A Note on: "News from the Online Traveling Repairman" *

Patrick Jaillet † and Michael Wagner ‡

July 22, 2004

Abstract

This note identifies and corrects some errors in the recent paper "News from the online traveling repairman," S. O. Krumke, W. E. de Paepe, D. Poensgen, L. Stougie, *Theoretical Computer Science*, 295, pp. 279-294, 2003. We also provide additional results that corroborate our corrections. Finally, we consider a new online algorithm that is motivated by an algorithm in the above paper and we prove a technical result for this new algorithm.

1 Introduction

In this note, we identify and correct some mistakes in the paper "News from the online traveling repairman," S. O. Krumke, W. E. de Paepe, D. Poensgen, L. Stougie, *Theoretical Computer Science*, 295, pp. 279-294, 2003. We refer the reader to the original paper for details about the terminology which we use here without additional explanation.

We first identify a feasibility error in the definition of algorithm INTERVAL$_\alpha$ (and consequently RANDINTERVAL$_\alpha$). This error led to erroneous statements of certain theorems as well as an incorrect result associated with the upper bound on the competitive ratio of RANDINTERVAL$_\alpha$. We provide corrections. We also provide a general framework for minimizing the upper bounds on the competitive ratios. Finally, we define a new online algorithm that is motivated by algorithm INTERVAL$_\alpha$; we then provide a proof of a critical lemma from Krumke et al. (2003) for this new algorithm (which also serves as an alternate proof of the original result).

*S. O. Krumke, W. E. de Paepe, D. Poensgen, L. Stougie, "News from the online traveling repairman," *Theoretical Computer Science*, 295, pp. 279-294, 2003.

†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.

2 A feasibility error and corrected theorems

The range of parameters for which algorithm INTERVAL$_\alpha$ is feasible is not correct. In Krumke et al. (2003) the β parameter is defined as

$$\beta = \frac{(\alpha+1)}{\alpha(\alpha-1)}$$

for the range $\alpha \in [1+\sqrt{2}, 3]$. The design of the algorithm requires $\beta \geq 1$, but this in turn requires that $\alpha \leq 1+\sqrt{2}$. As a result, Theorem 7 is not entirely correct as stated. Instead it should read:

Theorem 7 Algorithm INTERVAL$_\alpha$ is $\alpha(\alpha+1)/(\alpha-1)$-competitive for the L-OLDARP for any $\underline{\alpha \in (1, 1+\sqrt{2}]}$. For $\alpha = 1+\sqrt{2}$, this yields a competitive ratio of $(1+\sqrt{2})^2 < 5.8285$.

Results related to algorithm RANDINTERVAL$_\alpha$ must also be restated:

Theorem 11 Algorithm RANDINTERVAL$_\alpha$ is $(\alpha{+}1)/(\ln\alpha)$-competitive for the L-OLDARP against an oblivious adversary, where $\underline{\alpha \in (1, 1+\sqrt{2}]}$. Choosing $\underline{\alpha = 1+\sqrt{2}}$ yields a competitive ratio of $\underline{(2+\sqrt{2})/(\ln(1+\sqrt{2})) < 3.8738}$ against an oblivious adversary.

Corollary 12 For $\underline{\alpha = 1+\sqrt{2}}$, algorithm RANDINTERVAL$_\alpha$ is $\underline{(2+\sqrt{2})/(\ln(1+\sqrt{2}))}$-competitive for the OLTRP.

Note that the new upper bounds on the competitive ratio are larger than those in Krumke et al. (2003). The original bounds of $4/\ln(3) < 3.6410$, calculated using $\alpha = 3$, are not necessarily true.

2.0.1 Optimizing α and β: A general approach using nonlinear programming

It is possible to define two associated nonlinear programs, in α and β, that can be used to mimimize the upper bounds on the appropriate competitive ratios.

For the deterministic case, we relax all definitions of α and β, other than $\alpha > 1$ and $\beta \geq 1$. We notice that for the algorithms to be well-formed, schedule i must finish before schedule $(i+1)$ is slated to start. We distinguish between $i = 1$ and $i > 1$. For $i = 1$, we require that $B_1 + B_1 = 2B_1 \leq \beta B_2 = \beta\alpha B_1$, which can be rewritten as $\frac{2}{\alpha} \leq \beta$. For $i > 1$, we require $\beta B_i + B_i + B_{i-1} \leq \beta B_{i+1}$, which can be rewritten as $\frac{\alpha+1}{\alpha(\alpha-1)} \leq \beta$. Define $\mathcal{X}$ as

$$\mathcal{X} = \{(\alpha, \beta) \mid \frac{2}{\alpha} \leq \beta,\ \frac{\alpha+1}{\alpha(\alpha-1)} \leq \beta,\ \alpha > 1,\ \beta \geq 1\}.$$

For the deterministic algorithm INTERVAL$_\alpha$, it can be seen that the solution to the following nonlinear program minimizes the upper bound on the competitive ratio:

$$\begin{aligned} &\min \alpha^2\beta \\ &\text{s.t. } (\alpha, \beta) \in \mathcal{X}. \end{aligned}$$

Likewise, for the randomized algorithm RANDINTERVAL$_\alpha$, it can be seen that the solution to the following nonlinear program minimizes the upper bound on the competitive ratio:

$$\begin{array}{l} \min \frac{\alpha\beta(\alpha-1)}{\ln \alpha} \\ \text{s.t. } (\alpha, \beta) \in \mathcal{X}. \end{array}$$

The solution to both these nonlinear programs is $(\alpha^*, \beta^*) = (1+\sqrt{2}, 1)$, which agrees with the corrected statements of Theorems 7 and 11.

3 A new online algorithm and a technical result

In this section we consider a new online algorithm, motivated by algorithm INTERVAL$_\alpha$, for the situation where the repairman receives a fixed amount of advanced notice; i.e., if a city is released at time r, then the repairman learns of the city at time $(r-a)$ for some fixed a.

Let $\lambda = (1+\sqrt{2})$, b_0 equal the first release date r_j such that $r_j > \frac{a}{\lambda}$ and $b_i = \lambda^i b_0$. Also, let $\tilde{b}_i = b_i - a$. The latter $\tilde{b}_i$ parameters are the breakpoints where the online algorithm BREAK (to be defined shortly) will generate some re-optimization. This is a generalization of INTERVAL$_\alpha$ by Krumke et al. (2003), 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 $Q_i \subseteq Q_{i+1}$. Note that at time $\tilde{b}_i$ the online salesman 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 $w(S) = \sum_{i \in S} w_i$. Online algorithm BREAK is as follows:

Definition 1 *Online algorithm BREAK:*

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.*

We now prove Lemma 4 from Krumke et al. (2003) for the new algorithm BREAK, assuming the optimal parameters $\alpha = (1+\sqrt{2})$ and $\beta = 1$. This proof also serves as an alternate proof of Lemma 4 from Krumke et al.; we need only set $a = 0$.

Lemma 4 $\sum_{i=1}^k w(R_i) \geq \sum_{i=1}^k w(R_i^*)$ *for* $k = 1, 2, \ldots$.

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 server were at the origin at time zero, he could obviously serve all the cities in the set R by time b_k.

Now, consider an online server 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 (due to the definition of algorithm BREAK), the server could serve the cities in R by time $\tilde{b}_k + b_{k-1} + b_k = \tilde{b}_{k+1}$ (equality since $\alpha = (1+\sqrt{2})$ and $\beta = 1$). Thus, in iteration k, *if* faced with the set R, the server could serve cities of total weight $w(R)$.

Unfortunately, it is not clear whether the online server will encounter the set R, since the R_i^* are not known until all cities are released. However, the server'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 server 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,

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

which gives the result. ■