Minimizing the maximum flow time in the online-TSP on the real line

Citation for published version (APA):

Document status and date:
Published: 01/01/2003

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.

Download date: 14. Jun. 2020
Minimizing the Maximum Flow Time in the Online-TSP on the Real Line

Sven O. Krumke\textsuperscript{a,1} Luigi Laura\textsuperscript{b} Maarten Lipmann\textsuperscript{c,3} Alberto Marchetti-Spaccamela\textsuperscript{b} Willem E. de Paepe\textsuperscript{d,3} Diana Poensgen\textsuperscript{a,3} Leen Stougie\textsuperscript{e,2}

\textsuperscript{a}Konrad-Zuse-Zentrum für Informationstechnik Berlin, Department Optimization, Takustr. 7, 14195 Berlin-Dahlem, Germany. Email: \{krumke,poensgen\}@zib.de
\textsuperscript{b}Dipartimento di Informatica e Sistemistica, Università di Roma “La Sapienza”, Via Salaria 113, 00198, Rome, Italy. Email: \{alberto,laura\}@dis.uniroma1.it
\textsuperscript{c}Technical University of Eindhoven, Dept. of Mathematics and Computer Science, P. O. Box 513, 5600 MB Eindhoven, The Netherlands. Email: m.lipmann@tue.nl
\textsuperscript{d}Department of Technology Management, Technical University of Eindhoven, P. O. Box 513, 5600MB Eindhoven, The Netherlands. Email: w.e.d.paepe@tm.tue.nl
\textsuperscript{e}Department of Mathematics, Technical University of Eindhoven, P. O. Box 513, 5600MB Eindhoven, The Netherlands and Centre for Mathematics and Computer Science (CWI), P. O. Box 94079, NL-1090 GB Amsterdam, The Netherlands. Email: leen@win.tue.nl

Abstract

In the online traveling salesman problem (OLTSP) requests for visits to cities arrive online while the salesman is traveling. The $F_{\text{max}}$-OLTSP has as objective to minimize the maximum flow time, which is particularly interesting for applications. Unfortunately, there can be no competitive algorithm, neither deterministic nor randomized. Hence, competitive analysis fails to distinguish online algorithms. Not even resource augmentation which is helpful in scheduling works as a remedy. This motivates the search for alternative analysis methods.

We introduce a natural restriction on the adversary for the $F_{\text{max}}$-OLTSP on the real line. A non-abusive adversary may only move in a direction where there are yet unserved requests. Our main result is an 8-competitive algorithm against the non-abusive adversary.

\textit{Key words:} Online Algorithms, Competitive Analysis, Maximum Flow Time, Comparative Analysis
1 Introduction

In the online traveling salesman problem (OLTsp) requests for visits to cities arrive online while the salesman is traveling. An online algorithm learns from the existence of a request only at its release time. The OLTsp has been studied for the objectives of minimizing the makespan [2,1,5], the weighted sum of completion times [5,8], and the maximum/average flow time [6]. In view of applications, the maximum flow time is of particular interest. For instance, it can be identified with the maximal dissatisfaction of customers. Alas, there can be no competitive algorithm, neither deterministic nor randomized [6]. Moreover, in contrast to scheduling [11], resource augmentation, e.g. providing the online algorithm with a faster server, does not help, the crucial difference being that servers move in space.

The only hope to overcome the weaknesses of standard competitive analysis in the context of the $F_{\text{max}}$-OLTsp is to restrict the powers of the adversary. In this paper we consider the $F_{\text{max}}$-OLTsp on the real line and introduce a natural restriction on the adversary: a non-abusive adversary may move its server only in a direction, if yet unserved requests are pending on that side. We construct an algorithm, called DETOUR which achieves a competitive ratio of eight against the non-abusive adversary.

Our approach fits the concept of comparative analysis for restricting the adversary introduced by Koutsoupias and Papadimitriou [7]. The fair adversary of Blom et al. [3] implements this concept in the context of the OLTsp as follows: a fair adversary may only move within the convex hull of all requests released so far. While one can obtain improved competitiveness results for the minimization of the makespan against a fair adversary [3], still a constant competitive ratio for the maximum flow time is out of reach (see Theorem 1). The non-abusive adversary presented in this paper can be viewed as a refinement of the fair adversary.

An extension of the fairness concept to the uniform metric space, i.e., the complete graph with unit edge lengths, was recently considered in [9]. It is shown that the first-come-first-serve strategy is 2-competitive against the fair adversary on this metric space if requests are only given at the vertices. However, no algorithm can be competitive for the dial-a-ride extension of the problem, where objects have to be transported between sources and destinations.

---

1 Research supported by the German Science Foundation (DFG, grant Gr 883/10)
2 Supported by the TMR Network DONET of the European Community ERB TMRX-CT98-0202
3 Supprted by Research Training Network AMORE of the European Commission, contract no HPRN-CT-1999-00104
In Section 2 we formally define the $F_{\text{max}}$-OLTsp and the non-abusive adversary. We also show lower bound results for the competitive ratio against a fair and non-abusive adversary, respectively. Section 3 presents our algorithm DETOUR, the proof of its performance is given in Section 4.

2 Preliminaries

An instance of the Online Traveling Salesman Problem ($F_{\text{max}}$-OLTsp) consists of a metric space $M = (X, d)$ with a distinguished origin $o \in X$ and a sequence $\sigma = r_1, \ldots, r_m$ of requests. A server is located at the origin $o$ at time 0 and can move at most at unit speed. In this paper we are concerned with the special case that $M$ is $\mathbb{R}$, the real line endowed with the Euclidean metric $d(x, y) = |x - y|$; the origin $o$ equals the point 0. Each request is a pair $r_i = (t_i, x_i)$, where $t_i \in \mathbb{R}_+$ is the time at which request $r_i$ is released, and $x_i \in X$ is the point in the metric space to be visited. We assume that the sequence $\sigma$ of requests is given in order of non-decreasing release times.

For $t \geq 0$, we denote by $\sigma_{\leq t}$ ($\sigma_{< t}$) the subsequence of requests in $\sigma$ released up to time $t$ (strictly before time $t$).

An online algorithm $\text{ALG}$ gets to know request $r_j$ only at its release time $t_j$. In particular, $\text{ALG}$ has neither information about the release time of the last request nor about the total number of requests. Hence, at time $t$, $\text{ALG}$ must make its decisions only knowing the requests in $\sigma_{\leq t}$. An offline algorithm has complete knowledge about the sequence $\sigma$ already at time 0.

Given a sequence $\sigma$ of requests, an algorithm $\text{ALG}$ for the $F_{\text{max}}$-OLTsp must find a route for the server which starts in the origin and visits each point in $\sigma$, but not earlier than its release time. By $C_{j}^{\text{ALG}}$ and $F_{j}^{\text{ALG}} = C_{j}^{\text{ALG}} - t_j$ we denote the completion time and flow time of request $r_j$, respectively, in the solution produced by $\text{ALG}$. The goal in the $F_{\text{max}}$-OLTsp is to minimize the maximum flow time $\text{ALG}(\sigma) := \max_j F_{j}^{\text{ALG}}$.

Let $\text{OPT}$ denote an optimal offline algorithm. A deterministic online algorithm $\text{ALG}$ for the $F_{\text{max}}$-OLTsp is $c$-competitive, if there exists a constant $c$ such that for any request sequence $\sigma$, $\text{ALG}(\sigma) \leq c \cdot \text{OPT}(\sigma)$. If $\text{ALG}$ is randomized, then $\text{ALG}(\sigma)$ is replaced by the expected solution value (w.r.t. to the oblivious adversary model, see [4]). The competitive ratio of $\text{ALG}$ is the infimum over all $c$ such that $\text{ALG}$ is $c$-competitive.

The following lower bound result shows that the fairness restriction on the adversary introduced in [3] is still not strong enough to allow for competitive algorithms in the $F_{\text{max}}$-OLTsp.
Theorem 1 No randomized algorithm for the $F_{\text{max}}$-OLTsp on $\mathbb{R}$ can achieve a constant competitive ratio against an oblivious adversary. This result still holds, even if the adversary is fair, i.e., if at any moment in time $t$ the server operated by the adversary is within the convex hull of the origin and the requested points from $\sigma_{\leq t}$.

PROOF. Let $\varepsilon > 0$ and $k \in \mathbb{N}$. We give two request sequences $\sigma_1 = (\varepsilon, \varepsilon), (2\varepsilon, 2\varepsilon), \ldots, (k\varepsilon, k\varepsilon), (T, 0)$ and $\sigma_2 = (\varepsilon, \varepsilon), (2\varepsilon, 2\varepsilon), \ldots, (k\varepsilon, k\varepsilon), (T, k\varepsilon)$, each with probability $1/2$, where $T = 4k\varepsilon$.

The expected cost of an optimal fair offline solution is at most $\varepsilon$. Any deterministic online algorithm has cost at least $k\varepsilon/2$. The Theorem follows by applying Yao’s principle [4,10]. $\square$

The fair adversary is still too powerful in the sense that it can move to points where it knows that a request will appear without revealing any information to the online server before reaching the point. A non-abusive adversary does not possess this power.

Definition 2 (Non-Abusive Adversary) An adversary $ADV$ for the OLTsp on $\mathbb{R}$ is non-abusive, if the following holds: At any moment in time $t$, where the adversary moves its server from its current position $p^{ADV}(t)$ to the right (left), there is a request from $\sigma_{\leq t}$ to the right (left) of $p^{ADV}(t)$ which $ADV$ has not served yet.

In the sequel we slightly abuse notation and denote by $OPT(\sigma)$ the maximal flow time in an optimal non-abusive offline solution for the sequence $\sigma$. The following result shows that the $F_{\text{max}}$-OLTsp is still non-trivial against a non-abusive adversary.

Theorem 3 No deterministic algorithm for the $F_{\text{max}}$-OLTsp on $\mathbb{R}$ can achieve a competitive ratio less than 2 against a non-abusive adversary.

PROOF. Let $ALG$ be any deterministic online algorithm. The adversary first presents the following $2m$ requests: $(0, \pm 1), (3, \pm 2), \ldots, (\sum_{k=1}^{m-1}(1 + 2k), \pm m)$. W.l.o.g., let $ALG$ serve the request in $-m$ later than the one in $+m$, and let $T$ be the time it reaches $-m$. Clearly, $T \geq \sum_{k=1}^{m}(1 + 2k)$. At time $T$, the adversary presents one more request in $+(3m + 1)$ which results in a flow time of at least $4m + 1$ for $ALG$. On the other hand, a non-abusive offline algorithm can serve all of the first $2m$ requests with maximum flow time $2m + 1$ by time $\sum_{k=1}^{m}(1 + 2k)$, ending with the request at $+m$. From there it can easily reach the last request with flow time $2m + 1$. The theorem follows by letting $m \to \infty$. $\square$
We now present the algorithm DETOUR (short DTO) which achieves a constant competitive ratio for the $F_{\text{max}}$-OLTSP against a non-abusive adversary on $\mathbb{R}$. Before giving a concise statement of the algorithm, we describe its main idea.

DTO’s decisions at time $t$ are based on an approximation of $\text{OPT}(\sigma_{\leq t})$, called the guess $G(t)$. Roughly, DTO’s strategy is to serve all requests in a first-come-first-serve (FCFS) manner. However, blindly doing so makes it easy for the adversary to fool the algorithm. DTO enforces the offline cost in a malicious sequence to increase by making a detour on its way to the next “target”: it first moves its server in the “wrong direction” as long as it can serve the target with flow time thrice the guess. If the guess changes, the detour, and possibly the target, are adjusted accordingly (this requires some technicalities in the description of the algorithm).

We denote by $p_{\text{DTO}}(t)$ the position of the server operated by DTO at time $t$. The terms ahead of and in the back of the server refer to positions on the axis w.r.t. the direction the server currently moves: if it is moving from left to right on the $\mathbb{R}$-axis, “ahead” means to the right of the server’s current position, while a request “in the back” of the server is to its left. The other case is defined analogously.

Given a point $p \in \mathbb{R}$, we call the pair $(t_i, x_i)$ more critical than the request $r_j = (t_j, x_j)$ w.r.t. $p$ if both $x_i$ and $x_j$ are on the same side of $p$, and $d(p, x_i) − t_i > d(p, x_j) − t_j$. If request $r_i$ is more critical than $r_j$ w.r.t. DTO’s position $p_{\text{DTO}}(t)$ at time $t$, then $F_{i, \text{DTO}} > F_{j, \text{DTO}}$. Moreover, $r_i$ remains more critical than $r_j$ after time $t$ as long as both requests are unserved. Conversely, we have the following observation.

**Observation 1** If, at time $t$, request $r_i$ is more critical than $r_j$ w.r.t. $p_{\text{DTO}}(t)$, and DTO moves straight ahead to the more distant request after having served the one closer to $p_{\text{DTO}}(t)$, then $F_{j, \text{DTO}} \leq F_{i, \text{DTO}}$.

The critical region $V(r_j, p, G)$ w.r.t. a request $r_j$, a point $p \in \mathbb{R}$ and a bound $G$ for the allowed maximal flow time contains all those pairs $(t, x) \in \mathbb{R}_+ \times \mathbb{R}$ such that (i) $(t, x)$ is more critical than $r_j$ w.r.t. $p$, and (ii) $t + d(x, x_j) − t_j \leq G$.

Note that $V(r_j, p, G)$ is the intersection of a cone $C$ with the halfplane of points which are located on the same side of $p$ as $r_j$. The cone $C$ covers an angle of $\pi/2$ and stands on its tip, which is at distance $G/2$ in both coordinates from request $r_j$. Figure 1 illustrates the critical region.

In the setting of DTO, $p$ will be the position of the online server at a certain
time $t'$. Condition (ii) implies that a request in $(t, x)$ could be served before $r_j$ in an offline tour serving both $r_j$ and $(t, x)$ with flow time at most $G$.

**DTO** can assume three modes:

**idle** In this mode, **DTO**’s server has served all unserved requests, and is waiting for new requests at the point at which it served the last request.

**focus** Here, the server is moving in one direction serving requests until a request in its back becomes the oldest unserved one or all requests have been served.

**detour** In this case, the server is moving away from its current target (possibly serving requests on the way), thus making a “detour”.

At any time, at most one unserved request is marked as a target by **DTO**. Moreover, it keeps at most one critical region, denoted by $V$. Before we formalize the behavior of **DTO** we specify important building blocks for the algorithm.

- **Guess Update**: Replace the current guess value $G$ by $G'$, defined as follows: If $G = 0$, then $G' := \text{OPT}(\sigma \leq t)$. If $G > 0$, then $G' := 2^a G$, where $a$ is the smallest integer $k$ such that $\text{OPT}(\sigma \leq t) \leq 2^k G$.

- **Target Selection**: Given a candidate set $C$ and the current time $t$, let $s_0 = (t_0, x_0)$ be the most critical request from $C$ w.r.t. $p^{\text{DTO}}(t)$ with the property that $s_0$ is feasible in the following sense:

  Let $X_0$ be the point ahead of the server such that $t + d(p^{\text{DTO}}(t), X_0) + d(X_0, x_0) = t_0 + 3G$, provided such a point exists, otherwise let $X_0 := p^{\text{DTO}}(t)$. Define the **turning point** $T_0 = (T_0, X_0)$, where $T_0 := t + d(p^{\text{DTO}}(t), X_0)$. There is no unserved request ahead of the server further away from $p^{\text{DTO}}(t)$ than $T_0$ and older than $s_0$.

  If necessary, unmark the current target and turning point. Mark $s_0$ as a target and set $T_0$ to be the current turning point.

- **Mode Selection**: If $X_0 \neq p^{\text{DTO}}(t)$, then set $V := V(s_0, p^{\text{DTO}}(t), G)$ and enter the detour mode. Otherwise, set $V := \emptyset$ and unmark $s_0$ as a target. Change the direction and enter the focus mode.
Figure 2 illustrates the target selection. We now specify for each of the three states how DTO reacts to possible events. All events not mentioned are ignored. In the beginning, the guess value is set to $G := 0$, $V := \emptyset$ and the algorithm is in the idle mode.

**Idle Mode:** In the idle mode DTO waits for the next request to occur.

- A new request is released at time $t$.
  
  The pair $(t, p^{DTO}(t))$ is called a selection point. DTO performs a guess update. The direction of the server is defined such that the oldest unserved request is in his back. If there is one oldest request on both sides, the server chooses to have the most critical one in his back. DTO defines $C$ to be the set of unserved requests in the back of the server and performs a target selection, followed by a mode selection.

**Detour Mode:** In this mode, DTO has a current target $s_m = (t_m, x_m)$, a critical region $V \neq \emptyset$ and a turning point TP. Let $T$ be the time DTO entered the detour mode and $s_0$ the then chosen target. The server moves towards TP until one of following two events happens, where the first one has a higher priority than the second one, if both occur simultaneously.

- A new request is released at time $t$ or an already existing request has just been served.
  
  DTO performs a guess update. Then it enlarges the critical region to $V := V(s_0, p^{DTO}(T), G)$ where $G$ is the updated guess value. It replaces the old turning point by a new point TP which satisfies $t + d(p^{DTO}(t), TP) + d(TP, x_m) = t_m + 3G$ for the updated guess value $G$.
  
  DTO defines $C$ to be the set of unserved requests which are in $V$ and more critical than the current target $s_m$. If $C \neq \emptyset$, it executes the target selection and the mode selection. If $C = \emptyset$, DTO remains in the detour mode.

- The turning point TP is reached at time $t$.
  
  DTO unmarks the current target, sets $V := \emptyset$ and clears the turning point. The server reverses direction and enters the focus mode.

**Focus Mode:** When entering the focus mode, DTO’s server has a direction. It moves in this direction, reacting to the following events:

- A new request is released at time $t$.
  
  A guess update is performed, and the server remains in the focus mode.

- The last unserved request has been served.
  
  The server stops, forgets its direction and enters the idle mode.

- A request in the back of the server becomes the oldest unserved request.
  
  If this happens at time $t$, the pair $(t, p^{DTO}(t))$ is also called a selection point. DTO defines $C$ to be the set of unserved requests in the back of the server and performs a target selection, followed by a mode selection.
Fig. 2. The target selection in the detour mode

4 Analysis of DETOUR

For the analysis of DTO we compare intermediate solutions DTO(σ ≤ t) not only to the optimal non-abusive solution on σ ≤ t, but to a class of solutions ADV(t) defined as the set of all non-abusive offline solutions for the sequence σ ≤ t with the property that the flow time of any request in σ ≤ t is bounded from above by G(t). For ri ∈ σ ≤ t, the smallest achievable flow time αi(t) is defined to be the minimum flow time of ri taken over all solutions in ADV(t).

Notice that for any time t we have OPT(σ ≤ t) ≤ G(t) ≤ 2OPT(σ ≤ t), and, by definition, αi(t) ≤ G(t). In general, αi(t) ≥ αi(t′) for t′ > t, as an increase in the allowed flow time can help an adversary to serve a request ri earlier. On the other hand, we have the following property.

Observation 2 If t ≤ t′ and G(t) = G(t′), then αi(t) ≤ αi(t′) for any request ri = (ti, xi) with ti ≤ t.

To derive bounds on the flow times for DTO we would like to conclude as follows: if request ri is served by DTO “in time” and rj is served directly after ri (i.e., without any detour in between), then rj is also served “in time”.

Definition 4 (Served in time) Given ri = (ti, xi), we define τi := max{ti, T}, where T is the last time DTO reverses direction before serving ri. We say that ri is served in time by DTO, if C_i^{DTO} ≤ ti + 3G(τi) + αi(τi).

Notice that any request ri served in time has F_i^{DTO} ≤ 4G(τi) ≤ 4G(C_i^{DTO}) since αi(τi) ≤ G(τi) by definition.

Lemma 5 (In-Time-Lemma) Let ri = (ti, xi) and rj = (tj, xj) be two requests such that: (i) ri is served in time by DTO, (ii) C_j^{DTO} ≤ C_i^{DTO} + d(xi, xj),
(iii) $\tau_i \leq \tau_j$, (iv) $r_j$ is served after $r_i$ by all $ADV \in ADV(\tau_j)$. Then, DTO serves $r_j$ in time.

**PROOF.** From (iv) we have

$$t_j + \alpha_j(\tau_j) \geq t_i + \alpha_i(\tau_j) + d(x_i, x_j). \quad (1)$$

In particular, this implies

$$t_i + d(x_i, x_j) \leq t_j + G(\tau_j). \quad (2)$$

By (i), (ii), and the definition of served in time, we get:

$$C_j^{DTO} \leq t_i + \alpha_i(\tau_i) + 3G(\tau_i) + d(x_i, x_j). \quad (3)$$

If $G(\tau_i) = G(\tau_j)$, we have that $\alpha_i(\tau_i) \leq \alpha_i(\tau_j)$ by Observation 2. In this case, inequality (3) combined with (1) yields that

$$C_j^{DTO} \leq t_i + \alpha_i(\tau_j) + 3G(\tau_j) + d(x_i, x_j) \leq t_j + \alpha_j(\tau_j) + 3G(\tau_j).$$

If $G(\tau_i) < G(\tau_j)$, then $2G(\tau_i) \leq G(\tau_j)$, and inequalities (3) and (2) imply

$$C_j^{DTO} \leq t_i + 4G(\tau_i) + d(x_i, x_j) \leq t_j + 2G(\tau_j) + d(x_i, x_j) \leq t_j + G(\tau_j) + 2G(\tau_i).$$

□

An easy but helpful condition which ensures that assumption (iv) of the In-Time-Lemma holds, is that $t_j + d(x_j, x_i) > t_i + G(t)$. This yields the following observation which will be used frequently in order to apply the In-Time-Lemma:

**Observation 3** (i) If $d(x_i, x_j) > G(t)$ and $t_i \leq t_j \leq t$, then $r_i$, the older request, must be served before $r_j$ in any offline solution in $ADV(t)$.

(ii) If a request $r_j$ is outside the critical region $V(r_i, p, G(t))$ valid at time $t$, request $r_j$ is served after $r_i$ in any offline solution in $ADV(t)$.

We define a busy period to be the time period in between two consecutive idle modes of DTO.

**Lemma 6** Suppose that at the beginning of a busy period at time $t$, it holds for each request $r_j$ served in one of the preceding busy periods that $F_j^{DTO} \leq 4G(C_j^{DTO})$. Then, $d(p^{DTO}(t), p^{ADV}(t)) \leq 5/2G(t)$ for any $ADV \in ADV(t)$. 

9
PROOF. The claim of the lemma is trivially true for the first busy period, since a non-abusive adversary must keep its server in the origin until the first request is released. Hence, it suffices to consider the later busy periods.

Let \( r_l \) be last request served by DTO in the preceding busy period, so DTO enters the idle mode at time \( C_l^{\text{DTO}} \). Consider that \( \text{ADV} \in \mathcal{ADV}(t) \) in which the adversary’s server is furthest away from \( p^{\text{DTO}}(t) = x_l \).

**Case 1:** At time \( t \), ADV has served all requests in \( \sigma_{<t} \).

In this case, since ADV is non-abusive, its server satisfies \( p^{\text{ADV}}(t) = x_k \) for the request \( r_k \) it served last. DTO must have served \( r_k \) in the preceding busy period, hence no later than \( r_l \). This gives \( C_k^{\text{DTO}} \leq C_l^{\text{DTO}} - d(x_k, x_l) \) and hence

\[
t_k \leq C_l^{\text{DTO}} - d(x_k, x_l) \leq t_l + 4G(C_l^{\text{DTO}}) - d(x_k, x_l) \leq t_l + 4G(t) - d(x_k, x_l)
\]

since \( F_l^{\text{DTO}} \leq 4G(C_l^{\text{DTO}}) \) by the condition of the lemma. On the other hand, ADV serves \( r_l \) no later than \( r_k \), which implies that \( t_l + d(x_k, x_l) \leq t_k + G(t) \). Together with (4) this yields \( d(x_k, x_l) \leq t_k - t_l + G(t) \leq 5G(t) - d(x_k, x_l) \), implying the Lemma, since \( d(p^{\text{DTO}}(t), p^{\text{ADV}}(t)) = d(x_k, x_l) \).

**Case 2:** At time \( t \), there is a request from \( \sigma_{<t} \) yet unserved by ADV.

If \( r_l \) has not been served by ADV at time \( t \), \( d(p^{\text{ADV}}(t), x_l) = d(p^{\text{ADV}}(t), p^{\text{DTO}}(t)) \leq G(t) \), because otherwise the adversary’s flow time for \( r_l \) would be greater than \( G(t) \).

Otherwise, \( r_l \) has been served, but another request in \( \sigma_{<t} \) is yet unserved by ADV. Let \( r_k \) be the request in \( \sigma_{<t} \) which is furthest away from \( r_l \) and yet unserved by ADV. The same argument as in Case 1 shows that \( d(x_k, x_l) \leq \frac{5}{2}G(C_l^{\text{DTO}}) \). So, if the adversary’s server is between \( r_k \) and \( r_l \), the Lemma is true. Assume that the adversary’s server is further away from \( r_l \) than \( r_k \). Since the adversary is non-abusive, there must be a request \( r_j \) even further away from \( r_l \) than \( p^{\text{ADV}}(t) \), which ADV served last before (or at) time \( t \). In particular, ADV served \( r_l \) before \( r_j \). Thus, the same arguments as in Case 1 apply to \( r_j \) instead of \( r_k \), showing that \( d(x_j, x_l) \leq \frac{5}{2}G(C_l^{\text{DTO}}) \). \( \square \)

We further subdivide each busy period into *phases*, where a phase is defined to be the time between two subsequent selection points of DTO. Remember that DTO reaches a selection point whenever it leaves the idle mode, and each time at which a request in the server’s back becomes the oldest unserved one. The following statement is the key theorem of our analysis.

**Theorem 7** *The following is true for any phase \( \rho \geq 1$:*
(a) At any time $t$ in phase $\rho$ at which DTO is in the detour mode, $d(X_i, x_i) \geq G(t)$ for the turning point $TP_i = (X_i, T_i)$ valid at that time and its corresponding target $r_i = (t_i, x_i)$. Moreover, if at some time $t$ during phase $\rho$, a request $r_i$ failed to become a new target only because it was infeasible, the above inequality holds as well for $r_i$ and its hypothetical turning point $TP_i$.

(b) Any request $r_j$ served in phase $\rho$ has $F_{DTO}^j \leq 4G(C_{DTO}^j)$.

(c) The last request served in phase $\rho$ is served in time.

PROOF. We prove the statement by induction on the total number of phases. In the inductive step we distinguish whether phase $\rho$ is the first phase of a busy period or not. The former case includes the induction base ($\rho = 1$), i.e., the first phase of the first busy period, as a special case.

Let $\rho \geq 1$ be the number of the phase under consideration and assume that the three statements of the theorem all hold for all preceding phases. Note that if phase $\rho$ is the first phase of a busy period, Lemma 6 can be applied.

At the beginning of phase $\rho$, DTO determines a turning point which might be replaced later on in the phase. We call the turning point $TP^\rho = (T^\rho, X^\rho)$ at which the server actually reverses direction the realized turning point of the phase $\rho$. Each turning point ever considered has a corresponding target. Note that both the realized turning point $TP^\rho$ and its corresponding target $s^\rho = (t^\rho, x^\rho)$ are reached by DTO in the same phase. Moreover, at any time when DTO is in the detour mode, the algorithm has a valid turning point and a corresponding target.

Throughout the proof we assume without loss of generality that the realized turning point is to the right of the final target, that is, in phase $\rho$, DTO moves to the right while in the detour mode and to the left after entering the focus mode. We may also assume without loss of generality that at time 0 a request appears in the origin since this request does not increase the offline cost.

Proof of Statement (a): Let $TP_0 = (T_0, X_0)$ be the first turning point chosen in phase $\rho$, $TP_1$ the next one, etc. until $TP^\rho$, the realized turning point of the phase. Let $s_i = (t_i, x_i)$ be the target corresponding to $TP_i$. Part (a) is proven by induction on the number of turning points in phase $\rho$.

1. Phase $\rho$ is the first phase of a busy period.

The first target $s_0 = (t_0, x_0)$ must be among the requests whose release initiates the start of the busy period at time $t_0$, and $TP_0 = (T_0, X_0)$ is chosen such that $t_0 + d(p^DTO(t_0), X_0) + d(x_0, X_0) \geq t_0 + 3G(t_0)$. Since $d(p^DTO(t_0), X_0) < d(x_0, X_0)$, it readily follows that $d(X_0, x_0) > \frac{3}{2}G(t_0)$. 

11
Assume that (a) holds for the turning points $TP_0, \ldots, TP_{\rho-1}$ of phase $\rho$. Consider the next turning point $TP_i = (T_i, X_i)$, replacing $TP_{\rho-1}$ at time $t$, with target $s_i = (t_i, x_i)$.

If $s_i$ was released at time $t_0$, then the turning point $TP_i$ planned by DTO at time $t$ is the same as if the guess value at time $t_0$ had already been $G(t)$ and $s_i$ had been selected as a target at time $t_0$. Exactly as before we conclude that \( d(X_i, x_i) \geq \frac{3}{2} G(t) \).

If $s_i$ was released later than $t_0$, the detour taken by DTO is longer as the one chosen if $s_i$ had been released already at time $t_0$. Hence, the arguments of above apply again and \( d(X_i, x_i) \geq \frac{3}{2} G(t) \).

2. Phase $\rho$ is not the first phase of a busy period.

For the induction base, consider $TP_0 = (T_0, X_0)$, the turning point planned first in phase $\rho$. Both $s_0$ and $T_0$ are determined in the selection point SP which marks the end of phase $\rho - 1$ and the start of phase $\rho$. Thus, $SP = (C^0_{1, DTO}, x_i)$ for some request $r_i$. When DTO serves request $r_i$, the oldest unserved request, call it $r_{z,i}$, is in its back. Observe that SP cannot be reached before the final target $s^{\rho-1}$ of phase $\rho - 1$ is served: If that was the case, there would be an unserved request in the back of DTO’s server before $s^{\rho-1}$ is reached which is older than $s^{\rho-1}$. But in that case, $s^{\rho-1}$ would have been infeasible at the time it became a target, which is a contradiction.

Assume first that $s_0$ is located between $X^{\rho-1}$ and $x_i$. Then $t_0$, the release time of $s_0$, satisfies
\[
t_0 \geq C^0_{1, DTO} - d(x_0, x_i),
\]because otherwise $s_0$ would have been served on the way to $r_i$. Since $TP_0$ is chosen at time $C^0_{1, DTO}$ in such a way that $C^0_{1, DTO} + d(x_1, X_0) + d(X_0, x_0) \geq t_0 + 3G(C^0_{1, DTO})$, and we use Inequality 5 to obtain
\[
d(X_0, x_0) \geq t_0 + 3G(C^0_{1, DTO}) - C^0_{1, DTO} - d(x_1, X_0)
\]
\[
\geq 3G(C^0_{1, DTO}) - d(x_0, x_i) - d(x_1, X_0) = 3G(C^0_{1, DTO}) - d(x_0, X_0).
\]
Hence, \( d(x_0, X_0) \geq \frac{3}{2} G(C^0_{1, DTO}) \).

Now assume that $s_0$ is further away from $r_i$ than $TP^{\rho-1}$, that is, $d(x_0, x_i) > d(X^{\rho-1}, x_i)$. Notice that $r_i$ must be older than $s_0$: If $s_0$ was older, the oldest unserved request would have been in DTO’s back before reaching $r_i$, and $(C^0_{1, DTO}, x_i)$ would not have been the selection point. Observe also that $d(X^{\rho-1}, x_i)$ is at least the distance between the realized turning point $TP^{\rho-1}$ and the corresponding target, as the final target of a phase is always served within that phase, as shown above. From the inductive hypothesis for phase $\rho$
1, Statement (a), we therefore obtain that
\[ d(x_l, x_0) > G(T^{\rho-1}). \]  
(6)

As \( d(X_0, x_0) \geq d(x_l, x_0) \), Statement (a) holds directly if \( G(T^{\rho-1}) = G(C^{\text{DTO}}_l) \). Therefore, assume that \( 2^a G(T^{\rho-1}) = G(C^{\text{DTO}}_l) \) for some integer \( a \geq 1 \). We distinguish two cases.

**Case 1:** \( t_0 \geq T^{\rho-1} \). In this case we have \( T^{\rho-1} + d(X^{\rho-1}, X_0) + d(X_0, x_0) \geq t_0 + 3G(C^{\text{DTO}}_l) \), as DTO’s server started from \( X^{\rho-1} \) at time \( T^{\rho-1} \) and chooses the turning point \( TP_0 \) at time \( C^{\text{DTO}}_l \) in such a way that the corresponding target \( s_0 \) is not served with a smaller flow time than \( 3G(C^{\text{DTO}}_l) \). Since \( d(x_0, x_l) \geq d(X^{\rho-1}, x_l) \), we also have \( d(x_0, X_0) \geq d(X^{\rho-1}, X_0) \), which yields \( 2d(x_0, X_0) \geq t_0 - T^{\rho-1} + 3G(C^{\text{DTO}}_l) \geq 3G(C^{\text{DTO}}_l) \), using that \( t_0 \geq T^{\rho-1} \). This implies the claim in Case 1.

**Case 2:** \( t_0 < T^{\rho-1} \).

Therefore, \( t_l \leq t_0 < T^{\rho-1} \) since \( r_l \) is older than \( s_0 \), as argued before. By (6) and Observation 3 (i), request \( s_0 \) is served after \( r_l \) by every \( \text{ADV} \in \text{ADV}(T^{\rho-1}) \). Hence,
\[ t_0 + \alpha_0(T^{\rho-1}) \geq t_l + \alpha_l(T^{\rho-1}) + d(x_0, x_l). \]
(7)

From the hypothesis that Statement (c) holds true for phase \( \rho - 1 \), we know that \( r_l \) is served in time, i.e.,
\[ C^{\text{DTO}}_l \leq t_l + \alpha_l(T^{\rho-1}) + 3G(T^{\rho-1}), \]
(8)

because \( T^{\rho-1} \) was the last time DTO turned around before it served \( r_l \), and \( t_l \leq T^{\rho-1} \). Hence, if DTO’s server turned around immediately after serving \( r_l \), then
\[
C^{\text{DTO}}_0 \leq C^{\text{DTO}}_l + d(x_0, x_l) \\
\leq t_l + \alpha_l(T^{\rho-1}) + 3G(T^{\rho-1}) + d(x_0, x_l) \quad \text{by (8)} \\
\leq t_0 + \alpha_0(T^{\rho-1}) + 3G(T^{\rho-1}) \quad \text{by (7)} \\
\leq t_0 + 4G(T^{\rho-1}) \leq t_0 + 2^{-a+2}G(C^{\text{DTO}}_l).
\]

Thus, \( s_0 \) would be served with a flow time of at most \( 2G(C^{\text{DTO}}_l) \), because \( a \geq 1 \). Since DTO never plans its turning point in such a way that the target is reached with a flow time of less than three times the current guess value, we can deduce that the server does in fact not turn around at time \( C^{\text{DTO}}_l \), but can spend another \( (3-2^{-a+2})G(C^{\text{DTO}}_l) \) time units on a detour. Thus, the distance between \( x_l \) and the turning point \( TP_0 \) planned at time \( C^{\text{DTO}}_l \) is at
least $\frac{1}{2}(3 - 2^{-a+2})G(C_i^{DTO})$, and we conclude that
\[
d(X_0, x_0) = d(X_0, x_i) + d(x_i, x_0) \\
\geq \frac{1}{2} \left(3 - 2^{-a+2}\right) G(C_i^{DTO}) + d(x_i, x_0) \\
\geq \left(1 - 2^{-a}\right) G(C_i^{DTO}) + G(T^{\rho-1}) \\
= \left(1 - 2^{-a}\right) G(C_i^{DTO}) + 2^{-a}G(C_i^{DTO}) \geq G(C_i^{DTO}),
\]
which proves the remaining case of the induction base.

Assume now that (a) holds for the turning points TP_0, . . . , TP_{i-1} of phase $\rho$, and consider the next turning point TP_i = (T_i, X_i) with target $s_i = (t_i, x_i)$. Assume that TP_i replaces TP_{i-1} at time $t$ of phase $\rho$, and let $t' < t$ be the time when TP_{i-1} was valid for the first time. Recall that we assumed that TP_i is to the right of $s_i$.

If $s_i = s_0$, then $G(t) \geq 2G(t')$ because the turning point changes but the target does not. That is, DTO has an extra $3(G(t) - G(t'))$ time units to spend on its detour. Hence, $d(X_i, X_{i-1}) \geq \frac{3}{2}(G(t) - G(t')) \geq G(t) - G(t')$. By the induction hypothesis, $d(X_{i-1}, x_0) \geq G(t')$. Consequently,
\[
d(X_i, x_0) = d(X_i, X_{i-1}) + d(X_{i-1}, x_0) \geq G(t) - G(t') + G(t') = G(t).
\]

If $s_i \neq s_0$ and the new target $s_i$ was released later than time $T^{\rho-1}$, then independent of the relative position $s_i$ w.r.t $T^{\rho-1}$, we can conclude as in the proof for TP_0, that $d(X_i, x_i) \geq \frac{3}{2}G(t)$.

If $s_i \neq s_0$ and $s_i$ had already been released at time $T^{\rho-1}$, then $s_i$ must have been infeasible at time $C_i^{DTO}$ when $s_0$ was chosen as target, as $s_i$ was more critical than $s_0$ at that time: otherwise it could not have become a target later on. Hence, there exists a request $r_j$ older than $s_i$ and to the right of $s_i$’s hypothetical turning point TP_i = (T_i, X_i) considered at time $C_i^{DTO}$.

By exactly the same arguments used for TP_0 above, we can deduce for the hypothetical turning point TP_i considered at time $C_i^{DTO}$ that
\[
d(X'_i, x_i) \geq G(C_i^{DTO}). \tag{9}
\]

If $G(C_i^{DTO}) = G(t)$, we obtain that $d(X_i, x_i) > d(X'_i, x_i) \geq G(C_i^{DTO}) = G(t)$. If $G(t) \geq 2G(C_i^{DTO})$, then DTO has at least $3(G(t) - G(C_i^{DTO}))$ time units more to spend on its detour to $x_i$ than it would have had if $s_i$ had become the target at time $C_i^{DTO}$. Hence, $d(X'_i, X_i) \geq \frac{3}{2}(G(t) - G(C_i^{DTO})) \geq G(t) - G(C_i^{DTO})$, which, together with (9) yields
\[
d(X_i, x_i) = d(X_i, X'_i) + d(X'_i, x_i) \geq G(t) - G(C_i^{DTO}) + G(C_i^{DTO}) = G(t).
\]
This proves the inductive step.

Notice that exactly the same arguments apply whenever a request is not made a target because it is not feasible: its hypothetical turning point considered at that time \( t \) must be at least at distance \( G(t) \) to the corresponding target.

**Proof of Statements (b) and (c):** Let SP denote the selection point which defines the end of the previous phase. If no previous phase exists, we define \( \text{SP} = (0, 0) \). By definition, \( \text{SP} = (C^{\text{DTO}}_l, x_l) \) for some request \( r_l \). We distinguish two cases: in Case I we consider the situation that DTO’s server immediately turns around in the selection point, not entering the detour mode, in Case II, we assume that it enters the detour mode at the selection point. Furthermore, we partition the set of requests served in phase \( \rho \) into three classes, defined by the part of DTO’s route in which they are served: Class 1 contains all requests served between the realized turning point \( TP^\rho \) and its corresponding target \( s^\rho \), Class 2 consists of those requests served in phase \( \rho \) after \( s^\rho \). All requests served between SP and \( TP^\rho \) belong to Class 3 (see Figure 3).

![Fig. 3. Classification of the requests served in phase \( \rho \).](image)

Let \( r_z \) be the oldest unserved request at time \( C^{\text{DTO}}_l \). As before, denote by \( TP_0 = (T_0, X_0) \) the turning point chosen at time \( C^{\text{DTO}}_l \), and by \( s_0 = (t_0, x_0) \) its corresponding target.

**Case I: DTO turns around in the selection point.**

1. **Phase \( \rho \) is the first phase of a busy period.**

   All requests served in phase \( \rho \) are released not earlier than time \( t_0 \), and since DTO immediately enters the focus mode at the beginning of the phase, they are all served without any detour. By Lemma 6, \( d(p^{\text{DTO}}(t_0), p^{\text{ADV}}(t_0)) \leq \frac{5}{2} G(t_0) \). Therefore, DTO reaches all requests served in phase 1 at most \( \frac{5}{2} G(t_0) \) time units later than the adversary, hence all requests are served in time.

2. **Phase \( \rho \) is not the first phase of a busy period.**
TP₀ = SP = (Cᵢ^{DTO}, xᵢ), because the server turns around immediately as it cannot serve s₀ with a flow time of 3G(Cᵢ^{DTO}) or less. Thus, TP₀ = TPᵢ = (Tᵢ, Xᵢ), s₀ = sᵢ, and Cᵢ^{DTO} = Cᵢ^{DTO} + d(x₀, xᵢ). Note that Class 3 is empty in this case, and that t₀ ≤ tᵢ. If s₀ was older than rᵢ, DTO would not have served rᵢ anymore, as it only remains in the focus mode until the oldest unserved request is in its back. Moreover, by Statement (a), d(xᵢ, x₀) = d(Xᵢ, x₀) ≥ G(Tᵢ) = G(Cᵢ^{DTO}).

Therefore, by Observation 3 (i), in all offline solutions in \( \mathcal{ADV}(Tᵢ) \), request rᵢ must be served before s₀. It is easy to see that \( τᵢ ≤ τ₀ = Tᵢ \), and since Statement (c) for phase \( ρ − 1 \) tells us that rᵢ is served in time, we can apply the In-Time-Lemma and conclude that s₀ is also served in time. Notice that exactly the same arguments apply to all requests in Class 2. This ensures Statement (c).

It remains to consider all other requests of Class 1. To this end, let rⱼ be a request served between rᵢ and s₀ by DTO. If rⱼ was more critical than s₀, then \( tⱼ ≤ t₀ \), because rⱼ is to the right of s₀. Since rⱼ was not chosen as target at time \( Cᵢ^{DTO} \), it must have been infeasible, that is, there is a request rᵦ older than rⱼ and to the right of rⱼ’s hypothetical turning point TPⱼ. But as rⱼ is more critical than s₀, and DTO turns around immediately to serve s₀, we have that TPⱼ = TPᵢ = Tᵢ, which implies that also s₀ cannot have been feasible at time \( Cᵢ^{DTO} \), a contradiction. Thus, rⱼ must be less critical than s₀. Hence, \( Fⱼ^{DTO} ≤ F₀^{DTO} ≤ 4G(Tᵢ) ≤ 4G(Cᵢ^{DTO}) \), proving Case I.

**Case II: DTO enters the detour mode at the selection point.**

The key arguments used in this case are similar to the ones in Case I, but more involved: With S defined as the set of all requests ever marked as a target, except the final target, we show that each request in S is served with a flow time at most 3G(Tᵢ). This allows us to deduce that requests which are less critical than those in S are served with smaller flow times. In order to show that other requests rⱼ are served with the desired flow time, we apply the In-Time-Lemma with a careful choice of the request rᵢ, which is served before rⱼ by ADV. Observation 3 will be used to determine a suitable rᵢ. Another helpful ingredient is the following: if \( d(p^{DTO}(Tᵢ), p^{ADV}(Tᵢ)) ≤ 3G(Tᵢ) \), then all requests served by both servers after time \( Tᵢ \) are served in time (ADV ∈ \( \mathcal{ADV}(Tᵢ) \)).

Since the proof for this case is largely the same for whether phase \( ρ \) is the first phase of a busy period or not, we only make the distinction when needed.

We start with the requests in Class 1. Let us first consider an arbitrary request sᵢ which was ever marked as target during the current phase \( ρ \) but did not become the final target. We prove the stronger statement that sᵢ is served with a flow time \( Fᵢ^{DTO} ≤ 3G(Tᵢ) \). Consider the time \( Tᵢ \) at which DTO reverses direction. If sᵢ was still the target at \( Tᵢ \), and TPᵢ = (Tᵢ, Xᵢ) the corresponding turning point valid at that time, then \( Fᵢ^{DTO} \) would be 3G(Tᵢ). As sᵢ was not
the target at time $T^\rho$ anymore, the final target $s^\rho$ must be more critical than $s_i$. Hence, the realized turning point $TP^\rho = (T^\rho, X^\rho)$ must be closer to the selection point $SP = (C^\rho_{iD}, x_i)$ than the point $TP_i = (T_i, X_i)$, since DTO must turn around earlier for the more critical request $s^\rho$. Consequently, $s_i$ is reached even earlier than if it was not replaced. We can conclude that in both cases, $F^\rho_{iD} \leq 3G(T^\rho)$.

**Conclusion 1** Let $S$ be the set of all requests which were ever marked as target during the current phase except for the final target. For all $r_j \in S$, $F^\rho_j \leq 3G(T^\rho)$.

From this we can conclude that the oldest request $r_z$ is also served with a flow time of at most $3G(T^\rho)$, since it is less critical than the initial target $s_0$ of this phase: if it was more critical, it would have been selected as target at time $C^\rho_{iD}$ instead of $s_0$ (as the oldest unserved request overall $r_z$ cannot be infeasible).

**Conclusion 2** The oldest unserved request $r_z$ has $F^\rho_z \leq 3G(T^\rho)$.

Before we consider the final target and requests from Class 1 which are less critical than the final target, let us show that any request $r_i = (t_i, x_i)$ in Class 1 which is more critical than the final target is served in time. As a member of Class 1, request $r_i$ must lie between the turning point and the final target. Since it is also more critical than $s^\rho$, it must be older than $s^\rho$. Consider the time $t \leq T^\rho$ at which the candidate setup was performed last during phase $\rho$. By definition, $s^\rho$ was either made a target at time $t$ or it remained the target valid at that time. Hence, $s^\rho \in V(s_0, p^\rho_{iD}(C^\rho_{iD}, G(t))$, the critical region valid at time $t$. Since $r_i$ is more critical than $s^\rho$ and also closer to the selection point, it must also be inside $V(s_0, p^\rho_{iD}(C^\rho_{iD}, G(t))$. Hence, the only reason why $r_i$ was not made a target at time $t$ is that it was infeasible. Notice that this also holds if $t = C^\rho_{iD}$ and no critical region had yet been defined when $s^\rho$ was marked as a target.

As no further candidate setup takes place, the guess value does not change after time $t$, so $r_i$ is still infeasible at time $T^\rho$. This means that there exists a request $r_b = (t_b, x_b)$ between the hypothetical turning point $TP_b = (T_b, X_b)$ corresponding to $r_i$ and $TP^\rho$, which is older than $r_i$. By Statement (a), this implies that $d(X_b, x_i) \geq G(T^\rho)$. Hence, as $r_b$ is even further away from $X_b$ than $X_b$, we deduce that $d(x_b, x_i) \geq G(T^\rho)$, which by Observation 3 (i) implies that $r_b$ must be served before $r_i$ in any offline solution in $ADV(T^\rho)$. Observe that the oldest request $r_z$ must lie between $r_b$ and $r_i$ as $r_i$ is more critical and younger than $r_z$. Hence, $r_i$ is served after $r_z$ in any such offline solution. Clearly, $C^\rho_{iD} = C^\rho_{zD} + d(x_z, x_i)$, and as $t_i \leq T^\rho$, we have that $t_i = \tau_z = T^\rho$. Hence, we can apply the In-Time-Lemma to deduce from Conclusion 2 that also $r_i$ is served in time.
**Conclusion 3** All requests $r_i$ of Class 1 which are more critical than the final target are served in time.

Before continuing the proof for Class 1, let us briefly consider a subclass of Class 2. To this end, let $r_j$ be a request of Class 2 which is released by time $T^\rho$ and more critical than the final target $s^\rho$. Again, consider the last time $t \leq T^\rho$ at which a candidate setup was performed. As above, we need to investigate why $r_j$ was not made a target at time $t$. In this case, it was either infeasible or outside the critical region $V(s_0, p^{\text{DTO}}(C^l_\rho), G(t))$ valid at time $t$. If it was infeasible, the same argument as above proves that $r_j$ is served in time (note that again, $\tau_j = \tau_z = T^\rho$). Now suppose that $r_j$ is outside $V(s_0, p^{\text{DTO}}(C^l_\rho), G(t))$. By definition of $t$, we have that $V(s_0, p^{\text{DTO}}(C^l_\rho), G(t))$ is still valid at time $T^\rho$, so by Observation 3 (ii), $r_j$ must be served after $s_0$ in all offline solutions in $\text{ADV}(T^\rho)$. Since DTO serves $r_j$ immediately after $s_0$, Conclusion 1 and the fact that $\tau_0 = \tau^*_j = T^\rho$ allow us to apply the In-Time-Lemma and deduce that $r_j$ is served in time.

**Conclusion 4** All requests $r_j$ of Class 2 which are more critical than the final target and released no later than $T^\rho$ are served in time.

Now consider the final target $s^\rho$. Clearly, if DTO does not turn around immediately when marking $s^\rho$ as a target, then $s^\rho$ is served with flow time $3G(T^\rho)$. So assume that DTO enters the focus mode at the time it marks $s^\rho$ as target. We need to distinguish two subcases: (i) $t^\rho < T^\rho$, and (ii) $t^\rho = T^\rho$.

Consider first case (i). Let $t' < T^\rho$ be the last time before $T^\rho$ at which a candidate setup was performed. Hence, $t^\rho \leq t'$. Let TP$' = (T', X')$ be the turning point chosen at time $t'$. Since $s^\rho$ was not made a target at time $t'$, it was either infeasible at time $t'$ or feasible but outside the critical region valid at time $t'$.

If it was infeasible, there must be a request $r_b = (t_b, x_b)$ older than $s^\rho$ yet unserved at time $t'$ and which lies to the right of $s^\rho$’s hypothetical turning point considered at time $t'$. Since a target candidate setup is also performed whenever DTO serves a request while in the detour mode, form the definition of $t'$ and by the assumption that $r_b$ is yet unserved at time $t'$, it follows that $C^\text{DTO}_b \geq t'$. On the other hand, we assumed that the time $T^\rho$ at which DTO turns around is the time at which it marks $s^\rho$ as a target. Since $s^\rho$ must be feasible at that time, request $r_b$ must have been served by then, which means that $C^\text{DTO}_b \leq T^\rho$. Hence, $(C_B^\text{DTO}, x_b) = (T^\rho, X^\rho)$. Statement (a) implies $d(x_b, x^\rho) \geq G(T^\rho)$, so by Observation 3 (i), $s^\rho$ is served after $r_b$ in all offline solutions in $\text{ADV}(T^\rho)$. But then, $s^\rho$ must be served after $r_z$ in all such offline solutions as well, since $r_z$ lies between $r_b$ and $s^\rho$ and is older than both. Since $\tau^\rho = \tau_z = T^\rho$, Conclusion 2 and the In-Time-Lemma imply that $s^\rho$ is served in time in that case.
We now consider the case that \( s^o \) was feasible but outside the critical region at time \( t' \). Since \( s^o \) was marked as target at time \( T^o \), it was inside the critical region valid at time \( T^o \), and we deduce that \( 2G(t') \leq G(T^o) \). Since \( s^o \) was feasible at time \( t' \), it would have satisfied the preconditions of Conclusion 3 or 4 if the sequence had ended at time \( t' \). Consequently, the turning point TP' chosen at time \( t' \) was chosen in such way that \( s^o \) would be served with a flow time of \( F^o_{DTO} \leq 4G(t') \).

Thus, for all \( t \in [t', T^0) \), \( t + d(p_{DTO}(t), X') + d(X', x^o) \leq t^o + 4G(t') \). This implies that \( T^o + d(p_{DTO}(T^o), x^o) \leq t^o + 4G(t') \leq t^o + 2G(T^o) \), contradicting the assumption that DTO had to turn around immediately at time \( t^o \). Notice that we showed that in the case that \( t^o < T^o \), the final target \( s^o \) must have been infeasible at the last time \( t' < T^o \) at which a target candidate setup was performed.

Now consider case (ii), where \( s^o \) is made a target at its release time: \( t^o = T^o \). Let us first investigate the length of the detour made by DTO and prove a bound on \( d(x_t, X^o) = T^o - C^o_{DTO} = t^o - C^o_{DTO} \). To this end, we make use of the assumption that DTO can not serve \( s^o \) with a flow time of \( 3G(T^o) \). Hence, \( d(X^o, x^o) > 3G(T^o) \). On the other hand, \( d(x^o, x_0) \leq G(T^o) - (t^o - t_0) \) since \( s^o \) is inside the critical region valid at time \( T^o \) and younger than \( s_0 \). Putting the two inequalities together, we obtain

\[
d(x_0, X^o) = d(x^o, X^o) - d(x^o, x_0) > 2G(T^o) + t^o - t_0. \tag{10}\]

Making use of the fact that DTO serves \( s_0 \) with flow time \( F^o_{DTO} \leq 3G(T^o) \) (Conclusion 1), we have \( C^o_{DTO} + d(x_t, X^o) + d(X^o, x_0) \leq t_0 + 3G(T^o) \). Therefore,

\[
d(x_t, X^o) \leq t_0 + 3G(T^o) - C^o_{DTO} - d(X^o, x_0) < t_0 + 3G(T^o) - C^o_{DTO} - 2G(T^o) - t^o + t_0 \tag{by (10)}
\]

\[
= G(T^o) + (t_0 - C^o_{DTO} - t^o + t_0) \leq G(T^o) - t^o + C^o_{DTO} \leq G(T^o) - d(x_t, X^o) \] as \( t_0 \leq C^o_{DTO} \).

We thus obtain that

\[
d(x_t, X^o) = T^o - C^o_{DTO} \leq \frac{1}{2} G(T^o). \tag{11}\]

Recall that \( S \) is the set of requests which contains all requests in Class 1 that were ever marked as target, except for \( s^o \) itself. We showed before that each request \( r_j \in S \cup \{r_j\} \) has \( F^o_{DTO} \leq 3G(T^o) \) (Conclusions 1 and 2), in particular in time. Furthermore, each such \( r_j \) has been released before time \( T^o \), which is the last time DTO reverses direction before serving that request. Hence, if for every offline solution in \( ADV(T^o) \) there exists an \( r_j \in S \cup \{r_j\} \) which is served before \( s^o \), then the In-Time-Lemma yields that \( s^o \) is served in time.
Now consider an arbitrary, but fixed \( \mathbf{ADV} \in \mathcal{ADV}(T^\rho) \) in which \( s^\rho \) is served before all requests in \( S \cup \{r_z\} \). Recall that we assumed that \( X^\rho \) is to the right of \( x^\rho \). At time \( T^\rho \), \( \mathbf{ADV} \)'s server must be located left of all requests in \( S \cup \{r_z\} \), and all these requests are yet unserved by \( \mathbf{ADV} \). Our aim is to show for the completions time of \( s^\rho \) that

\[
C^\text{DTO}_\rho \leq C^\text{ADV}_\rho + 3G(T^\rho). \tag{12}
\]

Alas, in one subcase we will only prove the weaker claim \( C^\text{DTO}_\rho \leq t^\rho + 4G(T^\rho) \).

However, we will deduce in that subcase that there is a request in Class 2 which is served in time by \( \text{DTO} \). In all other cases, (12) holds, i.e., the final target is reached by \( \text{DTO} \) no later than \( 3G(T^\rho) \) time units after the adversary reaches it in the considered offline solution, and as we considered an arbitrary \( \mathbf{ADV} \in \mathcal{ADV}(T^\rho) \), we can deduce that \( \text{DTO} \) serves \( s^\rho \) in time. Hence if Class 2 is empty, (12) yields statement (c).

Consider \( p^\text{ADV}(T^\rho) \), which by our assumption is further to the left than any point in \( S \cup \{r_z\} \). Since the adversary is non-abusive, there must be a request \( r_k \) left of its position at time \( T^\rho \) which it just served or which it is heading to. This request \( r_k \) must have release time \( t_k < T^\rho \). Another case distinction is needed.

**Case (α):** \( r_k \) was already served by \( \text{DTO} \) by time \( T^\rho \).

Notice that this situation can not occur if we are in the first phase of the first busy period. Hence we may assume that \( \rho \geq 2 \). We show that in this case,

\[
d(p^\text{ADV}(T^\rho), x_l) \leq \frac{5}{2}G(T^\rho). \tag{13}
\]

This, together with (11) then yields \( d(p^\text{DTO}(T^\rho), p^\text{ADV}(T^\rho)) \leq 3G(T^\rho) \), from which (12) easily follows, as \( \text{DTO} \) immediately heads to serve \( s^\rho \) at time \( T^\rho \), while \( \mathbf{ADV} \) cannot proceed to serve \( s^\rho = (t^\rho, x^\rho) \) before \( t^\rho = T^\rho \).

By assumption, \( r_k \) is served before \( r_l \) by \( \text{DTO} \), and by Statement (b) for phase \( \rho - 1 \) applied to \( r_l \) we have the inequality

\[
t_k + d(x_k, x_l) \leq t_l + 4G(C^\text{DTO}_l) \leq t_l + 4G(T^\rho). \tag{14}
\]

If the adversary serves \( r_k \) after \( r_l \), then \( t_l + d(x_k, x_l) \leq t_k + G(T^\rho) \), and together with (14), we obtain

\[
d(x_k, x_l) \leq t_k + G(T^\rho) - t_l \leq t_l + 4G(T^\rho) - d(x_k, x_l) + G(T^\rho) - t_l = 5G(T^\rho) - d(x_k, x_l).
\]

This yields (13).

Now consider the case that \( \mathbf{ADV} \) serves \( r_k \) before \( r_l \). Since the adversary is heading to or just coming from \( r_k \) at time \( T^\rho \) and is still left of \( S \), this means
that it hasn’t served \( r_l \) yet at time \( T^p \geq C^\text{DTO}_l \geq t_l \). Hence, at time \( T^p \), its server must be within range \( G(T^p) \) from \( x_l \), so in particular (13) holds.

Notice that, in the previous line of reasoning, we did neither make use of the assumption that DTO serves the final target with flow time more than \( 3G(T^p) \), nor that \( t^p = T^p \).

**Case (\( \beta \))**: \( r_k \) has not been served yet by DTO at time \( T^p \).

Recall that we are in the situation that the final target \( s^p \) was made a target by DTO at its release time \( t^p = T^p \), and that the server turns around immediately at that time. Let \( t' \) be the last time strictly before time \( T^p \) at which a target candidate setup is performed by DTO. As \( t_k < T^p \), and since a target candidate setup is performed whenever a new request is released, we have that \( t_k \leq t' \).

Note that request \( r_k \) must be served by DTO in the current phase \( \rho \). If it wasn’t served in phase \( \rho \), there would be an older request which remains unserved at least until time \( T^p \) and which is in the server’s back after it turned in \( T^p \). But then, this request must be older than \( s^p \) and would have caused \( s^p \) to be infeasible.

Let \( s_m \) be the target valid at time \( t' \) (after the target selection), and \( T_{\text{P}}_m \) the corresponding turning point. In the proof for Conclusions 1 and 2 we showed that for all \( r_j \in S \cup \{r_z\} \), their flow time satisfies \( F^\text{DTO}_{j} \leq 3G(t') \) if \( T_{\text{P}}_m \) is also the realized turning point. We are in the situation that \( T_{\text{P}}_m \) is replaced at time \( T^p \) by \( T^p \). But as DTO turns around immediately when replacing \( T_{\text{P}}_m \), it turns earlier than planned and hence serves the requests \( r_j \in S \cup \{r_z\} \) even earlier, in particular with a flow time of \( F^\text{DTO}_{j} \leq 3G(t') \).

Now consider request \( r_k \). There are three possible reasons why \( r_k \) is not selected as target at time \( t' \): (i) it is less critical than \( s_m \), (ii) \( r_k \) is more critical than \( s_m \) but infeasible at time \( t' \), or (iii) it is more critical than \( s_m \) but outside the critical region valid at that time.

In case (i), i.e. if \( r_k \) is less critical than \( s_m \), it also has \( F^\text{DTO}_{k} \leq 3G(t') \leq 3G(T^p) \). In particular, \( C^\text{DTO}_{k} \leq C^\text{ADV}_k + 3G(T^p) \) for the considered ADV \( \in \text{ADV}(T^p) \). Furthermore, \( s^p \) is more critical than \( r_k \) and younger. Therefore, it must be to the left of \( r_k \), which in turn was left of \( p^\text{ADV}(T^p) \). As a consequence, \( s^p \) is served after \( r_k \) by ADV, and we conclude \( C^\text{DTO}_{\rho} = C^\text{DTO}_k + d(x^p, x_k) \leq C^\text{ADV}_k + 3G(T^p) + d(x_k, x^p) \leq C^\text{ADV}_\rho + 3G(T^p) \), which was our claim (12).

Now let us investigate case (ii), in which \( r_k \) is infeasible at time \( t' \). That means that there exists a request \( r_b = (t_b, x_b) \) older than \( r_k \) and to the right of the hypothetical turning point corresponding to \( r_k \) at time \( t' \). If ADV served \( r_b \) before time \( T^p \), then it must have served \( r_z \) on its way from \( r_b \) to its current position, as \( r_z \) is older than \( r_b \) and located between \( r_b \) and \( p^\text{ADV}(T^p) \). This contradicts our assumption that ADV has not served any of the requests in
$S \cup \{r_z\}$ by time $T^\rho$. Consequently, $r_b$ must be unserved by ADV at time $T^\rho$, which yields $d(p_{ADV}(T^\rho), x_b) \leq G(T^\rho)$. Since $r_b$ must be to the right of the selection point $(C_{\rho}^{DTO}, x_i)$, we obtain in particular that $d(p_{ADV}(T^\rho), x_i) \leq G(T^\rho)$, which together with (11) implies $d(p_{ADV}(T^\rho), p_{DTO}(T^\rho)) \leq \frac{4}{3}G(T^\rho)$. Hence, DTO reaches $s^\rho$ at most $\frac{3}{2}G(T^\rho)$ time units later than ADV, thus $s^\rho$ is served in time.

Finally, consider Case (iii): request $r_k$ is outside the critical region valid at time $t'$. Consequently, $r_k$ is served after $s_0$ in all offline solutions in $ADV(t')$. Recall that we showed before that in the current case, all requests $r_j \in S \cup \{r_z\}$ have $F_j^{DTO} \leq 3G(t')$. Consequently,

$$T^\rho + d(p_{DTO}(T^\rho), x_k) = C_k^{DTO} = C_0^{DTO} + d(x_0, x_k) \leq t_0 + 3G(t') + d(x_0, x_k) \leq t_k + \alpha_k(t') + 3G(t').$$

If $2G(t') \leq G(T^\rho)$, we obtain together with $t_k < T^\rho$ that $d(p_{DTO}(T^\rho), x_k) \leq 4G(t') \leq 2G(T^\rho)$, which lets us conclude that $d(p_{DTO}(T^\rho), p_{ADV}(T^\rho)) \leq 2G(T^\rho)$. Thus, $C_\rho^{DTO} \leq C_\rho^{ADV} + 2G(T^\rho)$.

If $G(t') = G(T^\rho)$, Property 2 says that $\alpha_k(t') \leq \alpha_k(T^\rho)$. Thus, $r_k$ is served in time, since $t_k = T^\rho$. If $s^\rho$ is served after $r_k$ by ADV, we conclude with the In-Time-Lemma that (12) holds. Otherwise, $s^\rho$ must be to the right of $r_k$, is therefore, as the younger one, less critical than $r_k$ and served with a flow time of at most $4G(T^\rho)$ by DTO.

Note that we proved the stronger statement that $s^\rho$ is served in time for all cases except for the case that all of the following statements are simultaneously true:

- $t^\rho = T^\rho$ and DTO turns around immediately at $t^\rho$.
- there exists ADV $\in ADV(T^\rho)$ in which $s^\rho$ is served before all requests in $S \cup \{r_z\}$.
- there is a request $r_k$ to the left of $p_{ADV}(T^\rho)$ with $t_k < T^\rho$ which is outside the critical region at the last time $t' < T^\rho$ at which a candidate setup was performed by DTO,
- $r_k$ is served after $s^\rho$ by ADV.

In that special case, it is shown that $F_\rho^{DTO} \leq 4G(T^\rho)$, and that $r_k$ is served in time by DTO.

**Conclusion 5** The final target $s^\rho$ is served either in time, or $F_\rho^{DTO} \leq 4G(T^\rho)$ and there exists a request $r_k$ in Class 2 with $t_k \leq T^\rho$ and which is served in time by DTO.

Note that this implies Statement (c) for the case that Class 2 is empty.
Finally, let \( r_i = (t_i, x_i) \) be an arbitrary request of Class 1 which was never marked as target and which is less critical than the final target \( s^{\rho} \). As it is less critical, its flow time satisfies \( F_i^{\text{DTO}} \leq F_\rho^{\text{DTO}} \), hence \( F_i^{\text{DTO}} \leq 4G(T^{\rho}) \leq 4G(C_i^{\text{DTO}}) \), which was our claim.

**Conclusion 6** Any request \( r_i \) in Class 1 which does not belong to any of the sets of requests covered by Conclusions 1–3 or by Conclusion 5 has \( F_i^{\text{DTO}} \leq 4G(T^{\rho}) \).

We now consider Class 2. To this end, let \( r_j = (t_j, x_j) \) be a request served in phase \( \rho \) after \( s^{\rho} \) by DTO. First consider the case that \( t_j \leq T^{\rho} \). We proved already that \( r_j \) is served in time if it is more critical than the final target \( s^{\rho} \) (Conclusion 4). Consider the case that \( r_j \) is less critical than \( s^{\rho} \). If \( F_\rho^{\text{DTO}} \leq 3G(T^{\rho}) \), then also \( r_j \) is served with that flow time, hence in time. So assume that \( F_\rho^{\text{DTO}} > 3G(T^{\rho}) \). Since \( r_j \) is less critical and to the left of \( s^{\rho} \), it must be strictly younger: \( t^\rho < t_j \leq T^{\rho} \). We showed before that in this case, \( s^{\rho} \) must have been infeasible at time \( t' \), the last time before \( T^{\rho} \) at which a candidate setup was performed, and that there exists a request \( r_b \) older than \( s^{\rho} \) for which \( (C_b^{\text{DTO}}, x_b) = (T^{\rho}, X^{\rho}) \). By Statement (a), we deduce that \( d(x_b, x_j) \geq d(x_a, x^\rho) = d(X^{\rho}, x^\rho) \geq G(T^{\rho}) \). As \( r_b \) is older than \( r_j \), Observation 3 (i) implies that it must be served before \( r_j \) in all offline solutions in \( \text{ADV}(T^{\rho}) \).

As reasoned before, then also \( r_z \) must be served before \( r_j \) in all such offline solutions. Since DTO serves \( r_j \) immediately after \( r_z \), and as \( \tau_z = \tau_j = T^{\rho} \), we can apply the In-Time-Lemma to conclude that \( r_j \) is served in time.

It remains to consider those requests \( r_j \in \text{Class 2} \) for which \( t_j > T^{\rho} \). Note that \( \tau_j = t_j > T^{\rho} \) in this case. Let

\[
A := \{r_z\} \cup S \cup \{r_i \in \text{Class 2} : t_i \leq T^{\rho}\}.
\]

It is easy to see that each request \( r_a \in A \) is released by time \( T^{\rho} \), hence \( \tau_a = T^{\rho} < \tau_j \). We showed before that all requests in \( A \) are served in time by DTO. Consider an arbitrary \( \text{ADV} \in \text{ADV}(\tau_j) \). Assume that there exists a request \( r_a \in A \) which is served by ADV before \( r_j \). No matter whether \( x_j \) is left of \( x_a \) or not, we have that \( C_j^{\text{DTO}} \leq C_a^{\text{DTO}} + d(x_j, x_a) \). Hence, we can apply the In-Time-Lemma and deduce that \( r_j \) is served in time.

Therefore, we can restrict our attention to the case that ADV serves all requests in \( A \) after \( r_j \). In particular, this means that \( A \) has not yet served any of the requests in \( A \) at time \( T^{\rho} \). We distinguish two subcases: (i) there is a request \( r_a \in A \) which is left of \( p^{\text{ADV}}(T^{\rho}) \), and (ii) ADV’s position at time \( T^{\rho} \) is to the left of all requests in \( A \).

In case (i), \( r_j \) cannot be to the left of \( r_a \), since otherwise it would be served after \( r_a \) by ADV, contradicting our assumption in this case. Hence, it suffices to show that \( F_j^{\text{DTO}} \leq 4G(C_j^{\text{DTO}}) \), as \( r_j \) cannot be the last request served by
DTO in the current phase. As an element of the set $A$, request $r_a$ is served by
time $t_a + 4G(T^p)$. Since $r_a$ was released before $T^p$, we have $t_a < t_j$, and because
DTO serves $r_j$ on the way to $r_a$, we obtain $C_j^{DTO} \leq C_a^{DTO} \leq t_a + 4G(T^p) \leq
\displaystyle t_j + 4G(T^p) \leq t_j + 4G(C_j^{DTO})$, which was our claim.

Now consider case (ii): ADV’s position at time $T^p$ is to the left of all requests
in $A$. Since the adversary is non-abusive, there must be a request $r_k$ to its
left which it just served or where it is heading to at time $T^p$. In particular,
time $t_k < T^p$, from which we can deduce that $C_k^{DTO} \leq T^p$. If not, then $r_k$
would belong to the set $A$, contradicting that it is left of ADV which in turn is left of
all requests in $A$. Furthermore, $r_k$ cannot be served by DTO in a later phase:
if so, there would be an older request in the server’s back, and DTO would
have to turn around before reaching $r_j$, thus serving $r_j$ also in a later phase,
contradicting the assumption that $r_j$ is served in the current phase.

Exactly as in the proof of inequality (13) (used for the final target), we can in
this case deduce for the distance of ADV’s position at time $T^p$ to the selection
point $SP = (C_i^{DTO}, x_i)$ that

$$d(p^{ADV}(T^p), x_i) \leq \frac{5}{2} G(T^p). \quad (15)$$

Let $L := d(X^p, x_i) = T^p - C_i^{DTO}$ be the length of the detour made by DTO at
the beginning of the current phase. If $L \leq G(T^p)/2$, we obtain from (15) that
d($p^{ADV}(T^p), p^{DTO}(T^p)) \leq 3G(T^p)$, which implies that $r_j$ is served in time: ADV
cannot serve $r_j$ before time $t_j > T^p$, and DTO proceeds towards $r_j$ without
any detour after time $T^p$.

So assume that $L = T^p - C_i^{DTO} > G(T^p)/2$. Since DTO serves $r_z$ with flow
time $F_z^{DTO} \leq 3G(T^p)$, we obtain $C_i^{DTO} + L + d(X^p, x_z) \leq t_z + 3G(T^p)$, which
implies

$$d(X^p, x_z) \leq t_z + 3G(T^p) - L - C_i^{DTO}. \quad (16)$$

Moreover, by assumption, ADV serves $r_z \in A$ after $r_j$, and we have that
$T^p + d(p^{ADV}(T^p), x_j) + d(x_j, x_z) \leq t_z + G(T^p)$. In particular,

$$T^p + d(x_j, x_z) \leq t_z + G(T^p). \quad (17)$$

Note that $t_z \leq C_i^{DTO} \leq T^p < t_j$. We obtain

$$C_j^{DTO} = T^p + d(X^p, x_z) + d(x_z, x_j) \leq t_z + 3G(T^p) - L - C_i^{DTO} + t_j + G(T^p) \text{ by (16) and (17)}$$

$$= t_j + 3G(T^p) + [G(T^p) + 2t_z - C_i^{DTO} - t_j - L] \leq t_j + 3G(T^p) + [G(T^p) + C_i^{DTO} - T^p - L] \text{ as } t_z \leq C_i^{DTO} \text{ and } t_j \geq T^p$$

$$= t_j + 3G(T^p) + [G(T^p) - 2L] \text{ as } L = T^p - C_i^{DTO} < t_j + 3G(T^p) \text{ by the assumption that } L > G(T^p)/2.$$
Hence, in this case, $F_{j}^{DTO} \leq 3G(T^\rho)$, and in particular $r_j$ is served in time.

**Conclusion 7** Each request $r_j$ in Class 2 is either served in time, or there exists a request $r_k$ served later in the phase which is served in time, while $F_{j}^{DTO} \leq 4G(C_{j}^{DTO})$.

Note that this implies Statement (c) for Case II if Class 2 is non-empty.

Finally, we consider the requests in Class 3: Let $r_k$ be served between SP and TP$^\rho$. Since the oldest request $r_z$ lies in the server’s back at time $C_{i}^{DTO}$, request $r_k$ is younger than $r_z$. At time $C_{k}^{DTO}$, DTO is either in the detour mode and has a valid turning point TP$^m$, or it has already turned in TP$^\rho$. In the first case, $C_{2}^{DTO} \leq t_z + 3G(C_{k}^{DTO})$ if TP$^m$ wasn’t replaced, as shown before. Therefore,

$$C_{k}^{DTO} + d(x_k, TP_m) + d(TP_m, x_z) \leq t_z + 3G(C_{k}^{DTO}) \leq t_k + 3G(C_{k}^{DTO}).$$

In the second case, $C_{k}^{DTO} \geq T^\rho$, and as $F_{z}^{DTO} \leq 3G(T^\rho)$, we conclude that $C_{k}^{DTO} + d(x_k, x_z) \leq t_z + 3G(T^\rho) \leq t_k + 3G(C_{k}^{DTO})$.

**Conclusion 8** Each request $r_k$ in Class 3 has flow time $F_{k}^{DTO} \leq 3G(C_{k}^{DTO})$.

This completes the proof of Theorem 7. □

**Theorem 8** DTO is $8$-competitive against a non-abusive adversary for the $F_{\text{max}}$-OLTSP.

**PROOF.** By Theorem 7 we have $F_{i}^{DTO} \leq 4G(C_{i}^{DTO})$ for any request $r_i$. If $C_{last}^{DTO}$ is the time at which the last request is served by DTO, then all requests are served with flow time at most $4G(C_{last}^{DTO})$, which, by construction, is bounded by $8\text{OPT}(\sigma)$, thence the claim. □

**References**


