Online Routing Problems: Value of Advanced Information as Improved Competitive Ratios

Patrick Jaillet * and Michael R. Wagner †

April 2005; Revised November 2005, January 2006

Abstract

We consider online versions of the Traveling Salesman Problem (TSP) and Traveling Repairman Problem (TRP) where instances are not known in advance. Cities (points) to be visited are revealed over time, while the server is en route serving previously released requests. These problems are known in the literature as the Online TSP (TRP, respectively). The corresponding offline problems are the TSP (TRP) with release dates, problems where each point has to be visited at or after a given release date. In the current literature, the assumption is that a request becomes known at the time of its release date. In this paper we introduce the notion of a request's *disclosure date*, the time when a city's location and release date are revealed to the server. In a variety of disclosure date scenarios and metric spaces, we give new online algorithms and quantify the value of this advanced information in the form of improved competitive ratios. We also provide a general result on polynomial-time online algorithms for the online TSP.

1 Introduction

The Traveling Salesman Problem (TSP) is one of the most intensely studied problems in optimization. In one of its simplest forms we are given a metric space and a set of points in the space, representing cities. Given an origin city, the task is to find a tour of minimum total length, beginning and ending at the origin, that traverses each city at least once. Assuming a constant speed, we can interpret this objective as minimizing the *time* required to complete a tour. The Traveling Repairman Problem (TRP) is defined similarly, only that we are interested in minimizing the weighted sum of city completion times, where a city's completion time is the first time that a city is visited; this objective is also referred to as the *latency*. These two objectives embody important but very

*Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, jaillet@mit.edu.

†Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA 02139, mikew@mit.edu.

different managerial measures. The TSP objective is closely related to the notion of makespan, the maximum completion date of all cities; this measure is traditionally used if one were to optimize with the server's interest in mind. Alternatively, the latency is closely related to the (weighted) average completion date of all cities, which clearly has the customers' interest in mind. In both problems we may also incorporate release dates, where a city must be visited on or after its release date; in this case the problems are known as the "TSP with release dates" and the "TRP with release dates," respectively.

The assumption that problem instances are completely known a priori is unrealistic in many applications. Taxi services, buses and courier services, for example, require an *online* model in which cities (points) to be visited are revealed over time, while the server is en route serving previously released requests. The focus of this paper is on studying algorithms for the online TSP and TRP. They are evaluated using the competitive ratio, which is defined as the worst case ratio of the online algorithm's cost to the cost of an optimal offline algorithm.

1.1 Literature Review

The literature for the TSP is vast. The interested reader is referred to the books by Lawler, Lenstra, Rinnooy Kan, Shmoys [17] and Korte and Vygen [15] for comprehensive coverage of results concerning the TSP. Probabilistic versions of the TSP, where a different approach is used to represent limited knowledge of the problem instance, have also attracted interest (e.g., see Jaillet [12] and Bertsimas [4]). Offline routing problems with release dates can be found in Psaraftis, Solomon, Magnanti, Kim [20] and Tsitsiklis [22]. We also mention two offline results that will play a part in our analysis: the 3/2-approximation algorithm for the TSP in metric space by Christofides [8] and the polynomial-time approximation scheme for the TSP in Euclidean space by Arora [1].

A systematic study of online algorithms was given by Sleator and Tarjan [21], who suggested comparing an online algorithm with an optimal offline algorithm. Karlin, Manasse, Rudolph, Sleator [14] introduced the notion of a *competitive ratio*. An online algorithm is said to be r-competitive ($r \geq 1$) if, given any instance of the problem, the cost of the solution given by the online algorithm is no more than r multiplied by that of an optimal offline algorithm:

$$\mathrm{Cost}_{\mathrm{online}}(I) \leq r\mathrm{Cost}_{\mathrm{optimal}}(I), \ \forall \text{ problem instances } I.$$

The infimum over all r such that an online algorithm is r-competitive is called the competitive ratio of the online algorithm. An online algorithm is said to be best-possible if there does not exist another online algorithm with a strictly smaller competitive ratio. Online algorithms have been used to analyze paging in computer memory systems, distributed data management, navigation problems in robotics, multiprocessor scheduling, etc.; see the books of Borodin and El-Yaniv [7] and Fiat and Woeginger [10] for more details and references.

Research concerning online versions of the TSP and TRP have been introduced relatively recently. Kalyanasundaram and Pruhs [13] have examined a unique version of an online traveling salesman problem where new cities are revealed locally during the traversal of a tour (i.e., an arrival at a city reveals any adjacent cities that must also be visited). More related to our paper is the stream of works which started with the paper by Ausiello, Feuerstein, Leonardi, Stougie, Talamo [3]. In this paper, the authors studied the online TSP version we consider here; they analyzed the problem on the real line and on general metric spaces, developing online algorithms for both cases and achieving a best-possible online algorithm for general metric spaces, with a competitive ratio of 2. These authors also provide a polynomial-time online algorithm, for general metric spaces, which is 3-competitive. Subsequently, Ausiello, Demange, Laura, Paschos [2] gave a polynomial-time algorithm, for general metric spaces, which is 2.78-competitive. Lipmann [18] developed a best-possible online algorithm for the real line, with a competitive ratio of approximately 1.64. Blom, Krumke, de Paepe, Stougie [5] gave a best-possible online algorithm for the non-negative real line, with a competitive ratio of $\frac{3}{2}$. This last paper also considers different adversarial algorithms in the definition of the competitive ratio.

Considering the online TRP, Feuerstein and Stougie [9] gave a lower bound of $(1 + \sqrt{2})$ for the competitive ratio and a 9-competitive algorithm, both for the online TRP on the real line. Krumke, de Paepe, Poensgen, Stougie [16] improved upon this result to give a $(1+\sqrt{2})^2$-competitive deterministic algorithm for the online TRP in general metric spaces as well as a Θ-competitive randomized algorithm, where $\Theta \approx 3.64$; in this paper, we correct this result to $\Theta \approx 3.86$. All of the aforementioned works only consider the case where a revealed city is ready for immediate service; i.e., all the disclosure dates equal their respective release dates.

1.2 Our Contributions

In this paper we introduce the notion of "disclosure dates," i.e., dates at which requests become known to the online player, ahead of the release dates (the dates at which requests can first be served). In many applications these two sets of dates do not coincide. Consider the taxi and courier examples mentioned previously; in each of these scenarios, there is the possibility of calling ahead (disclosure date) and requesting a pickup time (release date). In many cases, a fixed amount of time between a request for service and readiness exists; for example, many taxi companies usually say "it'll be 15 minutes."

In addition to providing more realism, the introduction of this advanced information is a natural mechanism to increase the "power" of online players against all-knowing adversaries in a competitive analysis framework. Note, also, that these disclosure dates provide a natural bridge between online routing problems and their offline versions – when all the disclosure dates are zero, we have the offline problems; when all the disclosure dates are equal to their respective release dates, we have the online routing problems considered so far in the literature, which we denote the *traditional* online problems. In other words, we can vary the "online-ness" of the problems with these disclosure dates.

By introducing disclosure dates, we have defined a new optimization problem: the **online TSP with disclosure dates**. We measure the quality of algorithms for this problem using the competitive ratio; the denominator of this ratio is again the optimal value of the TSP with release dates since disclosure dates are irrelevant in the offline situation. For a variety of disclosure date scenarios, we give new online algorithms and derive improved competitive ratios (with respect to the ratios for the traditional online problems), which are functions of the advanced information. In this way, we quantify the *value* of the advanced information given by the disclosure dates. We almost exclusively consider the case where there is a "fixed amount" of advanced notice for each city. In this case, we introduce α and β, two convenient problem parameters that relate the advanced information to the "dimensions" (time and space) of the traditional online problems (exact definitions of α and β will be given in Sections 3 and 5, respectively); we quantify the value of the advanced information in terms of these two parameters. We first detail our results for the online TSP. For the non-negative real line, we give an algorithm that is $\max\{1, \frac{3}{2} - \alpha\}$-competitive and we also prove that this result is best-possible for our disclosure date structure. These results improve upon the $\frac{3}{2}$-competitiveness of a best-possible online algorithm in the traditional case. For the general situation, where cities belong to an arbitrary metric space, we give an algorithm that is $(2 - \frac{\alpha}{1+\alpha})$-competitive. This result improves upon the 2-competitiveness of a best-possible online algorithm for the traditional metric case. Next, we consider the online TRP. We analyze a deterministic algorithm and show it is $((1+\sqrt{2})^2 - \frac{\alpha\beta}{\alpha+\beta})$-competitive, where $(1+\sqrt{2})^2$ is the best provable worst-case ratio to-date for the traditional online problem (though this latter result is probably not best-possible). We also give a very similar result for a randomized modification of the previous algorithm; we show this variant is $(\Theta - \frac{\alpha\beta}{\alpha+\beta})$-competitive, where Θ is the traditional competitiveness result.

Finally, we consider polynomial-time algorithms for the online TSP. We show that, if we have a ρ-approximation algorithm for the TSP, we then have a 2ρ-competitive algorithm for the online TSP. If the metric space is Euclidean, for any $\epsilon > 0$, we have a $(2+\epsilon)$-competitive polynomial-time algorithm.

Outline: The remainder of the paper is as follows: after giving basic definitions in Section 2, we first study in Section 3 the online TSP on the non-negative real line $\mathbb{R}_+$. Then, in Sections 4 and 5 we study the online TSP and TRP in general metric spaces, respectively. Finally, we study polynomial-time online algorithms for the online TSP in Section 6 and give concluding thoughts in Section 7.

2 Preliminaries

Let us first state the assumptions and definitions about the problems we consider in the paper.

1. City locations belong to some metric space $\mathcal{M}$.

2. A city is revealed to the salesman (repairman) at its disclosure date.

3. A city is ready for service at its release date. The service requirement at a city is zero.

4. The disclosure date for a given city is less than or equal to the city's release date.

5. The salesman (repairman) travels at unit speed or is idle.

6. The problem begins at time 0, and the salesman (repairman) is initially at a designated origin of the metric space.

7. The online TSP objective is to minimize the time required to visit all cities and return to the origin.

8. The online TRP objective is to minimize the weighted sum of completion times, where each city's completion time is weighted by a given non-negative number, revealed at the city's disclosure date.

The data common to both the online TSP and online TRP is a set of points $(l_i, r_i, q_i),\ i = 1, \ldots, n$, where n is the number of cities. The quantity $l_i \in \mathcal{M}$ is the i^{th} city's location. The quantity $r_i \in \mathbb{R}_+$ is the i^{th} city's release date; i.e., r_i is the first time after which that city i will accept service. The quantity $q_i \in \mathbb{R}_+$ is the i^{th} city's disclosure date; i.e., at time q_i, the salesman learns about city i's request and its corresponding values l_i and r_i. We also let $\mathcal{N} = \{1, \ldots, n\}$. We have that $r_i \geq q_i \geq 0,\ \forall i \in \mathcal{N}$. Finally, we let $w_i,\ i \in \mathcal{N}$ denote the non-negative weights on the completion times of cities for the online TRP, which become known at times q_i.

From the online perspective, the total number of requests, represented by the parameter n, is not known, and city i only becomes known at time q_i. $C_A(n)$ will denote the cost of online algorithm A on an instance of n cities and $C_{\text{OPT}}(n)$ is the optimal offline cost on n cities (at times, the n term will be suppressed). Finally, let $r_{max} = \max_{i \in \mathcal{N}}\{r_i\}$ and define L_{TSP} as the optimal TSP tour length through all cities in an instance.

3 The Online TSP on $\mathbb{R}_+$

In this section, we study the online TSP when the city locations are all on the non-negative real line; i.e., $\mathcal{M} = \mathbb{R}_+$. We begin with an offline analysis.

We consider the offline TSP with release dates on the non-negative real line. For this problem, Psaraftis et al. [20] proposed an optimal strategy:

Optimal Offline Algorithm

(1) Go directly to city $l_{max} = \max_{i \in \mathcal{N}}\{l_i\}$.

(2) Wait at city l_{max} for $\max_{i \in \mathcal{N}}\{\max\{0, r_i - 2l_{max} + l_i\}\}$ units of time.

(3) Proceed directly back to the origin.

The waiting time is calculated to ensure the salesman's return to the origin finds each city ready for service. A closed-form expression for $C_{\mathrm{OPT}}(n)$ is as follows:

$$\begin{aligned} C_{\mathrm{OPT}}(n) &= 2l_{max} + \max_{i\in\mathcal{N}}\{\max\{0, r_i - 2l_{max} + l_i\}\} \\ &= \max_{i\in\mathcal{N}}\{\max\{2l_i, r_i + l_i\}\}. \end{aligned}$$

3.1 Online Algorithms

In this subsection, we consider two online algorithms. The first considers the case $q_i = r_i,\ \forall i \in \mathcal{N}$ and was first proposed and analyzed by Blom et al. [5], under the name of Move-Right-If-Necessary. Subsequently, we present a generalization of this algorithm for the case $q_i \le r_i,\ \forall i \in \mathcal{N}$.

3.1.1 The Move-Right-If-Necessary Algorithm

We assume that $q_i = r_i,\ \forall i \in \mathcal{N}$ and we consider the following online strategy hereafter called the Move-Right-If-Necessary (MRIN) algorithm.

Algorithm MRIN

(1) If there is an unserved city to the right of the salesman, he moves towards it at unit speed.

(2) If there are no unserved cities to the right of the salesman, he moves back towards the origin at unit speed.

(3) Upon reaching the origin, the salesman becomes idle.

The cost of the MRIN algorithm on an instance of n cities is denoted by $C_{\mathrm{MRIN}}(n)$. We have the following theorem from [5].

Theorem 1 ([5]) $C_{MRIN}(n) \le \frac{3}{2}C_{OPT}(n),\ \forall n.$

We also have a hardness result which can be obtained from the analysis in [5].

Theorem 2 ([5]) *Let ρ be the competitive ratio for any deterministic online algorithm for the online TSP on $\mathbb{R}_+$. Then $\rho \ge \frac{3}{2}$.*

Thus, MRIN is a best-possible online algorithm (restricted to the case where $q_i = r_i,\ \forall i \in \mathcal{N}$).

3.1.2 The Move-Left-If-Beneficial Algorithm

We now consider the case where $q_i \le r_i,\ \forall i \in \mathcal{N}$. Notice that by ignoring the existence of requests until their release dates, MRIN can be applied again and will yield the same competitive ratio of 3/2. However, a natural adaptation of MRIN does benefit from the disclosure dates. Thus, we define the Move-Left-If-Beneficial (MLIB) algorithm.

Algorithm MLIB

(1) If there is an unserved city to the right of the salesman, he moves towards it at unit speed.

(2) If there are no unserved cities to the right of the salesman, he moves back towards the origin if and only if the return trajectory reaches all unserved cities on or after their release date; otherwise the salesman remains idle at his current location.

(3) Upon reaching the origin, the salesman becomes idle.

The cost of the MLIB algorithm on an instance of n cities is denoted by $C_{\text{MLIB}}(n)$. We would like to emphasize that the MLIB algorithm applied to an instance where $q_i = r_i,\ \forall i \in \mathcal{N}$ is indistinguishable from the MRIN algorithm applied to the same instance. In addition, the MLIB algorithm applied to an instance where $q_i = 0,\ \forall i \in \mathcal{N}$ is also indistinguishable from the optimal offline algorithm. In this sense, MLIB fully incorporates the advanced information of the disclosure dates. In the next subsections, we first analyze the MLIB algorithm for a special case and then we give a general analysis.

3.2 Equal Amounts of Advanced Notice

In this subsection, we first give some technical results for the general case. Then we introduce a special structure for the disclosure dates and we show that MLIB is best-possible while MRIN is not.

Lemma 1 $C_{MLIB}(n) \leq \max_{i \in \mathcal{N}} \{\max\{q_i + 2l_i, r_i + l_i\}\}$.

Proof Suppose $C_{\text{MLIB}}(n) = z > \max_{i \in \mathcal{N}} \{\max\{q_i + 2l_i, r_i + l_i\}\}$. Consider the final segment of the MLIB salesman's trajectory; i.e., the segment of the trajectory where the salesman returns directly to the origin without changing direction or waiting. We can fully describe this segment of the trajectory as $x_t = z - t,\ t \in [t_0, z]$ for some t_0, the time the salesman begins his final return. Note that it is possible that $t_0 = z$. We have two cases to consider at time t_0:

Case (1): At t_0^-, the salesman was moving away from the origin toward a city k and reached it at t_0 such that $t_0 \geq r_k$. City k is the rightmost unserved city at time t_0 and the salesman then starts the x_t trajectory, returning to the origin, reaching each unserved city along the way on or after its release date. Since the salesman was moving away from the origin, the worst possible location for him to be when city k was disclosed was the origin. So the salesman should arrive at city k at time no later than $q_k + l_k$. Thus $x_{t_0} = l_k$, for some $t_0 \leq q_k + l_k$, implying that $z = l_k + t_0 \leq q_k + 2l_k$, which contradicts our assumption.

Case (2): The salesman has just finished waiting at some point, possibly the origin, so that the x_t trajectory reaches all cities on or after their release date. Thus, $\exists m$ such that $x_t = l_m$, for $t = r_m$, where $r_m \in [t_0, z]$. Consequently, $z = l_m + r_m$, which again contradicts our assumption. ■

We now prove a proposition that simplifies the subsequent analysis. This proposition depends on the concept of an "ignored city," which is defined as follows: An ignored city is viewed to have never existed; i.e., it will not be taken into account when calculating the online and offline costs.

Proposition 1 *For any instance of the online TSP on $\mathbb{R}_+$ that has both a request away from the origin and a request at the origin, ignoring the latter will not decrease the ratio $C_{MLIB}(n)/C_{OPT}(n)$.*

Proof Let $\tilde{C}_{\text{MLIB}}$ denote the cost if the request at the origin was ignored. If the release date of the request at the origin is later than $\tilde{C}_{\text{MLIB}}$, the proposition is trivially true. Otherwise, the behavior of MLIB is not affected by the request, but the optimal solution value may decrease by deleting it. ■

When all cities are located at the origin, we have that $C_{\text{MLIB}}(n) = C_{\text{OPT}}(n)$. The above proposition allows us to make the following assumption without a loss of generality (for our intention of proving upper bounds on competitive ratios).

Assumption 1 $l_i > 0$ *for all* $i \in \mathcal{N}$.

We now consider the situation where the online salesman receives a fixed amount of advanced notice for each city in a problem instance. In particular, there exists a constant $a \in [0, r_{max}]$ such that

$$q_i = (r_i - a)^+, \ \forall i \in \mathcal{N},$$

where $(x)^+ = \max\{x, 0\}$. Noting that $L_{TSP} = 2l_{max}$, we have the following theorem.

Theorem 3 $C_{MLIB}(n) \leq \max\left\{1, \dfrac{3}{2} - \alpha\right\} C_{OPT}(n)$, *where* $\alpha = \dfrac{a}{L_{TSP}}$.

Proof From Lemma 1, we have that

$$C_{\text{MLIB}}(n) \leq \max_{i \in \mathcal{N}} \left\{\max\left\{q_i + 2l_i, r_i + l_i\right\}\right\}. \tag{1}$$

Define $\mathcal{S} = \{i \in \mathcal{N} \mid q_i > 0\}$; note that for $i \in \mathcal{S}$, $q_i = r_i - a$. If $\mathcal{S} = \emptyset$, $C_{\text{MLIB}}(n) = C_{\text{OPT}}(n)$ trivially. Otherwise , we write the RHS of Equation (1) as

$$\max\{\max_{i \in \mathcal{S}} \left\{\max\left\{q_i + 2l_i, r_i + l_i\right\}\right\}, \max_{i \in \mathcal{N} \setminus \mathcal{S}} \left\{\max\left\{2l_i, r_i + l_i\right\}\right\}\},$$

which is less than or equal to $\max\{\max_{i \in \mathcal{S}} \left\{\max\left\{q_i + 2l_i, r_i + l_i\right\}\right\}, C_{\text{OPT}}(n)\}$. Let us assume $\max_{i \in \mathcal{S}} \left\{\max\left\{q_i + 2l_i, r_i + l_i\right\}\right\} > C_{\text{OPT}}(n)$; otherwise $C_{\text{MLIB}}(n) = C_{\text{OPT}}(n)$ and we are done. We can now re-write Equation (1) as $C_{\text{MLIB}}(n) \leq \max_{i \in \mathcal{S}} \left\{\max\left\{q_i + 2l_i, r_i + l_i\right\}\right\}$. The latter term can be re-written as $\max_{i \in \mathcal{S}} \left\{r_i + l_i + \max\left\{(l_i - a), 0\right\}\right\} \leq \max_{i \in \mathcal{S}} \left\{r_i + l_i + \max\left\{(l_{max} - a), 0\right\}\right\}$. Now, if $a > l_{max}$, we have that $C_{\text{MLIB}}(n) \leq \max_{i \in \mathcal{S}} \left\{r_i + l_i\right\}$, which implies that $C_{\text{MLIB}}(n) = C_{\text{OPT}}(n)$, and the first part of the lemma is proved. Now, considering the case where $a \leq l_{max}$, we have that

$$C_{\text{MLIB}}(n) \leq \max_{i \in \mathcal{S}} \left\{r_i + l_i + \max\left\{(l_{max} - a), 0\right\}\right\} = \max_{i \in \mathcal{S}} \left\{r_i + l_i\right\} + (l_{max} - a) \leq C_{\text{OPT}}(n) + (l_{max} - a).$$

We re-write $(l_{max} - a)$ as υl_{max}, where $\upsilon = \frac{l_{max} - a}{l_{max}} \leq 1$. Note that $\upsilon l_{max} \leq \frac{\upsilon}{2} C_{\text{OPT}}(n)$. Thus, $C_{\text{MLIB}}(n) \leq C_{\text{OPT}}(n) + (l_{max} - a) = C_{\text{OPT}}(n) + \upsilon l_{max} \leq \left(1 + \frac{\upsilon}{2}\right) C_{\text{OPT}}(n) = \left(\frac{3}{2} - \frac{a}{2l_{max}}\right) C_{\text{OPT}}(n)$, and this completes the proof of the second part of the lemma. ■

Recalling that $\frac{3}{2}$ is the best-possible competitive ratio in the traditional setting, we say that the *value* of the disclosure dates is α. We now show that MLIB is in fact a best-possible algorithm in this situation.

Theorem 4 *Let A be an arbitrary deterministic online algorithm with cost $C_A(n)$ on an instance of n cities. Then $\forall n \geq 2$, there exists an instance of size n where the online cost is at least $(\frac{3}{2} - \alpha) \in [1, \frac{3}{2}]$ times the optimal offline cost, where $\alpha = a/L_{TSP}$.*

Proof Let $n \geq 2$. Generate an instance of $(n-1)$ cities arbitrarily and let $C_A(n-1)$ be the online cost of this algorithm on these $(n-1)$ cities; i.e., algorithm A serves all $(n-1)$ cities and returns to the origin at time $t = C_A(n-1)$. At this time, city n becomes known to algorithm A:

$$(l_n, r_n, q_n) = (a + C_A(n-1), a + C_A(n-1), C_A(n-1)).$$

Note that $l_{max} = l_n = a + C_A(n-1)$ since $C_A(n-1) \geq C_{\mathrm{OPT}}(n-1) \geq 2l_i,\ \forall i < n$. Considering algorithm A, its salesman is at the origin at time q_n. Thus,

$$C_A(n) \geq q_n + 2l_n = 3C_A(n-1) + 2a.$$

Considering the optimal offline algorithm, we have that $C_{\mathrm{OPT}}(n) = \max\{C_{\mathrm{OPT}}(n-1), 2(C_A(n-1)+a)\} = 2(C_A(n-1)+a)$, since $C_A(n-1) \geq C_{\mathrm{OPT}}(n-1)$. Note that $C_{\mathrm{OPT}}(n) > 0$ by Assumption 1. Thus,

$$\begin{aligned}
\frac{C_A(n)}{C_{\mathrm{OPT}}(n)} &\geq 1 + \frac{C_A(n-1)}{2(C_A(n-1)+a)} \\
&= 1 + \frac{C_A(n-1)}{2l_{max}} \\
&= 1 + \frac{l_{max} - a}{2l_{max}} \\
&= \frac{3}{2} - \frac{a}{2l_{max}}.
\end{aligned}$$

Note that by construction, $a \leq l_{max}$ and, consequently, $\frac{3}{2} - \frac{a}{2l_{max}} \in [1, \frac{3}{2}]$. ■

Notice that disclosure dates do not affect MRIN; a single city instance where $r_1 = l_1$ still induces an online cost which is $\frac{3}{2}$ times the optimal offline cost. We thus have the following corollary:

Corollary 1 *Algorithm MLIB is a best-possible online algorithm under the restriction $q_i = (r_i - a)^+,\ \forall i \in \mathcal{N}$. In addition, algorithm MRIN is not best-possible.*

3.3 In-Depth Online Analysis of MLIB Under General Disclosure Dates

In this subsection, we give a general result (of a technical nature) for the MLIB algorithm and we also present an interesting example where advanced information is actually detrimental. We first introduce some definitions:

Definition 1

1. $\delta = \min_{j \in S_\delta(n)} \{\frac{q_j}{l_j}\}$, *where* $S_\delta(n) = \{j \mid q_j + 2l_j = \max_{i \in \mathcal{N}} \{\max\{q_i + 2l_i, r_i + l_i\}\}\}$.

2. $\kappa = \min_{j \in S_\kappa(n)} \{\frac{l_j}{q_j}\}$, *where* $S_\kappa(n) = S_\delta(n) \bigcap \{j \mid q_j > 0\}$.

3. $\gamma = \min_{j \in S_\gamma(n)} \{\frac{q_j}{r_j}\}$, *where* $S_\gamma(n) = S_\delta(n) \bigcap \{j \mid r_j > 0\}$.

Theorem 5

1. *If either or both of the sets* $S_\kappa(n)$ *and* $S_\gamma(n)$ *are empty, then* $C_{MLIB}(n) = C_{OPT}(n)$.

2. *Otherwise,* $C_{MLIB}(n) \leq (1 + \min\{\frac{\gamma}{2}, \frac{\delta}{2}, \frac{\kappa}{1+\kappa}\})C_{OPT}(n)$.

3. *In addition, when well defined,* $(1 + \min\{\frac{\gamma}{2}, \frac{\delta}{2}, \frac{\kappa}{1+\kappa}\}) \leq \frac{3}{2}$.

Proof We first analyze the second part of the theorem, where $S_\kappa(n)$ and $S_\gamma(n)$ are both non-empty. We let m be the index that attains the minimum in the definition of δ; i.e., $q_m = \delta l_m$. By Lemma 1 and Equation (1), we have that

$$\begin{aligned} C_{\text{MLIB}}(n) &\leq q_m + 2l_m \\ &= (\delta + 2)l_m \\ &\leq (1 + \frac{\delta}{2})C_{\text{OPT}}(n). \end{aligned}$$

Let p be the index that attains the minimum in the definition of κ; i.e., $l_p = \kappa q_p$. By Lemma 1 and Equation (1), we have that

$$\begin{aligned} C_{\text{MLIB}}(n) &\leq q_p + 2l_p \\ &= q_p + l_p + l_p\left(\frac{1+\kappa}{1+\kappa}\right) \\ &= q_p + l_p + \frac{\kappa q_p + \kappa l_p}{1+\kappa} \\ &= (1 + \frac{\kappa}{1+\kappa})(q_p + l_p) \\ &\leq (1 + \frac{\kappa}{1+\kappa})C_{\text{OPT}}(n). \end{aligned}$$

Finally, we let k be the index that attains the minimum in the definition of γ; i.e., $q_k = \gamma r_k$. By Lemma 1, we have that

$$\begin{aligned} C_{\text{MLIB}}(n) &\leq q_k + 2l_k \\ &= \gamma r_k + 2l_k. \end{aligned}$$

We consider three possibilities:

(1) If $l_k > r_k$, we have that $2l_k + \gamma r_k < (2 + \gamma)l_k \leq (1 + \frac{\gamma}{2})C_{\text{OPT}}(n)$.

(2) If $l_k < (1-\gamma)r_k$, $2l_k + \gamma r_k < l_k + r_k \leq C_{\text{OPT}}(n)$.

(3) If $(1-\gamma)r_k \leq l_k \leq r_k$, we can let $l_k = (1-\hat{\gamma})r_k$ for some $\hat{\gamma} \in [0, \gamma]$. After some simple algebra, we see that

$$2l_k + \gamma r_k = (r_k + l_k) + \frac{(\gamma - \hat{\gamma})}{(1-\hat{\gamma})} l_k \leq C_{\text{OPT}}(n) + \gamma l_k \leq (1 + \frac{\gamma}{2}) C_{\text{OPT}}(n),$$

where the first inequality holds because the function $f_\gamma(\hat{\gamma}) = \frac{(\gamma-\hat{\gamma})}{(1-\hat{\gamma})}$ attains a maximum of γ (when $\hat{\gamma} = 0$) on the domain $[0, \gamma]$, since $\gamma \leq 1$. Thus, $C_{\text{MLIB}}(n) \leq (1 + \frac{\gamma}{2}) C_{\text{OPT}}(n)$. As the previous analyses were mutually exclusive, we may conclude that, if $S_\kappa(n)$ and $S_\gamma(n)$ are both not empty, $C_{\text{MLIB}}(n) \leq (1 + \min\{\frac{\gamma}{2}, \frac{\delta}{2}, \frac{\kappa}{1+\kappa}\}) C_{\text{OPT}}(n)$.

We now analyze the first part of the theorem. We have that either or both $S_\kappa(n)$ and $S_\gamma(n)$ are empty. We first consider the case where the superset $S_\delta(n) = \emptyset$. In this situation, there exists a city j s.t. $r_j + l_j = \max_{i \in \mathcal{N}} \{\max\{q_i + 2l_i, r_i + l_i\}\}$. By Lemma 1 and Equation (1) we have that

$$\begin{aligned} C_{\text{MLIB}}(n) &\leq r_j + l_j \\ &\leq C_{\text{OPT}}(n). \end{aligned}$$

Recalling that $C_{\text{MLIB}}(n) \geq C_{\text{OPT}}(n)$, we conclude that $C_{\text{MLIB}}(n) = C_{\text{OPT}}(n)$. Now, assume $S_\delta(n)$ contains at least one element. If $S_\kappa(n)$ is empty, then $\delta = 0$. The analysis that results in Equation (2) proves that $C_{\text{MLIB}}(n) \leq C_{\text{OPT}}(n)$. Thus, $C_{\text{MLIB}}(n) = C_{\text{OPT}}(n)$. Now, if $S_\gamma(n)$ is empty, $r_j = 0$, $\forall j \in S_\delta(n)$. This again implies that $\delta = 0$ and, consequently, $C_{\text{MLIB}}(n) = C_{\text{OPT}}(n)$.

We conclude by analyzing the third part of the theorem. Since $\min\{\delta, \kappa\} \leq 1$ (also $\gamma \leq 1$), $(1 + \min\{\frac{\gamma}{2}, \frac{\delta}{2}, \frac{\kappa}{1+\kappa}\}) \leq \frac{3}{2}$. ■

Since the best-possible online algorithm, with no disclosure dates, has a competitive ratio of $\frac{3}{2}$, we say that the *value* of the disclosure dates is

$$\begin{cases} \frac{1}{2} - \min\{\frac{\gamma}{2}, \frac{\delta}{2}, \frac{\kappa}{1+\kappa}\}), & \text{if } \kappa \text{ and } \gamma \text{ are well defined} \\ \frac{1}{2}, & o.w. \end{cases}$$

To conclude our analysis of the online TSP on the non-negative real line, we provide an example where the advanced information of the disclosure dates is actually detrimental.

Example: Consider the two city instance where $q_1 = 0$, $r_1 = l_1 = 1$, $q_2 = r_2 = 2$ and $l_2 = 1$. This instance induces the following costs: $C_{\text{MRIN}}(2) = 3$ and $C_{\text{MLIB}}(2) = 4$.

However, we have conducted computational experiments that confirm the intuitively clear superiority of MLIB over MRIN on average.

4 The Online TSP on General Metric Spaces

We now consider the general case where cities belong to a generic metric space $\mathcal{M}$. Let $d(\cdot, \cdot)$ be the metric for the space and o the origin. We consider the value of advanced information, for the structure $q_i = (r_i - a)^+, \ \forall i \in \mathcal{N}$, providing lower and upper bounds on the competitive ratio. The proof of our first result consists of simple modifications of the proof of Theorem 3.1 in [19].

Theorem 6 *Any ρ-competitive algorithm for the online TSP on a metric space $\mathcal{M}$, with $q_i = (r_i - a)^+, \ i \in \mathcal{N}$, has $\rho \geq 2/(1+\alpha)$, where $\alpha = a/L_{TSP}$.*

Proof Define a metric space $\mathcal{M}$ as a graph with vertex set $V = \{1, 2, \ldots, n\} \cup \{o\}$ with distance function d that satisfies the following: $d(o,i) = 1$ and $d(i,j) = 2$ for all $i \neq j \in V \setminus \{o\}$.

At time 0, there is a request at each of the n cities in $V \setminus \{o\}$. If an online server visits the request at city i at time $t \leq 2n - 1 - \epsilon$, for some small ϵ, then at time $t + \epsilon$, a new request is disclosed at city i.

In this way, at time $2n-1$ the online server still has to serve requests at all n cities; furthermore, at time $2n-1$, all cities have only been disclosed, not necessarily released. Therefore, the online cost is at least the corresponding value in the situation where all cities have been released by time $2n-1$. This latter value is at least $4n-2$. Therefore, denoting C_A as the online cost of an arbitrary online algorithm A, we have that $C_A \geq 4n-2$.

The optimal offline server will also have some difficulty with the differences between the disclosure dates and release dates. We first note that, had the cities been released at the above mentioned times, rather than disclosed, the optimal offline cost would have been $2n$. We now exploit the structure of the disclosure date/release date relationship: by waiting a units of time at any disclosed city, the city's release date will arrive. Therefore, it is clear that $C_{\text{OPT}} \leq 2n + a$. Finally, by noting that $L_{TSP} = 2n$, we have that

$$\frac{C_A}{C_{\text{OPT}}} \geq \frac{4n-2}{2n+a} = \frac{2}{1+\alpha} - \frac{2}{2n+a}.$$

Taking n arbitrarily large proves the theorem. ■

Now, we give the first of two generalizations of the 2-competitive online algorithm PAH, originally proposed by Ausiello et al. [3]. We call our algorithm Plan-At-Home-disclosure-dates (PAH-dd).

Algorithm PAH-dd

(1) Whenever the salesman is at the origin, it starts to follow a tour that serves all cities whose disclosure dates have passed but have not yet been served; this tour is constructed using an algorithm A that *exactly* solves an offline TSP with release dates.

(2) If at time q_i, for some i, a new city is presented at point x, the salesman takes one of two actions depending on the salesman's current position p:

- (2a) If $d(x,o) > d(p,o)$, the salesman goes back to the origin (following the shortest path from p) where it appears in a Case (1) situation.
- (2b) If $d(x,o) \leq d(p,o)$, the salesman ignores the city until it arrives at the origin, where again it re-enters Case (1).

Theorem 7 *Algorithm PAH-dd is* $(2 - \dfrac{\alpha}{1+\alpha})$*-competitive, where* $\alpha = \frac{a}{L_{TSP}}$.

Proof Let $p(t)$ be the position of the salesman at time t. As in Ausiello et al. [3], we provide a case by case analysis. Let us consider the state of the algorithm at time q_n, the final disclosure date.

Case (1): The salesman is at the origin at time q_n. Let $\mathcal{T}$ be the tour, calculated by algorithm A at time q_n, that visits all unserved cities; for simplicity, we let $\mathcal{T}$ also denote the duration of the tour. Letting $C_{\text{PAH-dd}}$ denote the online cost of our new algorithm, we have that

$$\begin{aligned} C_{\text{PAH-dd}} &= q_n + \mathcal{T} \\ &= r_n + \mathcal{T} - a \\ &\leq C_{\text{OPT}} + (\mathcal{T} - a) \\ &= C_{\text{OPT}} + (1 - \frac{a}{\mathcal{T}})\mathcal{T} \\ &\leq C_{\text{OPT}} + (1 - \frac{a}{\mathcal{T}})C_{\text{OPT}}, \end{aligned}$$

where the last inequality is by $\mathcal{T} \leq C_{\text{OPT}}$. Inserting the obvious bound $\mathcal{T} \leq a + L_{TSP}$ proves the theorem for this case.

Case (2a): We have that $d(o, l_n) > d(o, p(q_n))$ and the salesman returns to the origin, arriving before time $q_n + d(o, l_n) = r_n + d(o, l_n) - a$. Once at the origin, the salesman uses algorithm A to compute a tour $\mathcal{T}'$. Clearly, $r_n + d(o, l_n) \leq C_{\text{OPT}}$. Thus, we have that

$$C_{\text{PAH-dd}} \leq r_n + d(o, l_n) + (\mathcal{T}' - a) \leq C_{\text{OPT}} + (1 - \frac{\alpha}{1+\alpha})C_{\text{OPT}} = (2 - \frac{\alpha}{1+\alpha})C_{\text{OPT}}.$$

Case (2b): We have that $d(o, l_n) \leq d(o, p(q_n))$. Suppose that the salesman is following a route $\mathcal{R}$ that had been computed the last time he was at the origin. Clearly, $\mathcal{R} \leq C_{\text{OPT}}$. Let $\mathcal{Q}$ be the set of cities temporarily ignored since the last time the salesman was at the origin. Let j be the index of the first city in $\mathcal{Q}$ that is visited by the optimal offline algorithm. Let $\mathcal{P}_{\mathcal{Q}}$ be the shortest path starting from location l_j at time r_j, visiting all other cities in $\mathcal{Q}$, while respecting the release dates, and terminating at the origin. Clearly, $r_j + \mathcal{P}_{\mathcal{Q}} \leq C_{\text{OPT}}$.

Since city j was ignored when it was disclosed, we have that $d(o, l_j) \leq d(o, p(q_j))$. Thus, at time q_j the salesman had already traveled at least a distance $d(o, l_j)$ on $\mathcal{R}$ and will complete $\mathcal{R}$ at the latest at time $t_{\mathcal{R}} = q_j + \mathcal{R} - d(o, l_j)$. Next, the salesman will compute $\mathcal{T}_{\mathcal{Q}}$, a tour covering $\mathcal{Q}$.

At time $t_{\mathcal{R}}$, consider an alternate strategy that first goes to city j, possibly waits for city j to be released, and then follows the shortest path through the cities in $\mathcal{Q}$; this latter path is at most $\mathcal{P}_{\mathcal{Q}}$. Clearly, $\mathcal{T}_{\mathcal{Q}}$ will finish before this alternate strategy finishes. Next, notice that the completion time of $\mathcal{T}_{\mathcal{Q}}$ is also the completion time of PAH-dd; therefore, we have that

$$
\begin{aligned}
C_{\text{PAH-dd}} &\leq \max\{t_{\mathcal{R}} + d(o, l_j), r_j\} + \mathcal{P}_{\mathcal{Q}} \\
&= \max\{t_{\mathcal{R}} + d(o, l_j) + \mathcal{P}_{\mathcal{Q}}, r_j + \mathcal{P}_{\mathcal{Q}}\} \\
&\leq \max\{t_{\mathcal{R}} + d(o, l_j) + \mathcal{P}_{\mathcal{Q}}, C_{\text{OPT}}\} \\
&= \max\{q_j + \mathcal{R} + \mathcal{P}_{\mathcal{Q}}, C_{\text{OPT}}\} \\
&= \max\{(r_j + \mathcal{P}_{\mathcal{Q}}) + (\mathcal{R} - a), C_{\text{OPT}}\} \\
&\leq \max\left\{C_{\text{OPT}} + \left(1 - \frac{\alpha}{1+\alpha}\right) C_{\text{OPT}}, C_{\text{OPT}}\right) \\
&= \left(2 - \frac{\alpha}{1+\alpha}\right) C_{\text{OPT}}. \blacksquare
\end{aligned}
$$

Since the best-possible algorithm for the online metric TSP has a competitive ratio of 2, Theorems 6 and 7 indicate that the *value* of the disclosure dates is at least $\frac{\alpha}{1+\alpha}$ and no more than $\frac{2\alpha}{1+\alpha}$.

5 The Online TRP on General Metric Spaces

Thus far, we have been analyzing versions of the online TSP, where the objective is arguably in the salesman's interest. We now consider another objective, the weighted latency, which is an objective that is arguably in the cities' interest; additionally, the weights may be chosen to favor certain cities over others.

In this section, we consider the online TRP with arbitrary weights. Our objective is to minimize $\sum_{i \in \mathcal{N}} w_i C_i$, where C_i is the completion time of city i, the first time it is visited after its release date, and the w_i are arbitrary non-negative weights. Again, $l_i \in \mathcal{M}$, for any metric space $\mathcal{M}$ and we consider the situation where $q_i = (r_i - a)^+, \ \forall i \in \mathcal{N}$.

5.1 A Deterministic Online TRP Algorithm for General $\mathcal{M}$

Let $\lambda = (1 + \sqrt{2})$, $b_0 = \min\{r_j \mid r_j \geq \frac{a}{\lambda}\}$ and $b_i = \lambda^i b_0$. Also, let $\tilde{b}_i = b_i - a$. The definition of b_0 ensures that $\tilde{b}_1 \geq 0$, which is necessary for step 1 of BREAK to be feasible. The latter $\tilde{b}_i$ parameters are the breakpoints where the online algorithm BREAK (to be defined shortly) will generate some re-optimization. Our algorithm is a generalization of the $(1 + \sqrt{2})^2$-competitive INTERVAL$_\alpha$ by Krumke et al. [16] (α in [16] is equivalent to λ in this paper), which re-optimizes at times b_i. Let $Q_i, \ i \geq 1$ denote the set of cities released up to and including time b_i; clearly $Q_i \subseteq Q_{i+1}, \ \forall i$. Note that at time $\tilde{b}_i$ the online repairman knows Q_i. Let R_i denote the set of cities served by algorithm BREAK in the interval $[\tilde{b}_i, \tilde{b}_{i+1}]$ and R_i^* the set of cities served by the optimal offline algorithm in the interval $[b_{i-1}, b_i]$. Finally, let $w(S) = \sum_{i \in S} w_i$.

Definition 2 *Online algorithm BREAK:*[1]

[1] Note that BREAK is not a polynomial-time algorithm since step 2 requires the exact solution of the NP-hard Orienteering Problem [6].

1. *Remain idle at the origin until time* $\tilde{b}_1$.

2. *At time* $\tilde{b}_1$ *calculate a path of length at most* b_1 *to serve a set of cities* $R_1 \subseteq Q_1$ *such that* $w(R_1)$ *is maximized.*

3. *At time* $\tilde{b}_i$, $i \geq 2$, *return to the origin and then calculate a path of length at most* b_i *to serve a set of cities* $R_i \subseteq Q_i \setminus \bigcup_{j<i} R_j$ *such that* $w(R_i)$ *is maximized.*

This algorithm is easily seen to be feasible – actions in iteration i are completed before actions in iteration $(i+1)$ are to begin. We begin our analysis of algorithm BREAK with the following lemma, which generalizes a result in [16]. Our proof of this lemma is quite different from that of [16] and follows the proof of a similar result in the machine scheduling literature (see [11]).

Lemma 2 *For any* $k \geq 1$, $\sum_{i=1}^{k} w(R_i) \geq \sum_{i=1}^{k} w(R_i^*)$.

Proof Consider iteration $k \geq 2$ and let $R = \bigcup_{l=1}^{k} R_l^* \setminus \bigcup_{l=1}^{k-1} R_l$. If a repairman were at the origin at time zero, he could serve all the cities in the set R by time b_k.

Now, consider an online repairman at time $\tilde{b}_k$. Suppose he knew the set R. Then by returning to the origin, taking at most b_{k-1} time units, the repairman could serve the cities in R by time $\tilde{b}_k + b_{k-1} + b_k = \tilde{b}_{k+1}$ (equality since $\alpha = (1+\sqrt{2})$). Thus, in iteration k, the repairman could serve cities of total weight $w(R)$.

Unfortunately, the repairman does not know R since the R_i^* are not known until all cities are released. However, the repairman's task is to find a subset of $S = Q_k \setminus \bigcup_{l=1}^{k-1} R_l$. Since $Q_k \supseteq \bigcup_{l=1}^{k} R_l^*$, $S \supseteq R$, and the online repairman is able to choose a subset of S to serve in iteration k of total weight at least $w(R)$, since choosing R as the subset is a feasible choice. A similar argument holds for $k = 1$.

Now, for any k,

$$
\begin{aligned}
w(R_k) &\geq w(R) \\
&= \sum_{j \in \bigcup_{l=1}^{k} R_l^* \setminus \bigcup_{l=1}^{k-1} R_l} w_j \\
&= \sum_{l=1}^{k} w(R_l^*) - \sum_{j \in \left(\bigcup_{l=1}^{k} R_l^*\right) \cap \left(\bigcup_{l=1}^{k-1} R_l\right)} w_j \\
&\geq \sum_{l=1}^{k} w(R_l^*) - \sum_{j \in \bigcup_{l=1}^{k-1} R_l} w_j \\
&= \sum_{l=1}^{k} w(R_l^*) - \sum_{l=1}^{k-1} w(R_l),
\end{aligned}
$$

which gives the result. ■

The following corollary is evident from Lemma 2.

Corollary 2 *Suppose the optimal offline algorithm visits the last city in its tour in interval* $(b_{p-1}, b_p]$ *for some* $p \geq 1$*. Then the online algorithm BREAK will visit its last city by time* $\tilde{b}_{p+1}$*.*

We now give the main theorem of this section.

Theorem 8 *Algorithm BREAK is* $((1+\sqrt{2})^2 - \frac{\alpha\beta}{\alpha+\beta})$*-competitive, where* $\alpha = \frac{a}{L_{TSP}}$ *and* $\beta = \frac{a}{r_{max}}$*.*

Proof of Theorem 8 We begin by stating Lemma 6 from [16]: Let $a_i, b_i \in \mathbb{R}_+$, for $i = 1, \ldots, p$. If $\sum_{i=1}^{p} a_i = \sum_{i=1}^{p} b_i$ and $\sum_{i=1}^{p'} a_i \geq \sum_{i=1}^{p'} b_i$ for all $1 \leq p' \leq p$, then $\sum_{i=1}^{p} \tau_i a_i \leq \sum_{i=1}^{p} \tau_i b_i$ for any non-decreasing sequence $0 \leq \tau_1 \leq \ldots \leq \tau_p$. Applying this lemma, we have that

$$\begin{aligned} C_{\text{BREAK}} &\leq \sum_{k=1}^{p} \tilde{b}_{k+1} w(R_k) \\ &\leq \sum_{k=1}^{p} \tilde{b}_{k+1} w(R_k^*) \\ &= \sum_{k=1}^{p} (b_{k+1} - a) w(R_k^*) \\ &= \sum_{k=1}^{p} (\lambda^2 b_{k-1} - a) w(R_k^*) \\ &= \sum_{k=1}^{p} \sum_{l \in R_k^*} (\lambda^2 b_{k-1} - a) w_l \\ &\leq \sum_{k=1}^{p} \sum_{l \in R_k^*} (\lambda^2 C_l^* - a) w_l, \end{aligned}$$

where C_l^* the the completion time of city l by the optimal offline algorithm. Now, suppose there exists γ such that $(\lambda^2 C_l^* - a) \leq \gamma C_l^*$, $\forall l$. Then, algorithm BREAK would be γ-competitive. It is clear to see that $\gamma = \lambda^2 - \frac{a}{C_{max}^*}$ is the smallest such value to satisfy the requirements, where $C_{max}^* = \max_{i \in \mathcal{N}} \{C_i^*\}$. Thus, algorithm BREAK is $(\lambda^2 - \frac{a}{C_{max}^*})$-competitive. Finally, using the fact that $C_{max}^* \leq r_{max} + L_{TSP}$, we achieve the result. ■

Since the best deterministic algorithm to date (INTERVAL_α) for the online metric TRP is $(1+\sqrt{2})^2$-competitive, we say that the value of the disclosure dates is $\frac{\alpha\beta}{\alpha+\beta}$.

5.2 A Randomized Online TRP Algorithm for General $\mathcal{M}$

We may also define a randomized algorithm BREAK-R as algorithm BREAK with the following substitution: $b_0 \mapsto \lambda^U b_0$, where U is a uniform random variable on $[0, 1]$. We have the following theorem for this randomized algorithm; its proof is quite similar to that of Theorem 8 and is omitted.

Theorem 9 *Algorithm BREAK-R is* $(\Theta - \frac{\alpha\beta}{\alpha+\beta})$*-competitive, where* $\alpha = \frac{a}{L_{TSP}}$*,* $\beta = \frac{a}{r_{max}}$ *and* $\Theta \approx 3.86$*.*

Remark 1 *To the best of our knowledge, algorithms INTERVAL$_\alpha$ and RANDINTERVAL$_\alpha$ ([16]) are the best online algorithms to-date for the online TRP, regardless of the metric space; i.e., we are not aware of any algorithms that improve these results for any simpler metric spaces, such as $\mathbb{R}_+$ or $\mathbb{R}$. Therefore, we do not have any new results specific to these particular metric spaces.*

5.3 A Correction to a Previously Published Result

When $a = 0$, algorithm BREAK-R corresponds to a realization of the α-parameterized (this α unrelated to the $\alpha = \frac{a}{L_{TSP}}$ utilized in this paper) online algorithm RANDINTERVAL$_\alpha$, given in [16]. The values of α for which RANDINTERVAL$_\alpha$ is a feasible algorithm were given incorrectly in [16]: The given range $\alpha \in [1+\sqrt{2}, 3]$ should have read $\alpha \in (1, 1+\sqrt{2}]$ since the algorithm requires $\frac{(\alpha+1)}{\alpha(\alpha-1)} \geq 1$. This led to an erroneous result that stated that RANDINTERVAL$_\alpha$ was $\tilde{\Theta}$-competitive, $\tilde{\Theta} \approx 3.64$. Using the correct range for α, it is straightforward to see that RANDINTERVAL$_\alpha$ is Θ-competitive, where $\Theta \approx 3.86$.

5.4 A Final Note on the Online Dial-A-Ride Problem

Finally, note that Theorems 8 and 9 also hold for the online Dial-A-Ride problem, which is a generalization of the TRP. Instead of a customer (city) requesting a visit, a customer requests a ride from a source location to a destination location. The completion time of a customer is the time that the customer reaches the destination. The subroutine in the BREAK algorithm that calculates paths maximizing the weight of served customers must simply be modified to incorporate the new requirements of a customer.

6 Polynomial-time Online Algorithms for the Online TSP

We now give our second generalization of algorithm PAH ([3]), which we shall denote as PAH-p as all subroutines are polynomial-time. In this section, we only consider the traditional case where $q_i = r_i,\ \forall i \in \mathcal{N}$.

Algorithm PAH-p

(1) Whenever the salesman is at the origin, it starts to follow a tour that serves all cities whose release dates have passed but have not yet been served; this tour is constructed using an ρ-approximation algorithm A that solves an offline TSP.

(2) If at time r_i, for some i, a new city is presented at point x, the salesman takes one of two actions depending on the salesman's current position p:

(2a) If $d(x, o) > d(p, o)$, the salesman goes back to the origin (following the shortest path from p) where it appears in a Case (1) situation.

(2b) If $d(x, o) \leq d(p, o)$, the salesman ignores the city until it arrives at the origin, where again it re-enters Case (1).

Theorem 10 *Algorithm PAH-p is 2ρ-competitive.*

Proof Let r_n be the time of the last request, l_n the position of this request and $p(t)$ the location of the salesman at time t. We show that in each of the Cases (1), (2a) and (2b), PAH-p is 2ρ-competitive.

Case (1): PAH-p is at the origin at time r_n. Then it starts a ρ-approximate tour that serves all the unserved requests. The time needed by PAH-p is at most $r_n + \rho L_{TSP} \leq (1+\rho)C_{\text{OPT}} \leq 2\rho C_{\text{OPT}}$.

Case (2a): We have that $d(o, l_n) > d(o, p(r_n))$. PAH-p returns to the origin, where it will arrive before time $r_n + d(o, l_n)$. After this, PAH-p computes and follows a ρ-approximate tour through all the unserved requests. Therefore, the online cost is at most $r_n + d(o, p(r_n)) + \rho L_{TSP} < r_n + d(o, l_n) + \rho L_{TSP}$. Noticing that $r_n + d(o, l_n) \leq C_{\text{OPT}}$, we have that the online cost is at most $(1+\rho)C_{\text{OPT}} \leq 2\rho C_{\text{OPT}}$.

Case (2b): We have that $d(o, l_n) \leq d(o, p(r_n))$. Suppose PAH-p is following a route $\mathcal{R}$ that had been computed the last time it was at the origin. Note that $\mathcal{R} \leq \rho L_{TSP} \leq \rho C_{\text{OPT}}$. Let $\mathcal{Q}$ be the set of requests temporarily ignored since the last time PAH-p was at the origin. Let l_q be the location of the first request in $\mathcal{Q}$ served by the offline algorithm and let r_q be the time at which l_q was released. Let $\mathcal{P}_{\mathcal{Q}}$ be the shortest path that starts at l_q, visits all the cities in $\mathcal{Q}$ and ends at o. Clearly, $C_{\text{OPT}} \geq r_q + \mathcal{P}_{\mathcal{Q}}$ and $C_{\text{OPT}} \geq d(o, l_q) + \mathcal{P}_{\mathcal{Q}}$.

At time r_q, the distance that PAH-p still has to travel on the route $\mathcal{R}$ before arriving at o is at most $\mathcal{R} - d(o, l_q)$, since $d(o, p(r_q)) \geq d(o, l_q)$ implies that PAH-p has traveled on the route $\mathcal{R}$ a distance not less than $d(o, l_q)$. Therefore, it will arrive at o before time $r_q + \mathcal{R} - d(o, l_q)$. After that it will follow a ρ-approximate tour $\mathcal{T}_{\mathcal{Q}}$ that covers the set $\mathcal{Q}$; letting $\mathcal{T}^*_{\mathcal{Q}}$ be the optimal tour over the set $\mathcal{Q}$, we have that $\mathcal{T}_{\mathcal{Q}} \leq \rho \mathcal{T}^*_{\mathcal{Q}}$. Hence, the completion time will be at most $r_q + \mathcal{R} - d(o, l_q) + \rho \mathcal{T}^*_{\mathcal{Q}}$. Since $\mathcal{T}^*_{\mathcal{Q}} \leq d(0, l_q) + \mathcal{P}_{\mathcal{Q}}$, we have that the online cost is at most

$$\begin{aligned} r_q + \mathcal{R} - d(o, l_q) + \rho d(0, l_q) + \rho \mathcal{P}_{\mathcal{Q}} &= (r_q + \mathcal{P}_{\mathcal{Q}}) + \mathcal{R} + (\rho - 1)(d(0, l_q) + \mathcal{P}_{\mathcal{Q}}) \\ &\leq C_{\text{OPT}} + \rho C_{\text{OPT}} + (\rho - 1)C_{\text{OPT}} \\ &= 2\rho C_{\text{OPT}}. \ \blacksquare \end{aligned}$$

Applying well-known algorithms by Chistofides [8] and Arora [1], we are able to attain two interesting corollaries.

Corollary 3 *If we choose A as Christofides' heuristic, Algorithm PAH-p is 3-competitive.*

This matches the 3-competitive polynomial-time algorithm given in [3]. However, a polynomial-time algorithm with a competitive ratio of at most 2.78 was recently given in [2].

Corollary 4 *If $\mathcal{M}$ is Euclidean and we choose A as Arora's PTAS, for any $\epsilon > 0$, Algorithm PAH-p is $(2+\epsilon)$-competitive.*

To the best of our knowledge, this is the first result for the online TSP in the Euclidean metric space.

Remark 2 *We have attempted to combine our analyses, to find a single result that unifies a polynomial-time algorithm and the value of information. Our approach would have improved upon the above results if we had a ρ-approximation algorithm for the TSP with release dates, where $\rho < \frac{5}{2}$. Trivially (wait until the last release date and then form a Christofides approximate tour) we have a $\frac{5}{2}$-approximation algorithm; unfortunately, we know of no algorithm that has a better approximation ratio.*

7 Conclusion

We have considered online versions of two well-studied routing optimization problems, the Traveling Salesman Problem and the Traveling Repairman Problem. These two problems embody two major types of objectives: optimizing in the server's interest and optimizing in the customers' interest. We introduced the notion of a disclosure date, which brings with it a number of benefits. First, we are allowed to relax the pessimistic definition of the competitive ratio. Second, this relaxation is natural, in the sense that realistic problems can be modeled with disclosure dates. Third, the disclosure dates allow us to vary the online-ness of a problem.

With these disclosure dates in place, we show their value in the form of improved competitiveness results for both the online TSP and TRP, in a variety of metric spaces. For the non-negative real line, we show that our algorithm is strictly optimal.

Finally, we consider polynomial-time online algorithms for the traditional (no disclosure dates) online TSP on metric spaces and we give the first competitiveness result for Euclidean metric spaces.

8 Acknowledgements

We thank the anonymous referees for very thoughtful comments that significantly improved this paper.

References

[1] S. Arora. Polynomial-time approximation schemes for euclidean tsp and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998.

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

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

[4] D.J. Bertsimas. *Probabilistic Combinatorial Optimization Problems*. PhD thesis, Massachusetts Institute of Technology, 1988.

[5] M. Blom, S.O. Krumke, W.E. de Paepe, and L. Stougie. The online tsp against fair adversaries. *INFORMS Journal on Computing*, 13(2):138–148, 2001.

[6] A. Blum, S. Chawla, D. Karger, T. Lane, A. Meyerson, and M. Minkoff. Approximation algorithms for orienteering and discounted-reward tsp. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 46–55, 2003.

[7] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, first edition, 1998.

[8] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. Management Sciences Research Report 388, Carnegie-Mellon University, February 1976.

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

[10] A. Fiat and G. Woeginger. *Online Algorithms: The State of the Art*. Springer Verlag LNCS State of the Art Survey, 1998.

[11] L.A. Hall, A. Schulz, D.B. Shmoys, and J. Wein. Scheduling to minimize average completion time: off-line and on-line approximation algorithms. *Mathematics of Operations Research*, 22(3):513–544, 1997.

[12] P. Jaillet. *Probabilistic Traveling Salesman Problems*. PhD thesis, Massachusetts Institute of Technology, 1985.

[13] B. Kalyanasundaram and K.R. Pruhs. Constructing competitive tours from local information. *Theoretical Computer Science*, 130(1):125–138, 1994.

[14] A. Karlin, M. Manasse, L. Rudolph, and D. Sleator. Competitive snoopy caching. *Algorithmica*, 3:79–119, 1988.

[15] B. Korte and J. Vygen. *Combinatorial Optimization, Theory and Algorithms*. Springer, second edition, 2002.

[16] S.O. Krumke, W.E. de Paepe, D. Poensgen, and L. Stougie. News from the online traveling repairman. *Theoretical Computer Science*, 295:279–294, 2003.

[17] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys. *The Traveling Salesman Problem, A Guided Tour of Combinatorial Optimization*. John Wiley & Sons Ltd., 1985.

[18] M. Lipmann. The online traveling salesman problem on the line. Master's thesis, University of Amsterdam, 1999.

[19] M. Lipmann. *On-line Routing*. PhD thesis, Technische Universiteit Eindhoven, 2003.

[20] H. Psaraftis, M. Solomon, T. Magnanti, and T. Kim. Routing and scheduling on a shoreline with release times. *Management Science*, 36(2):212–223, 1990.

[21] D. Sleator and R. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.

[22] J.N. Tsitsiklis. Special cases of traveling salesman and repairman problems with time windows. *Networks*, 22(3):263–282, 1992.