1}{\delta_2}}\}\}.\end{aligned}$$

□

Corollary 2. *For $\frac{1}{2} \leq T < 1$, we have $\lambda^{\mathcal{I}_{n,T}} = \beta_n$, where $\beta_1 = 1$, $\beta_2 = \frac{1}{2}$ and $\forall n \geq 3$,*

$$\beta_n = \inf_{\delta_2 \geq 0} \max\{\frac{1}{1+\delta_2}, \min\{\lambda^{\mathcal{I}_{n-2,T},\frac{1+\delta_2}{\delta_2}}, \lambda^{\mathcal{I}_{n-1,T},\frac{1}{\delta_2}}\}\}.$$

Proof. It follows from the definitions that the optimal performance $\lambda^{\mathcal{I}_{n,T}}$ is equal to the optimal weighted performance $\lambda^{\mathcal{I}_{n,T},0}$ with zero requests missed. □

Conjecture 1. *The formulas in Theorem 7 and Corollary 2 can be simplified as follows: for all $\frac{1}{2} \leq T < 1$ and $n \geq 3$,*

$$\lambda^{\mathcal{I}_{n,T},\delta_0} = \inf_{\delta_2 \geq 0} \max\{\frac{1}{\delta_0+1+\delta_2}, \lambda^{\mathcal{I}_{n-2,T},\frac{\delta_0+1+\delta_2}{\delta_2}}\}$$

$$\textit{and} \quad \lambda^{\mathcal{I}_{n,T}} = \inf_{\delta_2 \geq 0} \max\{\frac{1}{1+\delta_2}, \lambda^{\mathcal{I}_{n-2,T}, \frac{1+\delta_2}{\delta_2}}\}.$$

In the proof of Proposition 2 below, we verify that for $n = 3$ or 4, we have $\lambda^{\mathcal{I}_{n-2,T}, \frac{\delta_0+1+\delta_2}{\delta_2}} \leq \lambda^{\mathcal{I}_{n-1,T}, \frac{\delta_0+1}{\delta_2}}$. Using Proposition 2, we can easily verify that it is also true for $n = 5$. So the conjecture is true for small values of n. The conjecture states that it is inefficient for the adversary to pass on the possibility of releasing a request when the vehicle comes back near 0.

Table 4 summarises the results obtained so far:

Delay	Performance	Competitive Ratio
$T < T_1$	$\frac{1}{n}$	$\frac{1}{n}$
$T_1 \leq T < T_0$	$\frac{1}{n}$	$> \frac{1}{n}$
$T_0 \leq T < \frac{1}{2}$	$\geq \frac{1}{n} + \epsilon$	$\geq \frac{1}{n} + \epsilon$
$\frac{1}{2} \leq T < 1$	β_n	$\geq \beta_n$
$1 \leq T < 2 - \frac{1}{n-1}$	$\alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$	$\alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$
$2 - \frac{1}{n-1} \leq T$	1	1

Table 4: Performances and Competitive Ratios with $T_0 = \frac{1}{2^{n-3}+1}, T_1 = \frac{1}{2^{n-1}-2}$, $\epsilon = \frac{1}{n(n-1)(n+3)}$.

The performance remains unknown for $T_0 \leq T < \frac{1}{2}$, and the competitive ratio for $T_1 \leq T < 1$.

7. `OPTiWinD` with a total number of requests at most 4

In this section, we will give a complete description of what happens when the number of requests is at most 4.

When there is only one request, the vehicle can always reach it, so the performance and competitive ratio are equal to 1.

When the number of requests is at most two, the performance and competitive ratio are equal to $\frac{1}{2}$ if $T < 1$ and 1 otherwise.

For $n = 3$, the cases $T < \frac{1}{2}$ and $T \geq 1$ are covered by Theorem 4 and Theorem 2, respectively. The case $\frac{1}{2} \leq T < 1$ is given as an induction formula in Theorem 7. In the following proposition, we calculate the induction formula for $n = 3$ and $n = 4$.

Proposition 2. *For $\frac{1}{2} \leq T < 1$, we have $\lambda^{\mathcal{I}_{3,T},\delta_0} = \frac{2}{3+\delta_0+\sqrt{\delta_0^2+2\delta_0+5}}$ and $\lambda^{\mathcal{I}_{4,T},\delta_0} = \frac{2}{4+\delta_0+\sqrt{\delta_0^2+8}}$.*

Proof. Using the formulas given in Theorem 7, we obtain:

$$\lambda^{\mathcal{I}_{3,T},\delta_0} = \inf_{\delta_2 \geq 0} \max\Big\{\frac{1}{\delta_0+1+\delta_2}, \frac{1}{1+\frac{\delta_0+1+\delta_2}{\delta_2}}\Big\}.$$

In the inf max above, the first term is decreasing while the second is increasing in δ_2. Hence, the inf max is reached when there is equality.

$$\begin{aligned}\delta_0 + 1 + \delta_2 = 1 + \frac{\delta_0 + 1 + \delta_2}{\delta_2} \\ \Leftrightarrow \quad \delta_2^2 + (\delta_0 - 1)\delta_2 - (\delta_0 + 1) = 0 \\ \Leftrightarrow \quad \delta_2 = \frac{1 - \delta_0 + \sqrt{\delta_0^2 + 4\delta_0 + 5}}{2}\end{aligned}$$

Hence, $\lambda^{\mathcal{I}_{3,T},\delta_0} = \frac{2}{3+\delta_0+\sqrt{\delta_0^2+4\delta_0+5}}$.

It follows that for $\delta_0' = \frac{\delta_0+1}{\delta_2}$, we have

$$\begin{aligned}\lambda^{\mathcal{I}_{3,T},\delta_0'} &= \frac{2}{3 + \delta_0' + \sqrt{{\delta_0'}^2 + 4\delta_0' + 5}} \\ &< \frac{2}{3 + \delta_0' + \sqrt{(\delta_0' + 3)^2}} = \frac{1}{3 + \delta_0'} = \lambda^{\mathcal{I}_{2,T},1+\delta_0'}.\end{aligned}$$

Hence, $\lambda^{\mathcal{I}_{4,T},\delta_0} = \inf\limits_{\delta_2 \geq 0} \max\left\{\frac{1}{\delta_0+1+\delta_2}, \frac{1}{2+\frac{\delta_0+1+\delta_2}{\delta_2}}\right\}$.

Thus,

$$\begin{aligned}\delta_0 + 1 + \delta_2 = 2 + \frac{\delta_0 + 1 + \delta_2}{\delta_2} \\ \Leftrightarrow \quad \delta_2^2 + (\delta_0 - 2)\delta_2 - (\delta_0 + 1) = 0 \\ \Leftrightarrow \quad \delta_2 = \frac{2 - \delta_0 + \sqrt{\delta_0^2 + 8}}{2}.\end{aligned}$$

Hence, $\lambda^{\mathcal{I}_{4,T},\delta_0} = \frac{2}{4+\delta_0+\sqrt{\delta_0^2+8}}$. □

Corollary 3. *For $\frac{1}{2} \leq T < 1$, we have $\lambda^{\mathcal{I}_{3,T}} = \frac{1}{\varphi^2}$, where φ is the golden ratio, and $\lambda^{\mathcal{I}_{4,T}} = 1 - \frac{\sqrt{2}}{2}$.*

Proposition 3. *For $\frac{1}{2} \leq T < 1$, we have $\gamma^{\mathcal{I}_{3,T}} = \frac{1}{\varphi^2}$, where φ is the golden ratio.*

Proof. For $\frac{1}{2} \leq T < 1$ and $n = 3$, the adversary can limit the competitive ratio by using the following strategy:

(a) First, release a request $f_1 = (-1, [1, 3], 1)$.

(b) Assuming the vehicle moves towards f_1, release $f_2 = (1, [2 - \epsilon, 4 - \epsilon], \varphi)$, with $\epsilon < \min\{\frac{1}{3}, 1 - T\}$.

(c) If the vehicle keeps going towards f_1, do not release any more requests.

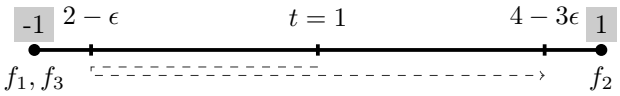

Figure 2: vehicle's itinerary for $n = 3$ and $\frac{1}{2} \leq T < 1$ in Proposition 3.

(d) Else, if the vehicle goes towards f_2, release $f_3 = (-1, [4-3\epsilon, 6-3\epsilon], \varphi)$. (see Figure 2)

Since $2i - 1 \in [t_i, t_i + 2]$, the offline algorithm can serve all the requests. If the vehicle chooses to keep going towards f_1 when f_2 is released, the competitive ratio will be $\frac{1}{1+\varphi}$. Else, if it changes directions, the competitive ratio will be $\frac{\varphi}{1+2\varphi} = \frac{1}{1+\varphi} = \frac{1}{\varphi^2}$. □

In Table 5, we summarise the results obtained for $n = 3$.

Delay	Performance	Competitive Ratio
$T < 1/2$	$1/3$	$1/3$
$1/2 \leq T < 1$	$1/\varphi^2$	$1/\varphi^2$
$1 \leq T < 1.5$	$\frac{1}{2}$	$\frac{1}{2}$
$1.5 \leq T$	1	1

Table 5: Performances and Competitive Ratios for $n = 3$

For $n = 4$, the cases $T < \frac{1}{6}$ and $T \geq 1$ are covered by Theorem 4 and Theorem 2, respectively. In the case where $\frac{1}{2} \leq T < 1$, the performance is given in Corollary 3.

Note that for $n = 4$ and $T < \frac{1}{6}$, the game can be represented as a decision tree shown in Figure 3. Assuming $\delta_1 = 1$, $\delta_4 = \delta_1$, $\delta'_4 = \delta_3$, $\delta''_4 = \delta'_3$ and $\delta'''_4 = \delta_2$, the optimal performance is

$$\lambda^{\mathcal{I}_{4,T}} = \min_{\delta_2} \max\{\min_{\delta_3} \max\{\frac{\delta_1}{S_4}, \frac{\delta_3}{S'_4}\}, \min_{\delta'_3} \max\{\frac{\delta'_3}{S''_4}, \frac{\delta_2}{S'''_4}\}\}.$$

Proposition 4. *For $\frac{1}{3} \leq T < \frac{1}{2}$, we have $\lambda^{\mathcal{I}_{4,T}} = 2 - \sqrt{3}$.*

Proof. For $\frac{1}{3} \leq T < \frac{1}{2}$, when the vehicle keeps moving towards -1, the adversary cannot release the fourth request before f_1 is served. This corresponds to replacing $\delta_4 = \delta_1$ in the case shown in Figure 3 with $\delta_4 = 0$. By setting $\frac{\delta'_3}{S''_4} = \frac{\delta_2}{S'''_4}$, we obtain $\delta'_3 = \delta_2$. Setting $\frac{\delta_1}{S_4} = \frac{\delta_3}{S'_4}$ with $\delta_4 = 0$ gives $\delta_3 = \frac{1-\delta_2}{2} + \frac{1}{2}\sqrt{\delta_2^2 + 2\delta_2 + 5}$. Then, setting $\frac{\delta_1}{S_4} = \frac{\delta_2}{S'''_4}$, we deduce that δ_2 is a root of $2X^3 - 3X - 1$. Hence, $\delta_2 = \frac{1+\sqrt{3}}{2}$, and $\lambda^{\mathcal{I}_{4,T}} = 2 - \sqrt{3}$.

The adversary can limit the performance to $2 - \sqrt{3}$ by using the following strategy with the optimal weights found above and $\epsilon = \frac{1}{2} - T$:

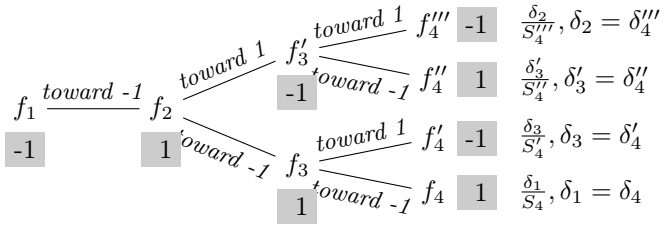

Figure 3: Decision tree for the vehicle's movements, $T < 1/6$ and at most 4 requests. The locations of the requests are given in the grey rectangles. The value of the competitive ratio and the condition on the weight of the fourth request relative to each branch are indicated on the right.

(a) First, release a request $f_1 = (-1, [1, 3], 1)$.

(b) Assuming the vehicle moves towards f_1, release $f_2 = (1, [2-T-\epsilon, 4-T-\epsilon], \delta_2)$.

(c) If the vehicle keeps going towards -1, release $f_3 = (1, [2-\epsilon, 4-\epsilon], \delta_3)$. If the vehicle keeps going towards -1, do not release a fourth request; else, if it changes directions, release a request $f_4' = (-1, [4-3\epsilon, 6-3\epsilon], \delta_4')$.

(d) Else, if the vehicle changes directions, release a request $f_3' = (-1, [4-3T-3\epsilon, 6-3T-3\epsilon], \delta_3')$. If it changes directions again, release a request $f_4'' = (1, [6-4T-5\epsilon, 8-4T-5\epsilon], \delta_4'')$; else, release $f_4''' = (-1, [4-2T-3\epsilon, 6-2T-3\epsilon], \delta_4''')$.

This strategy satisfies the condition relative to the delays and prevents the vehicle from serving two requests. Hence, $\lambda^{\mathcal{I}_{4,T}} = 2 - \sqrt{3}$. □

In order to avoid a lengthy case study, the remaining competitive ratios for $n = 4$ will be listed without detailed proof; but we will give a general idea of how they can be obtained.

Claim 1. *For $n = 4$ and $\frac{1}{6} \leq T < 1$, the competitive ratios are as follows:*

1. *for $\frac{1}{6} \leq T < \frac{1}{5}$, $\gamma^{\mathcal{I}_{4,T}} \approx 0.2578$,*
2. *for $\frac{1}{5} \leq T < \frac{1}{4}$, $\gamma^{\mathcal{I}_{4,T}} = 2 - \sqrt{3} \approx 0.2679$,*
3. *for $\frac{1}{4} \leq T < \frac{1}{3}$, $\gamma^{\mathcal{I}_{4,T}} \approx 0.2803$,*
4. *for $\frac{1}{3} \leq T < \frac{1}{2}$, $\gamma^{\mathcal{I}_{4,T}} = 1 - \frac{\sqrt{2}}{2} \approx 0.2929$,*
5. *for $\frac{1}{2} \leq T < 1$, $\gamma^{\mathcal{I}_{4,T}} \approx 0.3177$.*

General Idea of the proof: If all the requests in figure 3 are maintained, from the delay condition and from the fact that the adversary prevents the vehicle from serving two requests, we obtain the following conditions on the release times:

- $t_4 < \tau_1 = 2$,
- $t_3 \leq t_4 - T < 2 - T$,
- $t_2 \leq t_3 - T < 2 - 2T$,
- $t_4''' < \tau_2 = 2t_2 < 4 - 4T$,
- $t_3' < t_4''' - T < 4 - 5T$,
- $t_4'' < \tau_3 \leq t_3' + 2 - T < 6 - 6T$.

In order for the optimal offline algorithm to serve all the requests, we require that $t_4'' \geq 5$. Hence, this is possible only when $T < \frac{1}{6}$.

If the adversary chooses not to release f_4'', the conditions become:

- $t_4 < 2$,
- $t_3 \leq t_4 - T < 2 - T$,
- $t_2 \leq t_3 - T < 2 - 2T$,
- $t_4''' < \tau_2 = 2t_2 < 4 - 4T$,
- $t_3' < t_4''' - T < 4 - 5T$,

The optimal offline condition now gives $T < \frac{1}{5}$.

If the adversary chooses not to release f_4, the conditions become:

- $t_3 < \tau_1 = 2$,
- $t_2 \leq t_3 - T < 2 - T$,
- $t_4''' < \tau_2 = 2t_2 < 4 - 2T$,
- $t_3' < t_4''' - T < 4 - 3T$,
- $t_4'' < \tau_3 \leq t_3' + 2 - T < 6 - 4T$.

The optimal offline condition now gives $T < \frac{1}{4}$.

If the adversary chooses not to release f_4 and f_4'', the conditions become:

- $t_3 < \tau_1 = 2$,
- $t_2 \leq t_3 - T < 2 - T$,
- $t_4''' < \tau_2 = 2t_2 < 4 - 2T$,
- $t_3' < t_4''' - T < 4 - 3T$,

The optimal offline condition now gives $T < \frac{1}{3}$.

If the adversary chooses not to release f_4 and f_4''', the conditions become:

- $t_3 < \tau_1 = 2$,

- $t_2 \leq t_3 - T < 2 - T$,
- $t'_3 < \tau_2 = 2t_2 < 4 - 2T$,
- $t''_4 < \tau_3 \leq t'_3 + 2 < 6 - 2T$,

The optimal offline condition now gives $T < \frac{1}{2}$.

Solving the corresponding minmax problem in each of these cases yields the above competitive ratios. Note that choosing not to release f'_4 does nothing to loosen the conditions on the times. Also, choosing not to release f'''_4 gives the condition $T < \frac{1}{4}$ and yields the same competitive ratio as not releasing f_4. Finally, not releasing f''_4 and f'''_4 gives $T < \frac{1}{4}$, same as not releasing f'''_4, so it is not relevant. □

In Table 6, we summarise the results obtained for $n = 4$.

Delay	Performance	Competitive Ratio
$T < 1/6$	$1/4$	$1/4$
$1/6 \leq T < 1/5$		0.2578
$1/5 \leq T < 1/4$		$2 - \sqrt{3}$
$1/4 \leq T < 1/3$		0.2803
$1/3 \leq T < 1/2$	$2 - \sqrt{3}$	$1 - \sqrt{2}/2$
$1/2 \leq T < 1$	$1 - \sqrt{2}/2$	0.3177
$1 \leq T < 1.5$	$1/\varphi^2$	$1/\varphi^2$
$1.5 \leq T < 5/3$	$1/2$	$1/2$
$5/3 \leq T$	1	1

Table 6: Performances and Competitive Ratios for $n = 4$

We conjecture that the behaviour of the performance and competitive ratio is the same for $n > 4$ as that exhibited for $n = 4$:

Conjecture 2. *For a fixed $n \geq 4$, the functions $\lambda^{\mathcal{I}_{n,T}}$ and $\gamma^{\mathcal{I}_{n,T}}$ are step functions in T, constant on the interval $[\frac{1}{k}, \frac{1}{k-1}[$, for $2 \leq k \leq 2^{n-1} - 2$.*

8. Generalisation to Geodesic Metric Spaces

In this section, we investigate which of the previous results remain valid when the game is played on a geodesic space instead of a segment.

Recall that the *diameter* of $E = (X, d)$ is defined as $\sup_{(x,y)\in X^2} d(x, y)$.

Proposition 5. *Given a geodesic metric space E of diameter 2 with two points at distance 2 and given a delay T, the performance and competitive ratio on E are at most equal to that on $[-1, 1]$. In other words, any negative result concerning* `OPTiWinD` *on a segment can be extended to geodesic metric spaces for which the diameter is reached.*

Proof. Let $E = (X, d)$ be a geodesic metric space with diameter 2 and $A, B \in X$ such that $d(A, B) = 2$. Any strategy of the adversary applicable to $[-1, 1]$ can be used on a minimum distance path γ from A to B. If all requests are located on γ, since γ is a minimum distance path, the vehicle cannot improve its performance by moving outside of γ. Thus, for all delay T, the performance and competitive ratio on E are at most equal to those on $[-1, 1]$. □

Theorem 8. *Theorem 1, Theorem 3 and Theorem 4 can be extended to all geodesic metric spaces of diameter 2.*

Proof. Let $E = (X, d)$ be a geodesic metric space with diameter 2. The positive part of these results is given in Lemma 1, which states that the algorithm GR_0 guarantees a performance of $\frac{1}{n}$. This is still true when the game is played on E. If the diameter of E is reached, then the result follows from Proposition 5. Otherwise, for $\epsilon > 0$, we can find $A, B \in X$ such that $d(A, B) = 2 - \epsilon$ and a path γ from A to B of length $2 - \epsilon$. Using the same notations as in the proof of Theorem 3, we obtain the following conditions:

- $t_n < 2 - \epsilon$,
- $t_{n-1} < 2 - \epsilon - T$,
- $t_i < 2 - \epsilon - 2^{n-1-i}T$,
- $1 + T \leq t_2 < 2 - \epsilon - 2^{n-3}T$.

Hence, we obtain $T < (1 - \epsilon)T_0$. Thus, choosing $\epsilon < 1 - \frac{T}{T_0}$ is sufficient for the strategy described in the proof of Theorem 3 to work on γ. Theorem 1 also works on γ as it is a special case of Theorem 3.

In the proof of Theorem 4, all that changes when playing on γ is the values of the ETAs, which only affect the proof of point c). However, the relation $\tau_{i+1} = \tau_i + 2 - 2^{n-1-i}T + O(\epsilon)$ is still true. Thus, for ϵ small enough, the optimal offline algorithm will serve all requests, and Theorem 4 is still valid in this case. □

Definition 2. *We say that a geodesic metric space $E = (X, d)$ with origin O and diameter 2 is* centred *if $X \subset \overline{B}(O, 1)$, where $\overline{B}(O, 1)$ is the closed ball of radius 1 centred in O.*

Theorem 9. *Theorem 2 and Theorem 7 can be extended to centred geodesic metric spaces of diameter 2.*

Proof. Let $E = (X, d)$ be a geodesic metric space with diameter 2 and origin O. In the proof of Theorem 2, the positive part of the result is shown by considering algorithm GR_1. When playing on E, we adapt GR_1 by heading towards O instead of 0 in case (a). The proof that GR_1 work on E is identical to that of Theorem 2. The key point is that we still have $i_0 = \lfloor \frac{1}{2-T} \rfloor$.

In order to prove the negative result, we choose $A, B \in X$ such that $d(A, B) = 2 - \epsilon'$ and a minimum distance path γ from A to B. The adversary can apply

on γ the strategy described in the proof of Theorem 2 with one modification: for $i \geq i_0 + 2$, choose $\eta_i = \frac{i-i_0}{n} - \frac{3}{2n} - (i - \frac{3}{2})\epsilon'$. This allows the relation $t_i = \tau_{i-1} - \frac{1}{2n}$ for $i \geq i_0 + 2$ to be valid on a path of length $2 - \epsilon'$. It follows that for $\epsilon' < \frac{5}{n^2}$, we still have $\eta_i < 1$, $\forall i \geq i_0 + 2$. Therefore, this strategy of the adversary limits the competitive ratio of the vehicle to $\alpha_{n-\lfloor \frac{1}{2-T} \rfloor}$.

For Theorem 7, we choose A and B with $d(A, B) > 2T$. In the proof of Theorem 7, -1 and 1 are replaced with A and B, respectively, and the interval $]T-1, 1-T[$ becomes $X - (\overline{B}(A, T) \cup \overline{B}(B, T))$. The rest of the proof remains identical. □

Remark 5. *If E is a circle, E is not centred, and both Theorem 2 and Theorem 7 do not apply whenever $n \geq 3$.*

Proposition 6. *Depending on the geodesic metric space, the smallest delays for which the optimal performance and optimal competitive ratio are greater than $\frac{1}{n}$ may be greater than $T_0 = \frac{1}{2^{n-3}+1}$ and $T_1 = \frac{1}{2^{n-1}-2}$ respectively.*

Proof. Let E be a star with central point O and 3 branches of length 1. Lemma 2 and Lemma 3 do not apply to E as the adversary can now release requests at the end of the third branch. It follows that Theorem 5 and Theorem 6 do no apply when the game is played on E. Let A, B and C denote the endpoints of the branches of the star. For $n = 5$ and $T < \frac{1}{4}$, let $\epsilon < 1 - 4T$. The adversary will release a request $f_1 = (A, [1, 3], 1)$. Recall that when the vehicle may choose between two symmetric requests of equal weights, it is optimal to head towards the closest one. So, while the vehicle keeps going towards A, the adversary will release the following requests: $f_2 = (B, [2-3T-\epsilon, 4-3T-\epsilon], 1)$, $f_3 = (C, [2 - 2T - \epsilon, 4 - 2T - \epsilon], 1)$, $f_4 = (C, [2 - T - \epsilon, 4 - T - \epsilon], 1)$ and $f_5 = (C, [2 - \epsilon, 4 - \epsilon], 1)$. With this strategy, the adversary limits the optimal performance to $\frac{1}{5}$, even if $T_0 < T < \frac{1}{4}$. Note that this strategy was not possible on the segment because it requires that f_2 and f_3 be located in different branches of the star, otherwise the vehicle could reach both. A similar strategy can be used by the adversary to limit the optimal competitive ratio to $\frac{1}{5}$ when T is slightly greater than T_1. □

9. Conclusion

In this paper, we introduced the delay between requests as a new parameter in the standard online orienteering problem. While this new parameter seems very natural, to our knowledge, it had not been studied previously. We analysed the performances and competitive ratios in the case where the length of the time windows is equal to the diameter of the space. We obtained a complete resolution when the number of requests is at most 4. In the case of n requests, we solved the problem when $T \geq 1$ or $T < \frac{1}{2^{n-1}-2}$. Our results for small numbers of requests give us an accurate idea of what to expect in the intermediate case. Other choices regarding the length of the time windows, which may be relevant for different applications, remain to be investigated.

Acknowledgements

We acknowledge the support of GEO-SAFE, H2020-MSCA-RISE-2015 project # 691161.

References

[1] Ausiello, G., Demange, M., Laura, L., and Paschos, V. (2004). Algorithms for the on-line quota traveling salesman problem. *Information Processing Letters*, 92(2):89 – 94.

[2] Ausiello, G., Feuerstein, E., Leonardi, S., Stougie, L., and Talamo, M. (2001). Algorithms for the on-line travelling salesman1. *Algorithmica*, 29(4):560–581.

[3] Awerbuch, B., Azar, Y., Blum, A., and Vempala, S. (1997). New approximation guarantees for minimum-weight k-trees and prize-collecting salesmen. *SIAM Journal on Computing*, 28.

[4] Deza, M. M. and Deza, E. (2009). *Encyclopedia of distances*. Springer, Berlin.

[5] Feillet, D., Dejax, P., and Gendreau, M. (2005). Traveling salesman problems with profits. *Transportation Science*, 39(2):188–205.

[6] Golden, B. L., Levy, L., and Vohra, R. (1987). The orienteering problem. *Naval Research Logistics (NRL)*, 34(3):307–318.

[7] Gutiérrez, S., Krumke, S. O., Megow, N., and Vredeveld, T. (2006). How to whack moles. *Theoretical Computer Science*, 361(2):329–341.

[8] Irani, S., Lu, X., and Regan, A. (2004). On-line algorithms for the dynamic traveling repair problem. *Journal of Scheduling*, 7(3):243–258.

[9] Laporte, G. and Martello, S. (1990). The selective travelling salesman problem. *Discrete Applied Mathematics*, 26(2-3):193–207.

[10] Vansteenwegen, P., Souffriau, W., and Oudheusden, D. V. (2011). The orienteering problem: A survey. *European Journal of Operational Research*, 209(1):1 – 10.

Appendix

In this section, we study the sequence (α_n) defined in Section 4. We give an explicit formula for δ_i and calculate the limit of (α_n).

We have the following definition of α_n for $n \geq 1$:

$$\alpha_n = \inf_{\delta \in (\mathbb{R}_+^*)^n} \max\{\frac{\delta_1}{S_2}, \cdots, \frac{\delta_i}{S_{i+1}}, \cdots, \frac{\delta_{n-1}}{S_n}, \frac{\delta_n}{S_n}\},$$

where $S_i = \sum_{k=1}^{i} \delta_k$. Since multiplying all the δ_i by a positive constant makes no difference to the ratios, we may choose $\delta_1 = 1$. To lighten the notation, we will omit the index n and will denote α_n by α.

Proposition 7. *The value of α is realised for a unique vector $\delta \in (\mathbb{R}_+^*)^n$ with $\delta_1 = 1$ such that for $1 \leq i \leq n$:*

$$\delta_i = \Big(\frac{1}{2} + \frac{1-2\alpha}{2\sqrt{1-4\alpha}}\Big)\Big(\frac{1+\sqrt{1-4\alpha}}{2\alpha}\Big)^{i-1} + \Big(\frac{1}{2} - \frac{1-2\alpha}{2\sqrt{1-4\alpha}}\Big)\Big(\frac{1-\sqrt{1-4\alpha}}{2\alpha}\Big)^{i-1}.$$

Proof. Note that:

$$\begin{aligned}
&\inf_{\delta_{n-1},\delta_n} \max\{\frac{\delta_{n-2}}{S_{n-1}}, \frac{\delta_{n-1}}{S_n}, \frac{\delta_n}{S_n}\} \\
&\qquad = \inf_{\delta_{n-1}} \inf_{\delta_n} \max\{\frac{\delta_{n-2}}{S_{n-1}}, \max\{\frac{\delta_{n-1}}{S_n}, \frac{\delta_n}{S_n}\}\} \\
&\qquad = \inf_{\delta_{n-1}} \max\{\frac{\delta_{n-2}}{S_{n-1}}, \inf_{\delta_n} \max\{\frac{\delta_{n-1}}{S_n}, \frac{\delta_n}{S_n}\}\}.
\end{aligned}$$

By repeating this inversion, we obtain that:

$$\alpha = \inf_{\delta_2} \max\{\frac{1}{S_2}, \inf_{\delta_3} \max\{\frac{\delta_2}{S_3}, \inf_{\delta_4} \max\{ \cdots, \inf_{\delta_n} \max\{\frac{\delta_{n-1}}{S_n}, \frac{\delta_n}{S_n}\} \cdots\}\}\}$$

In the above equation, the operator $\inf_{\delta_i}$ is applied to the max of two terms, the first being decreasing in δ_i and the second increasing in δ_i. It follows that the optimal value is reached when:

$$\forall 1 \leq i \leq n-1, \ \frac{\delta_i}{S_{i+1}} = \frac{\delta_n}{S_n} = \alpha$$

Thus, we obtain the following equations (E_i), for all $2 \leq i \leq n$:

$$(E_i) \ \ \alpha = \frac{\delta_i}{S_{i+1}} = \frac{\delta_{i-1}}{S_i} \ \text{ and } \ (E_n) \ \ \delta_{n-1} = \delta_n = \alpha S_n.$$

Thus, $\delta_i = \alpha S_{i+1}$ and $\delta_{i-1} = \alpha S_i$. Hence, $\delta_i - \delta_{i-1} = \alpha\delta_{i+1}$. The sequence (δ_i) satisfies a second order linear recurrence relation. Its characteristic equation is $X^2 - \frac{1}{\alpha}X + \frac{1}{\alpha} = 0$ and the discriminant is $\Delta = \frac{1}{\alpha^2}(1 - 4\alpha)$.

If $\Delta = 0$, we have $\alpha = \frac{1}{4}$. Hence, $\delta_i = (i+1)2^{i-2}$, for $1 \leq i \leq n$. This contradicts (E_n). So $\Delta \neq 0$.

Thus the characteristic equation has two roots $r_\pm = \frac{1}{2\alpha}(1 \pm \sqrt{1-4\alpha})$, where $\sqrt{1-4\alpha}$ may be an imaginary number. So there exists a λ and a μ such that $\forall 1 \leq i \leq n, \delta_i = \lambda r_+^{i-1} + \mu r_-^{i-1}$.

Since $\delta_1 = 1$, we have $\lambda + \mu = 1$. Similarly, using δ_2, we obtain that $\lambda - \mu = \frac{2}{\sqrt{\Delta}}(\delta_2 - \frac{1}{2\alpha})$. The result follows. □

Lemma 4. *The roots of the characteristic equation $X^2 - \frac{1}{\alpha}X + \frac{1}{\alpha} = 0$ are complex conjugates.*

Proof. Using the same notations as above, let us assume that $\Delta > 0$ (i.e. $\alpha < \frac{1}{4}$). Then, r_+, r_-, λ and μ are real numbers. It follows from (E_1) that $\delta_2 = \frac{1}{\alpha} - 1$. Hence, $\lambda - \mu > 0$. Since $\lambda + \mu = 1$, we have $\lambda > 0$. Since $\sqrt{1-4\alpha} < 1$, we have $r_+ \geq r_- > 0$.

Since $\delta_{n-1} = \delta_n$, we have:

$$\lambda r_+^{n-2}(r_+ - 1) = -\mu r_-^{n-2}(r_- - 1) \tag{1}$$

Since $\alpha < \frac{1}{4}$, we have $r_+ \geq 2$. It follows that in Equation 1, the left hand side is positive and $\mu \neq 0$. Let us now consider two cases:

1. If $\mu < 0$, it follows from Equation 1 that $r_- - 1 > 0$, since all the other terms are positive. Yet, $\lambda > -\mu$ and $r_+ \geq r_-$. Hence, Equation 1 is impossible.

2. If $\mu > 0$, the right hand side of Equation 1 being positive requires $r_- < 1$. Since $r_+ \geq 2$ and $r_- > 0$, we have $r_+ - 1 > 1 - r_-$. Yet, since $\lambda \geq \mu$, Equation 1 is impossible.

Therefore, since $\Delta \neq 0$, we have $\Delta < 0$ and $\alpha > \frac{1}{4}$. Hence, r_+ and r_- are complex conjugates, as are λ and μ.

□

Lemma 5. *The sequence (α_n) is decreasing.*

Proof. Choosing $(\delta_1, \ldots, \delta_n)$ as the vector which realises α_{n-1} and $\delta_n = 0$ yields: $\max\{\frac{\delta_1}{S_2}, \cdots, \frac{\delta_i}{S_{i+1}}, \cdots, \frac{\delta_{n-1}}{S_n}, \frac{\delta_n}{S_n}\} = \alpha_{n-1}$. Hence, $\alpha_n \leq \alpha_{n-1}$. Since this is not the vector described in Proposition 7, we have $\alpha_n < \alpha_{n-1}$. □

Proposition 8. *The sequence (α_n) decreases towards $\frac{1}{4}$ when $n \to +\infty$.*

Proof. Using the same notations as above, it follows from Proposition 7 and Lemma 4 that $\delta_i = 2\,\mathrm{Re}(\lambda r_+^i) > 0$. Hence, $-\frac{\pi}{2} < \arg(\lambda r_+^i) < \frac{\pi}{2}$. Thus,

$$i.\arg r_+ + \arg\lambda \mod (2\pi) < \frac{\pi}{2}.$$

Since $\lambda + \mu = 1$, we have $\mathrm{Re}(\lambda) = \frac{1}{2}$, and $-\frac{\pi}{2} < \arg\lambda < \frac{\pi}{2}$. Hence, $0 < i.\arg r_+ \mod (2\pi) < \pi$. Thus, there exists an integer k_i, $0 \leq k_i < i$, such that $\frac{2k_i\pi}{i} < \arg r_+ < \frac{(2k_i+1)\pi}{i}$. Since this is true for all $1 \leq i \leq n$, we have $k_i = 0$ for all i, and $0 < \arg r_+ < \frac{\pi}{n}$. Therefore $\lim_{n\to+\infty} \arg r_+ = 0$. Since $\mathrm{Im}(r_+) = \frac{\sqrt{4\alpha-1}}{2\alpha}$, we have $\lim_{n\to+\infty} \alpha = \frac{1}{4}$.

□

Proposition 9. *The first values of* (α_n) *are* $\alpha_1 = 1$, $\alpha_2 = \frac{1}{2}$, $\alpha_3 = \frac{1}{\varphi^2}$, *where* φ *is the golden ratio, and* $\alpha_4 = \frac{1}{3}$.

Proof. While α_1 is trivially equal to 1, the following terms of the sequence are calculated using the equations (E_i).

- For $n = 2$, (E_2) yields $\delta_1 = \delta_2 = 1$ and $\alpha = \frac{1}{2}$.
- For $n = 3$, equations (E_2) and (E_3) yield $\frac{1}{S_2} = \frac{\delta_2}{S_3} = \frac{\delta_3}{S_3} = \alpha$. Thus, $\delta_2 = \delta_3 = \varphi$ and $\alpha = \frac{1}{\varphi^2}$.
- For $n = 4$, equations (E_2), (E_3) and (E_4) yield $\frac{1}{S_2} = \frac{\delta_2}{S_3} = \frac{\delta_3}{S_4} = \frac{\delta_4}{S_4} = \alpha$. Thus $\delta_2 = 2$ and $\delta_3 = \delta_4 = 3$ and $\alpha = \frac{1}{3}$.

□