TechReport: Competitive Analysis for Two-Level Ski-Rental Problem

Binghan Wu, Wei Bao, Dong Yuan
The University of Sydney

Proof of Lemma 1

Proof. Let $D(k)$ be the demand of instance k in sequence s and $D'(k)$ be the demand of instance k in s'. The cost of offline optimal given input s is:

$$\text{OPT}(s) = \min(C_c, \sum_{i=1}^{K} \min(C_s, D(i))), \tag{1}$$

and similarly

$$\text{OPT}(s') = \min(C_c, \sum_{i=1}^{K} \min(C_s, D'(i))). \tag{2}$$

Since $\lambda_s \leq C_s$, we have

$$\sum_{i=0}^{K} \min(C_s, D(i)) = \sum_{i=1}^{K} \min(C_s, D'(i)), \tag{3}$$

and thus

$$\text{OPT}(s) = \text{OPT}(s'). \tag{4}$$

Therefore, with both input sequences s and s', the cost of the offline optimal algorithm is the same.

With the sequence s, Algorithm 1 will not make single purchases for instance k or l, but it will make a single purchase for instance k with s', which will lead to

$$\text{DTSR}(s', \lambda_s, \lambda_c) - \text{DTSR}(s, \lambda_s, \lambda_c) = C_s - 1 > 0. \tag{5}$$

Therefore, the relationship of competitive ratio can be concluded as:

$$\frac{\text{DTSR}(s', \lambda_s, \lambda_c)}{\text{OPT}(s')} > \frac{\text{DTSR}(s, \lambda_s, \lambda_c)}{\text{OPT}(s)}. \tag{6}$$

In summary, Lemma 1 is proved. □

Proof of Lemma 2

Proof. In this proof, we use the same set of notations as we did in the proof of Lemma 1, and thus (1) and (2) are still true.

Since $D(k), D(l), D'(k)$, and $D'(l)$ are all less than or equal to C_s, we have:

$$\sum_{i=1}^{K} \min(C_s, D(i)) = \sum_{i=1}^{K} \min(C_s, D'(i)), \tag{7}$$

and thus

$$\text{OPT}(s) = \text{OPT}(s'). \tag{8}$$

Since $D(k) < \lambda_s$, $D(l) > \lambda_s$, $D'(k) \leq \lambda_s$, and $D'(l) \geq \lambda_s$, we have

$$\begin{aligned} &\text{DTSR}(s', \lambda_s, \lambda_c) - \text{DTSR}(s, \lambda_s, \lambda_c) \\ &= C_s + \lambda_s - D(k) - 1 \\ &> 0, \end{aligned} \tag{9}$$

if $D'(k) = \lambda_s$; or

$$\text{DTSR}(s', \lambda_s, \lambda_c) - \text{DTSR}(s, \lambda_s, \lambda_c) = \lambda_s - D(k) > 0, \tag{10}$$

otherwise.

In both cases, the relationship of competitive ratio can be concluded as:

$$\frac{\text{DTSR}(s', \lambda_s, \lambda_c)}{\text{OPT}(s')} > \frac{\text{DTSR}(s, \lambda_s, \lambda_c)}{\text{OPT}(s)}. \tag{11}$$

In summary, Lemma 2 is proved. □

Proof of Lemma 4

Proof.

Analysis of the standard sequence

Recall the characteristics of s_{std}:

- m instances have demand no less than C_s, where $m \geq 0$.
- n instances have demand equal to λ_s, where $n \geq 0$.
- At most 1 instance has demand x in $(0, \lambda_s)$.
- All other instances have 0 demand.

Given a standard input sequence s_{std}, the cost of the offline optimal is

$$\text{OPT}(s_{std}) = \min(C_c, mC_s + n\lambda_s + x). \tag{12}$$

Thus, the behavior of the offline optimal can be divided into two cases: (1) the combo purchase is not made and $\text{OPT}(s_{std}) = mC_s + n\lambda_s + x$. (2) the combo purchase is made and $\text{OPT}(s_{std}) = C_c$.

For Algorithm 1 we also consider two cases: (A) Algorithm 1 does not make the combo purchase, and thus

$$\text{DTSR}(s_{std}, \lambda_s, \lambda_c) = (m+n)(\lambda_s - 1 + C_s) + x. \tag{13}$$

(B) Algorithm 1 makes the combo purchase, and thus

$$\text{DTSR}(s_{std},\lambda_s,\lambda_c)=\lambda_c-1+C_c+\min(\lfloor\frac{\lambda_c-1}{\lambda_s-1}\rfloor,m+n)C_s. \tag{14}$$

Since the rental cost is at most λ_s-1 in total and each instance needs rental cost λ_c-1 to make a single purchase, the algorithm can only make at most $\lfloor\frac{\lambda_c-1}{\lambda_s-1}\rfloor$ single purchases. On the other hand, the number of instances which have possibility to make single purchases is restricted by $m+n$, so that we have the $\min$ term in the formula.

Therefore, we consider $2\times 2=4$ cases. Let

$$R(s,\lambda_s,\lambda_c)=\frac{\text{DTSR}(s,\lambda_s,\lambda_c)}{\text{OPT}(s)}. \tag{15}$$

Case 1-A:

$$R(s_{std},\lambda_s,\lambda_c)=\frac{(m+n)(\lambda_s-1+C_s)+x}{mC_s+n\lambda_s+x}. \tag{16}$$

Case 1-B:

$$\begin{aligned}&R(s_{std},\lambda_s,\lambda_c)\\&=\frac{\lambda_c-1+C_c+\min(\lfloor\frac{\lambda_c-1}{\lambda_s-1}\rfloor,m+n)C_s}{mC_s+n\lambda_s+x}.\end{aligned} \tag{17}$$

Case 2-A:

$$R(s_{std},\lambda_s,\lambda_c)=\frac{(m+n)(\lambda_s-1+C_s)+x}{C_c}. \tag{18}$$

Case 2-B:

$$\begin{aligned}&R(s_{std},\lambda_s,\lambda_c)\\&=\frac{\lambda_c-1+C_c+\min(\lfloor\frac{\lambda_c-1}{\lambda_s-1}\rfloor,m+n)C_s}{C_c}.\end{aligned} \tag{19}$$

Analysis of the four cases

For case 1-A, we have:

$$\begin{aligned}R(s_{std},\lambda_s,\lambda_c)&=\frac{(m+n)(\lambda_s-1+C_s)+x}{mC_s+n\lambda_s+x} &(20)\\&\le\frac{(m+n)(\lambda_s-1+C_s)}{(m+n)\lambda_s} &(21)\\&\le\frac{\lambda_s-1+C_s}{\lambda_s}. &(22)\end{aligned}$$

For the rest of three cases 1-B, 2-A, and 2-B, we can merge them as follows.

$$\begin{aligned}&R(s_{std},\lambda_s,\lambda_c)\\&\le\frac{\lambda_c-1+C_c+\min(\lfloor\frac{\lambda_c-1}{\lambda_s-1}\rfloor,m+n)C_s}{\min(C_c,mC_s+n\lambda_s+x)},\end{aligned} \tag{23}$$

and we are going to focus on the "worst" case of (23), which is

$$\begin{aligned}&R^*(s_{std},\lambda_s,\lambda_c)\\&=\max_{s_{std}}\min_{\lambda_s,\lambda_c}\frac{\lambda_c-1+C_c+\min(\lfloor\frac{\lambda_c-1}{\lambda_s-1}\rfloor,m+n)C_s}{\min(C_c,mC_s+n\lambda_s+x)}.\end{aligned} \tag{24}$$

Let s^*_{std} be the input that maximizes this formula.

We claim that in s^*_{std}, $m=0$, because for $\forall m_1\neq 0$ and n_1, $n_2=m_1+n_1$, we have

$$\begin{aligned}&\frac{\lambda_c-1+C_c+\min((m_1+n_1),\lfloor\frac{\lambda_c-1}{\lambda_s-1}\rfloor)C_s}{\min(m_1C_s+n_1\lambda_s+x,C_c)}\\&\le\frac{\lambda_c-1+C_c+\min(n_2,\lfloor\frac{\lambda_c-1}{\lambda_s-1}\rfloor)C_s}{\min(n_2\lambda_s+x,C_c)},\end{aligned} \tag{25}$$

and thus

$$\begin{aligned}&R^*(s_{std},\lambda_s,\lambda_c)\\&=\max_{s_{std}}\min_{\lambda_s,\lambda_c}\frac{\lambda_c-1+C_c+\min(n,\lfloor\frac{\lambda_c-1}{\lambda_s-1}\rfloor)C_s}{\min(n\lambda_s+x,C_c)} &(26)\\&=\max_{s_{std}}\min_{\lambda_s,\lambda_c}\frac{\lambda_c-1+C_c+nC_s}{\min(n\lambda_s+x,C_c)}, &(27)\\&\text{where}\quad n\le\lfloor\frac{\lambda_c-1}{\lambda_s-1}\rfloor.\end{aligned}$$

For $\min(n\lambda_s+x,C_c)$, it can be divided into two cases: (I) $n\lambda_s+x\ge C_c$; (II) $n\lambda_s+x\le C_c$. Note that case (I) and (II) has intersection when $n\lambda_s+x=C_c$, which is set for the convenience of analysis shortly.

Given a pair of input parameters λ_s and λ_c, n and x are chosen by the adversarial s_{std}. In (27), Algorithm 1 makes the combo purchase, so that we further claim that in s^*_{std}, $x\ge 1$. If $x=0$, there are at least one instance k such that $D(k)=\lambda_s$ and Algorithm 1 does not make the single purchase for instance k. However, the combo purchase will be made when the cumulative demand of instance k reaches some d, where $d\in[1,\lambda_s-1]$. Then, we can remove λ_s-d demand from instance k. By doing so, the cost of Algorithm 1 is the same, and the optimal cost may reduce. Thus, we have

$$x\in[1,\lambda_s-1], \tag{28}$$

and correspondingly

$$n\in[\frac{\lambda_c}{\lambda_s-1}-1,\frac{\lambda_c-1}{\lambda_s-1}]. \tag{29}$$

Here, we remove the rounding sign of $\frac{\lambda_c-1}{\lambda_s-1}$. We will give an example at the end to show that the bound can be reached in the worst case.

By taking derivative of (27) to n, we can get the two conditional choices of n: (a) when $\lambda_c\ge\frac{C_c}{C_s-1}+1$, $n=\frac{\lambda_c-1}{\lambda_s-1}$; (b) when $\lambda_c<\frac{C_c}{C_s-1}+1$, $n=\frac{\lambda_c}{\lambda_s-1}-1$.

Therefore, there are 2×2 cases to discuss: a-I, a-II, b-I, and b-II. For cases a-I and a-II, (27) is maximized when $n\lambda_s+x=C_c$ which is right set on the intersection. Also, we have

$$\lambda_s=C_s, \tag{30}$$

$$\lambda_c=\frac{C_s-1}{C_s}C_c+1, \tag{31}$$

$$R(s^*_{std},\lambda_s,\lambda_c)=3-\frac{1}{C_s}. \tag{32}$$

Case b-I will lead to zero λ_s which further leads to arbitrarily large competitive ratio. Case b-II will lead to

$$R(s^*_{std}, \lambda_s, \lambda_c) = C_s \gg 1. \quad (33)$$

Therefore, Algorithm 1 will choose case (a), and it leads to competitive ratio of $3 - \frac{1}{C_s}$.

At last, (32) > (22). So the adversary will choose the input leading to (32).

Proof of the tightness

For any given instances number K, we can construct the parameters in the following procedure so that the competitive ratio will exactly be $3 - \frac{1}{C_s}$:

1. Given arbitrary C_s.
2. Let $C_c = (K-1)C_s$.
3. Let s^* be the input where $K-1$ instances have demand C_s and 1 instance has demand 1.

It can be verify that the rental cost of Algorithm 1 is $(K-1)*(C_s-1) = \lambda_c - 1$. The single purchase costs $(K-1)*C_s$ and the cost of the combo purchase is C_c for Algorithm 1. The offline optimal only makes the combo purchase with cost C_c. The competitive ratio is:

$$R(s^*, \lambda_s, \lambda_c) = \frac{\frac{C_c}{C_s}(C_s - 1) + C_c + C_c}{C_c} = 3 - \frac{1}{C_s}. \quad (34)$$

Therefore, the bound is tight.

In summary, Lemma 4 is proved.

□

Proof of Theorem 1

Proof. We want to find the tight bound of the following max-min problem:

$$\max_{s_{std}} \min_{\lambda_s, \lambda_c} \frac{\text{DTSR}(s_{std}, \lambda_s, \lambda_c)}{\text{OPT}(s_{std})}. \quad (35)$$

The algorithm can choose λ_s and λ_c between two cases: (A) $\lambda_s \leq \lambda_c$ and (B) $\lambda_s > \lambda_c$. Given λ_s and λ_c, the adversary can choose an input sequence s in the following 2 cases: (1) The total demand $\mathbf{D}(s) < \min(\lambda_s, \lambda_c)$; (2) The total demand $\mathbf{D}(s) \geq \min(\lambda_s, \lambda_c)$.

In both cases A-1 and B-1, since $\mathbf{D}(s) < \min(\lambda_s, \lambda_c)$, no single purchase or combo purchase will be made by both Algorithm 1 and the offline optimal. So we have competitive ratio of 1.

Case A-2 is directly driven from Lemma 4. The competitive ratio is $3 - \frac{1}{C_s}$.

In case B-2, since $\lambda_c < \lambda_s$, Algorithm 1 will not make a single purchase given any input sequence s, so that the competitive ratio is

$$\frac{\lambda_c - 1 + C_c}{\lambda_c} \quad (36)$$

$$> 1 + \frac{C_c - 1}{\lambda_s} \quad (37)$$

$$> 1 + \frac{C_c - 1}{C_s} \quad (38)$$

$$> 3 - \frac{1}{C_s}. \quad (39)$$

The last line is derived by the assumption $C_c/C_s > 2$.

Whatever the adversary chooses, case (A) is better than (B) for the algorithm. Given that the algorithm chooses case (A), the adversary will choose case (1) to maximize the competitive ratio. Therefore, case A-2 is the equilibrium and also the solution to this max-min problem. As we have proved in Lemma 4, this competitive ratio bound is tight. Therefore, Theorem 1 is proved. □

Proof of Corollary 1

Proof. The analysis for Corollary 1 begins with (27). By considering the adversary's action, we have

$$\max_{s^*_{std}} \frac{\lambda_c - 1 + C_c + nC_s}{\min(n\lambda_s + x, C_c)} \quad (40)$$

$$\leq \max_{s^*_{std}} \frac{\lambda_c - 1 + C_c + nC_s}{n\lambda_s + x} \quad (41)$$

$$< \max_{s^*_{std}} \frac{\lambda_c - 1 + C_c + nC_s}{n\lambda_s} \quad (42)$$

$$< 1 + \frac{C_c}{\lambda_c - 1} + \frac{C_s}{\lambda_s}. \quad (43)$$

From (42) to (43), we relaxed the range of n to $[\frac{\lambda_c - 1}{\lambda_s}, \frac{\lambda_c - 1}{\lambda_s - 1}]$ comparing to (29), and thus (43) is derived by applying $n = \frac{\lambda_c - 1}{\lambda_s}$ to (42).

□

Proof of Theorem 2

Proof. Let s^D be an arbitrary sequence with total demand D. We consider the mean performance under different D values.

Case 1: If $D < \min(\lambda_s, \lambda_c)$, both RTSR and OPT pay for the demands by rental.

$$R(s^D, \lambda_s, \lambda_c) = 1. \quad (44)$$

Case 2: If $\lambda_s \leq D$ and $\lambda_s < \lambda_c$, from Corollary 1 and Lemma 3:

$$R(s^D, \lambda_s, \lambda_c) \leq 1 + \frac{C_c}{\lambda_c - 1} + \frac{C_s}{\lambda_s}. \quad (45)$$

Case 3: If $\lambda_c \leq D$ and $\lambda_c \leq \lambda_s$, RTSR will make the combo purchase and RTSR$(s^D, \lambda_s, \lambda_c) \leq (\lambda_c - 1 + C_c)$. Therefore,

$$R(s^D, \lambda_s, \lambda_c) \leq \frac{\lambda_c - 1 + C_c}{D}. \quad (46)$$

Define E_D as the competitive ratio of Algorithm 3 given the input sequence s with total demand D. We want to show that E_D is bounded by a competitive ratio c for any D.

Recall that we have randomly distributed λ_s and λ_c, with $P(\lambda_s = i) \triangleq P_i^{(s)}$ and $P(\lambda_c = i) \triangleq P_i^{(c)}$. We also define $\sum_{\lambda_s = i}^{C_s} P(\lambda_s = i) \triangleq F_i^{(s)}$ and $\sum_{\lambda_c = i}^{C_c} P(\lambda_c = i) \triangleq F_i^{(c)}$.

By taking (44)–(46) into account, we have

$$\begin{aligned} E_D \leq& 1 \cdot F^{(s)}_{D+1} F^{(c)}_{D+1} \\ &+ \sum_{i=1}^{D} \sum_{j=i}^{C_c} (1 + \frac{C_c}{j-1} + \frac{C_s}{i}) P^{(s)}_i P^{(c)}_j \\ &+ \sum_{i=1}^{D} \frac{i-1+C_c}{D} P^{(c)}_i F^{(s)}_{i+1}. \end{aligned} \tag{47}$$

Recall that the probability mass functions (PMFs) are stated as the follows.

$$\mathbb{P}(\lambda_s = i) \triangleq P^{(s)}_i = \begin{cases} 1, & \text{if } i = C_s, \\ 0, & \text{otherwise}, \end{cases} \tag{48}$$

$$\mathbb{P}(\lambda_c = i) \triangleq P^{(c)}_i = \begin{cases} aq^{i-1}, \text{ if } i \in [1, C_s - 1], \\ 1 - \frac{a(1-q)^{C_s-1}}{1-q}, \text{ if } i = C_c, \\ 0, \text{ otherwise}, \end{cases} \tag{49}$$

where $q \triangleq \frac{C_c}{C_c-1}$ and $a \triangleq \frac{3C_c-2}{C_c-1} \cdot \frac{1}{(3C_c-2)(e^\sigma-1)+\sigma(e^\sigma+1)-e^\sigma/C_s}$, and $\sigma \triangleq \frac{C_s-1}{C_c-1}$.

By taking (48) and (49) into (47), we can verify that $E_1 = E_2 = \ldots = c$, in which $c = (C_c-1)aq^{C_s-1}$, where q and a are defined below (49).

Let $\sigma = \frac{C_s-1}{C_c-1}$. Since $(1+1/\epsilon)^\epsilon$ approaches to e when ϵ is large, $q^{C_s-1} = (1+\frac{1}{C_c-1})^{C_s-1} \simeq e^\sigma$ because $C_s \gg 1$ and $C_c \gg 1$. As a result

$$c \simeq \frac{e^\sigma}{e^\sigma - 1}. \tag{50}$$

Please note that we can consider any distributions of $P^{(s)}_i$ and $P^{(c)}_i$, leading to other competitive ratios c' as long as $E_D \leq c', \forall D$. However, we use (48) and (49) because they show the following benefits: (1) They are in the closed form. (2) λ_s is equal to C_s with probability 1, leading to less complicated implementation. (3) c is the same for all D values, which gives a local minimum of c. $\square$