Dynamic Pricing for Reusable Resources in Competitive Market with Stochastic Demand

Jiang Rong¹, Tao Qin², Bo An³

1 The Key Lab of Intelligent Information Processing, ICT, CAS
2 University of Chinese Academy of Sciences, Beijing 100190, China
3 Microsoft Research, Beijing 100080, China
4 School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798

Abstract

The market for selling reusable products (e.g., car rental, cloud services and network access resources) is growing rapidly over the last few years, where service providers maximize their revenues through setting the optimal prices. While there has been lots of research on pricing optimization, existing works often ignore dynamic property of demand and the competition among providers. Thus, existing pricing solutions might be far from optimal in realistic markets. This paper provides the first study of service providers' dynamic pricing in consideration of market competition and makes three key contributions along this line. First, we propose a comprehensive model that takes into account the dynamic demand and interaction among providers, and formulate the optimal pricing policy in the competitive market as an equilibrium. Second, we propose an approximate Nash equilibrium to describe providers' behaviors, and design an efficient algorithm to compute the equilibrium which is guaranteed to converge. Third, we derive many properties of the model without any further constraints on demand functions, which can reduce the search space of policies in the algorithm. Finally, we conduct extensive experiments with different parameter settings, showing that the approximate equilibrium is very close to the Nash equilibrium and our proposed pricing policy outperforms existing strategies.

1 Introduction

In many real-world applications, the service providers’ resources are reusable and the number of available products changes over time because of the arrival and departure of customers’ demand. Dynamic pricing policy plays an important role in making profits from price-sensitive users, which has shown great success in industries, e.g., the car rentals (Geraghty and Johnson 1997), hotel reservations (Choi and Mattila 2004; Weatherford and Kimes 2003), network services (Paschalidis and Tsitsiklis 2000), and the cloud computing (Kantere et al. 2011; Xu and Li 2012), and has attracted lots of research attention (Sen 2013; Elmgharaby and Keskinocak 2003; Levin, McGill, and Nediak 2009; Wang et al. 2015; Xu et al. 2015). There are two important properties for the market: 1) users’ demand is stochastic over time, which leads to dynamic inventories of providers; and 2) providers that offer similar services need to compete against each other. However, existing works have partially neglected or treated these characteristics in an inadequate way and thus their pricing policies might lead to poor performance in the realistic market. Against this background, this paper investigates dynamic pricing to match demand with inventory in order to maximize providers’ long-term revenues in the competitive market, which gives solid theoretical and experimental analyses and makes three key contributions.

First, we propose a comprehensive model to describe the real-world applications with multiple providers and stochastic user demand, where a product can be reused, e.g., resources in a cloud platform. Existing works ignore either the competition or the dynamic feature. Demand forecast is studied in (Geraghty and Johnson 1997; Weatherford and Kimes 2003) and the most widely-used model to describe users’ dynamic demand is the Poisson process (Elmaghraby and Keskinocak 2003; Gallego and Van Ryzin 1994; Paschalidis and Tsitsiklis 2000; Xu and Li 2012). However, those works do not consider the market competition. Xu and Hopp (2006) assume that customers’ arrival rates follow the geometric Brownian motion and the perfect Bayesian equilibrium is used to model providers’ behaviors. Levin et al. (2009) consider strategic users and propose the subgame-perfect equilibrium. However, they focus on the one-shot inventory replenishment problem with dynamic pricing, which cannot describe the market with reusable products. In this paper, we consider both stochastic user demand and competition from providers. Following the common practice in the literature, the demand is also described by Poisson process, based on which we formulate the dynamic and competitive market as continuous-time Markov chains (Gopalratnam, Kautz, and Weld 2005; Norris 1998; Simmons and Younes 2004).

Since each provider aims to maximize his/her expected revenue, the optimal policy is supposed to be a Nash Equilibrium (NE). We show that it is difficult for providers to reach the NE in real world because a provider does not have the full information of others and his/her revenue cannot be explicitly represented as a function of his/her pricing policy. Our second contribution is then to propose an Approximate Equilibrium (AE) solution concept (Nisan et al. 2007;
Tsankalis and Spirakis 2008) and to design an algorithm based on the best-response principle to efficiently compute the AE, which is guaranteed to converge to an AE.

Third, we derive many properties of the model. The setting where the provider charges a customer a fee per call, no matter how long the customer uses the service is analyzed in (Paschalidis and Tsitsiklis 2000; Xu and Li 2013) study the monotonicity of the dynamic pricing policy for the cloud market, but their results need the demand functions to satisfy particular conditions, while our results do not make special assumptions on the demand process. The properties we found exhibit the monotonicity and concavity about the expected overall revenue and the monotonicity of AE policies with respect to the capacity utilization. These results are then used to reduce the search space in the computation of equilibrium strategies.

We conduct extensive experiments to evaluate our algorithm which shows good convergence performance. The results indicate that our pricing policy outperforms existing strategies and the proposed AE is very close to the NE.

2 Modeling Competitive Market with Stochastic Demand

2.1 Motivation Example

Competition is one of the key features of today’s service business, e.g., the cloud market, where different companies provide similar resources to users over the Internet (Wang et al. 2015; Xu et al. 2015). The most widely-used cloud services include Amazon’s AWS and Microsoft’s Azure. Another feature of the market is that users’ demand and providers’ inventories are dynamic, e.g., the number of virtual machines occupied by users in a cloud platform is dynamic. Since users are generally price-sensitive, one should strategically set prices to influence demand so as to better utilize unused capacity. Indeed, Amazon EC2 has introduced the “spot pricing” to dynamically update the price for a virtual instance. In this paper, we investigate providers’ optimal dynamic pricing policies to maximize their expected revenues in the competitive market. We first give our model in the following two subsections.

2.2 Stochastic Demand

We use $K$ to represent the set of service providers in the market. Following the common practice in the literature (Elmaghraby and Keskinocak 2003; Paschalidis and Tsitsiklis 2000; Xu and Li 2013; Xu and Hopp 2006), we assume that users’ demand for the service of provider $k \in K$ is determined by two independent Poisson Processes, namely the arrival process that models the coming of new demand and the departure process that corresponds to the leaving of existing requests, which are related to the provider’s own price $p_k$ and other providers’ prices $p_{-k}$ considering the market competition. Specifically, we use $\lambda_k(\cdot)$ to represent the Poisson arrival rate (number of new demand instances per unit time) for provider $k$, which satisfies the following properties (Dockner and Jørgensen 1988):

$$\lambda_k(p) \geq 0; \quad \frac{\partial \lambda_k(p)}{\partial p_k} < 0; \quad \frac{\partial \lambda_k(p)}{\partial p_{k'}} > 0, \quad \forall k' \neq k.$$  

where $p = (p_1, p_2, \ldots, p_{|K|})$. The above equations are consistent with the reality, where 1) the arrival rate can never be negative; 2) decreasing provider $k$’s own price will attract more new users to him/her; and 3) when others’ prices increase, some users may turn to choose $k$’s service and hence the arrival rate of $k$ increases. Similarly, the Poisson departure process is modeled by $\mu_k(\cdot)$, which satisfies that:

$$\mu_k(p) \geq 0; \quad \frac{\partial \mu_k(p)}{\partial p_k} > 0; \quad \frac{\partial \mu_k(p)}{\partial p_{k'}} < 0, \quad \forall k' \neq k. \tag{2}$$

We use the notation $(p_k, p_{-k}) = p$ and use $\delta_k$ to represent provider $k$’s discrete pricing space. The minimal price $p_k^{\text{min}}$ and maximal price $p_k^{\text{max}}$ of $\delta_k$ satisfy that $\mu_k(p_k^{\text{min}}, p_{-k}) = 0$ and $\lambda_k(p_k^{\text{max}}, p_{-k}) = 0$, respectively, $\forall p_{-k} \in \delta_k$, where $\delta_k = \{x_{i \in C(k)}\delta_i$ and $C(k) = K \setminus \{k\}$. We assume that both $\lambda_k(p_k, p_{-k})$ and $\mu_k(p_k, p_{-k})$ are bounded since in the real world providers cannot gain infinite arrival and departure rates. Let $N_k$ be the maximal capacity (i.e., total number of available instances) of provider $k$ and $|N_k|$ denote the set $\{0, 1, \ldots, N_k\}$. Since both the arrival and departure of demand are random process, the number of instances used by customers can be formulated as a continuous-time Markov process, where provider $k$’s state $n \in [N_k]$ is the number of his/her used instances. The pricing policy of provider $k$ is represented as $P_k = (p_{k,0}, p_{k,1}, \ldots, p_{k,N_k})$, where $p_{k,n}$ is the price set for state $n$. We have that $P_k \in \Delta_k$, where $\Delta_k = \times_{n=0}^{N_k} \delta_k$. We define $P_{-k} = \times_{i \in C(k)} P_i$ as the policy profile of other providers except $k$ and use the notations $\Delta_{-k} = \times_{i \in C(k)} \Delta_i$ and $P = (P_k, P_{-k}), \forall k \in K$.

2.3 Multiple-Provider Model

We first introduce the single-provider model. Assume that provider 1 is the only one in the market and then the transition rate matrix (Guo and Hernández-Lerma 2009; Norris 1998) of the Markov process for his/her state can be written as $Q_1(P_1) = (q_{i,j}^1(P_1))_{i,j \in [N_1]}$:

$$q_{i,j}^1(P_1) = \begin{cases} \lambda_1(p_{1,i}), & \text{if } j = i + 1; \\ \mu_1(p_{1,i}), & \text{if } j = i - 1; \\ -\sum_{i' \neq i} q_{i,i'}^1(P_1), & \text{if } j = i; \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

where $q_{i,j}^1(P_1)$ represents the rate of the process transition from state $i$ to state $j$. In the long-term view, the probability of the appearance of state $n$ in the continuous-time Markov process, denoted by $\pi_n(P_1), n \in [N_1]$, satisfies that $\sum_{n \in [N_1]} \pi_n(P_1) = 1$ and $\pi_1(P_1)\cdot Q_1(P_1) = 0$, where $\pi_1(P_1) = (\pi_{1,0}(P_1), \pi_{1,1}(P_1), \ldots, \pi_{1,N_1}(P_1))$ is called the stationary (or steady-state) probability.

When there are multiple providers, their pricing policies can affect each other’s demand arrival and departure as shown in Eqs.(1)-(2) and hence the stationary probability of each provider $k$ is a function of $P$ (not only $P_k$). Let $\pi_k(P) = (\pi_{k,0}(P), \pi_{k,1}(P), \ldots, \pi_{k,N_k}(P)), \pi_{-k}(P) = \times_{i \in C(k)} \pi_i(P)$, and $\pi(P) = (\pi_k(P), \pi_{-k}(P))$. Then the transition rate matrix for provider $k$ is $Q_k(P)$ =
\((q_{i,j}^k(P))_{i,j}, i,j \in [N_2]\):
\[
q_{i,j}^k(P) = \begin{cases} 
E_{p_{-k}=\lambda_k(p_{k,i},p_{-k})}^\pi_k & \text{if } j = i + 1; \\
E_{p_{-k}=\mu_k(p_{k,i},p_{-k})}^\pi_k & \text{if } j = i - 1; \\
-\sum_{m \neq i,j} q_{m,n}^k(P), & \text{if } j = i; \\
0, & \text{otherwise,}
\end{cases}
\tag{4}
\]

where \(E_{p_{-k}=\lambda_k(p_{-k})}^\pi_k\) and \(E_{p_{-k}=\mu_k(p_{-k})}^\pi_k\) are the probability of pricing \(p_{-k}\) given \(\pi_k\).

Similar with the single-provider model, we have
\[
\sum_{n\in[N_2]} \pi_{k,n}(P) = 1; \quad \pi_k(P) \cdot Q_k(P) = 0.
\tag{5}
\]

When \(n\) instances are being used by customers, provider \(k\) can receive \(n \cdot p_{k,n}\) revenue per unit time. Thus the average expected revenue rate for provider \(k\) is
\[
J_k(P_k, P_{-k}) = \sum_{n=0}^{N_k} \pi_{k,n}(P) \cdot n \cdot p_{k,n}.
\tag{6}
\]

### 3 Optimal Dynamic Pricing

Since each provider aims to maximize his/her revenue rate while considering the policies of others, we need to study the equilibrium pricing policies. We first show that it is difficult for providers to compute their NE policies. To address this problem, we introduce an AE and propose an efficient algorithm to calculate the equilibrium strategies.

#### 3.1 Equilibrium Policies

**Definition 1 (Nash equilibrium).** A Nash equilibrium is a pricing policy profile \(P^* = (p_k^*), k \in K\), such that \(\forall k \in K\),
\[
J_k(P_k^*, P_{-k}^*) \geq J_k(P_k, P_{-k}^*), \forall P_k \in \Delta_k.
\tag{7}
\]

That is, no one can gain higher revenue rate by unilaterally changing his/her equilibrium policy. However, \(J_k(P_k, P_{-k})\) is not an explicit function with respect to \(P = (P_k, P_{-k})\). Specifically, by solving Eq.(5), we get that
\[
\pi_{k,n}(P) = 1/(\sum_{m=0}^{N_k-1} \prod_{j=0}^{m-1} \frac{q_{k,j+1,i}^k(p_{k,i},p_{-k})}{q_{j+1,i}^k(p_{k,i},p_{-k})} + 1 \\
+ \sum_{m=n+1}^{N_k} \prod_{j=0}^{m-1} \frac{q_{k,j+1,i}^k(p_{k,i},p_{-k})}{q_{j+1,i}^k(p_{k,i},p_{-k})}), \forall n \in [N_2].
\]

The above equation implies that \(\pi_{k,n}(P)\) cannot be explicitly represented with \(P\) since the term \(q_{i,j}^k(P)\), in turn, involves computing \(\pi_{-k}(P)\) (see Eq.(4)). Thus, the revenue rate \(J_k(P_k, P_{-k})\) is an implicit function with respect to \(P\), which makes it hard to compute the NE because it is equivalent to optimizing a set of non-linear functions with non-linear constraints and existing algorithms for NE calculation (including the best-response iteration (Truong-Huu and Tham 2013; Goemans, Mirrokni, and Vetta 2005) and quantal-response correspondence (Rong et al. 2016; Turocy 2005)) always require the explicit representation of the revenue function. Besides, computing the NE needs full information of all providers’ demand functions, which is usually unavailable in the real world. Thus, it is difficult for providers in the real-world application to read an NE. To address these problems, when we optimize provider \(k\)’s policy, we view the steady-state probabilities \(\pi_{-k} = (\pi_{1,-k}, \ldots, \pi_{N_1,-k})\) of others as fixed (i.e., they do not change with \(P\)). Provider \(k\)’s stationary probability under this assumption, \(\tilde{\pi}_k(P)\), can be calculated based on the following linear equations:
\[
\sum_{n\in[N_2]} \tilde{\pi}_{k,n}(P) = 1, \quad \tilde{\pi}_k(P) \cdot Q_k(P) = 0.
\]

The revenue function defined in Eq.(8) is an explicit function of \(P\). If all providers aim to maximize this revenue function, it follows that the resulting pricing policy, denoted by \(\hat{P}^* = \times_k \hat{P}_k^*,\) satisfies that \(\forall k \in K\) and \(P_k \in \Delta_k,\)
\[
J_k(\hat{P}_k^*, \hat{P}_{-k}^*|\pi_{-k}^*) \geq J_k(P_k, \hat{P}_{-k}^*|\pi_{-k}^*).
\tag{9}
\]

The policy \(\hat{P}^*\) is not an NE according to Definition 1, which is an AE, as defined below.

**Definition 2 (Approximate equilibrium).** An \(\epsilon\)-approximate equilibrium is a pricing policy profile \(\hat{P}^*\) with a vector \(\epsilon = (\epsilon_1, \epsilon_2, \ldots, \epsilon_{|K|})\), such that \(\forall k \in K,\)
\[
J_k(\hat{P}_k^*, \hat{P}_{-k}^*|\pi_{-k}^*) \geq J_k(P_k, \hat{P}_{-k}^*|\pi_{-k}^*) + \epsilon_k.
\tag{10}
\]

We see that \(\epsilon\) measures the difference between \(\hat{P}^*\) and \(P^*\) (i.e., how approximate the strategy \(\hat{P}^*\) is to the NE \(P^*\)), and if \(\epsilon_k = 0\) for all \(k \in K\), then \(P^* = \hat{P}^*\). Since \(J_k(\hat{P}_k^*, \hat{P}_{-k}^*|\pi_{-k}^*) = J_k(\hat{P}_k^*, \hat{P}_{-k}^*)\), we have that \(\epsilon_k = \max_{P_k \in \Delta_k} J_k(P_k, \hat{P}_{-k}^* - J_k(\hat{P}_k^*, \hat{P}_{-k}^*|\pi_{-k}^*)\). That is, \(\epsilon_k\) can be viewed as the additional revenue provider \(k\) can gain by unilaterally deviating from \(\hat{P}^*\). Note that \(\hat{P}^*\) is the solution of Eq.(9), but not a function of \(\epsilon\). That is, given other parameters of the model, both \(\hat{P}^*\) and \(\epsilon\) are fixed. We show in experiments that \(\epsilon\) is very small.

We demonstrate that the policy \(\hat{P}^*\) is more practical than \(P^*\) in the real world because provider \(k\) usually does not know others’ demand functions \((\lambda_k(\cdot)\) and \(\mu_k(\cdot)\) for all \(k' \neq k\)) and thus cannot compute the NE \(P^*\). However, each provider can observe others’ \(P_{-k}\) and \(\pi_{-k}\) and then optimize his/her policy, which may make the policy to converge to \(\hat{P}^*\) eventually, as discussed in next section.

#### 3.2 Equilibrium Computation

The equilibrium solution concept implies that each provider’s policy is a best response to others’ policies. Motivated by this observation, the concept of Strategy Profile Graph (SPG) is introduced, which is a directed graph with nodes representing players’ strategy profiles and edges corresponding to best response actions of players, and the Best Response Dynamics (BRD) algorithm is proposed to compute the equilibrium (Chien and Sinclair 2007; Goemans, Mirrokni, and Vetta 2005; Nisan et al. 2007), the high level idea of which is to start from a node of SPG and then repeat-
edly transit to the next node along a best-response edge until there are not successive nodes. This algorithm is proved to converge in Congestion and Potential Games (Chien and Sinclair 2007), but it may fall into a cycle of the SPG and never converge in general normal-form games (Goemans, Mirrokni, and Vetta 2005). In our model, $\hat{P}^*$ can be viewed as a best response to others’ strategy profile $\hat{P}^*$ and their fixed stationary probability profile $\pi_{-k}(\hat{P}^*)$. We formally define provider $k$’s the best response to $\hat{P}^*$ and $\pi_{-k}$ as

$$\hat{B}_k(P_{-k}|\pi_{-k}) = \arg\max_{P_k\in\Delta_k} \hat{\gamma}_k(P_k, P_{-k}|\pi_{-k}).$$

(11)

The details of calculating $\hat{B}_k(P_{-k}|\pi_{-k})$ will be given in next section. Then we can build a SPG as follows: a node in the graph corresponds to a pricing policy profile $P \in \Delta$ and the set of outgoing edges of node $P$ is $\{\hat{B}_k(P_{-k}|\pi_{-k})|k \in K\}$, where $\pi_{-k} = \pi_{-k}(P)$. To address the non-convergence problem of the BRD algorithm, we design Algorithm 1 based on the SPG, which ensures to converge to the $\hat{P}^*$ if it exists in the game. The high-level idea is to search the nodes of the SPG for $\hat{P}^*$ in a depth-first manner, during which we delete the incoming edges of a searched node to make sure each node is visited at most once.

**Algorithm 1: Equilibrium Computation**

1. $s \leftarrow$ the set of strategy profiles;
2. $\pi_s \leftarrow$ the set of nodes that $s$ can transit to in the SPG, $\forall s \in S$;
3. while True do
4. Randomly choose a starting node $s \in S$;
5. if $s$ satisfies Eq. (9) then return $s$;
6. Initialize the path $A$ as empty;
7. while True do
8. if $\pi_s$ is not empty then
9. Randomly select an $s'$ in $\pi_s$;
10. if $s'$ satisfies Eq. (9) then return $s'$;
11. Append $s'$ to $A$;
12. $s \leftarrow s'$;
13. else if $A$ is empty then Break;
14. $s \leftarrow$ the last element of $A$;
15. Delete $s$ from $A$;
16. end
8. end
9. end
10. end
11. end
12. Delete $s$ from $A$;
13. end
14. end
15. end
16. end

**Theorem 1.** Algorithm 1 ensures to converge to a $\hat{P}^*$ if it exists.

**Proof.** We first prove that the second loop in line 7 always stops. Beginning with a starting node $s_0$, Algorithm 1 searches for the $\hat{P}^*$ in lines 7-17, which follows the rule that 1) if a node $s$ has a child node $s'$ (lines 9-10), append $s$ to path $A$ (line 12) and transit to $s'$ (line 13), otherwise (line 14), 2) go back to $s$’s parent node (line 16), delete its parent from path $A$ (line 17) and then apply the rule to its parent in next iteration. Loop 2 stops when a $\hat{P}^*$ is not found (line 11) or $A$ is empty (line 15), otherwise, the length of $A$ either increases (in line 12) or decreases (in line 17). Therefore, to prove that loop 2 always stops, we just need to prove that $A$’s length will stop changing at some time. Since we delete $s$ from the SPG (line 8) when the algorithm transit to it (thus each node can be transited to at most once) and $|S|$ is finite, the length of $A$ will stop increasing at some time. Thus if a $\hat{P}^*$ is found (line 11) or no node can be transited to. From that time on, $A$’s length can only decrease, which stops when $A$ is empty (line 15). Hence, loop 2 always stops.

If a $\hat{P}^*$ is found when loop 2 stops, our algorithm terminates, otherwise (if $A$ is empty), it means that all the nodes $s_0$ can transit to do not connect to a $\hat{P}^*$. Thus if there are paths from a new starting node $s'_0$ to $\hat{P}^*$ in the original SPG, the nodes that have been deleted did not cut off the paths. Since $|S|$ is finite and becomes smaller and smaller, a starting node that connects the $\hat{P}^*$ will eventually be sampled from $S$ (line 4) if the $\hat{P}^*$ exists, i.e., the first loop in line 3 always stops and returns the $\hat{P}^*$. \qed

In our experiments, Algorithm 1 always returns a $\hat{P}^*$. We discuss how to extend the algorithm to address the problem when $\hat{P}^*$ does not exist in Section 6.

### 3.3 Best Response Calculation

In this subsection we show how to compute $\pi(P)$ and the best response $\hat{B}_k(P_{-k}|\pi_{-k})$ to $P_{-k}$ and $\pi_{-k} = \pi_{-k}(P)$ in Algorithm 1. The former can be calculated with standard Newton-style methods. Next we focus on $\hat{B}_k(P_{-k}|\pi_{-k})$. The corresponding maximal revenue rate is defined as

$$J_k^*(P_{-k}|\pi_{-k}) = J_k(\hat{B}_k(P_{-k}|\pi_{-k}), P_{-k}|\pi_{-k}).$$

(12)

Since both $\lambda_k(p_k, p_{-k})$ and $\mu_k(p_k, p_{-k})$ are bounded, the continuous Markov process for each provider $k$ can be uniformized as a discrete-time Markov chain with transition probability matrix (Cassandras and Lafortune 2009; Gross and Miller 1984; Stewart 2009)

$$T_k(P_{|\pi_{-k}}) = \left(t_{i,j}^k(P_{|\pi_{-k}})\right)_{i,j} = I + \frac{Q_k(P_{|\pi_{-k}})}{v_k},$$

(13)

where $I$ is the identity matrix and $v_k \geq \max_{p_k \in \delta_k, p_{-k}} \{\lambda_k(p_k, p_{-k}) + \mu_k(p_k, p_{-k})\}$ is the uniformization parameter. Then, given $P$ and $\pi_{-k}$, the probabilities that provider $k$’s state transits from $n$ to $n+1$, $n-1$ and $n$ at each time point are $t_{n,n+1}^k(P_{|\pi_{-k}})$, $t_{n,n-1}^k(P_{|\pi_{-k}})$ and $t_{n,n}^k(P_{|\pi_{-k}}) = 1 - t_{n,n+1}^k(P_{|\pi_{-k}}) - t_{n,n-1}^k(P_{|\pi_{-k}})$, respectively, where $t_{n,n}^k(P_{|\pi_{-k}}) = \mathbf{E}_{p_k \in P_k} \{\lambda_k(p_k, p_{-k})\}/v_k$, $t_{n,n+1}^k(P_{|\pi_{-k}}) = \mathbf{E}_{p_k \in P_k} \{\mu_k(p_k, p_{-k})\}/v_k$. Naturally, we define $t_{n,n+1}^k(P_{|\pi_{-k}}) = t_{n-1,n}^k(P_{|\pi_{-k}}) = 0$.

Let $M_{k,n}(P_{-k}|\pi_{-k})$ denote provider $k$’s Markov process starting from state $n$ with best response policy $\hat{B}_k(P_{-k}|\pi_{-k})$ and $R_{k,n}(P_{-k}|\pi_{-k})$ represent the expected total revenue over infinite time of the process. Assume that $M_{k,n}(P_{-k}|\pi_{-k})$ remains at state $n$ at the first $m > 0$ time steps and then transits to state $n+1$ at time $m+1$, which happens with probability $t_{n,n}^k(M_{k,n}(\hat{B}_k(P_{-k}|\pi_{-k}), P_{-k}|\pi_{-k})), \vdots, t_{n,n+1}^k(M_{k,n}(\hat{B}_k(P_{-k}|\pi_{-k}), P_{-k}|\pi_{-k})).$ Thereafter, we can expect that the state of $M_{k,n}$
at time $i$ is the same with that of $M_{k,n+1}(P_{-k}|\pi_{-k})$ at time $i - m$ since they are using the same policy $\hat{B}_k(P_{-k}|\pi_{-k})$. The difference between the expected total revenue of $M_{k,n+1}(P_{-k}|\pi_{-k})$, $R_{k,n+1}(P_{-k}|\pi_{-k})$, and that of $M_{k,n}(P_{-k}|\pi_{-k})$ under the above assumption is equal to $m(\hat{J}^*_k(P_{-k}|\pi_{-k}) - n\hat{b}_k(P_{-k}|\pi_{-k}))$, where $\hat{b}_k,n(P_{-k}|\pi_{-k}) \in \hat{B}_k(P_{-k}|\pi_{-k})$ is provider $k$'s best-response price for state $n$. The similar result can be derived if $M_{k,n}(P_{-k}|\pi_{-k})$ transits to state $n - 1$ at time $m + 1$. Then we have that

$$R_{k,n+1}+R_{k,n+1}-R_{k,n} = \sum_{m=1}^{\infty} (t_{k,n+1}^m m[\hat{J}^*_k - n\hat{b}_k]$$

+(m(\hat{J}^*_k - n\hat{b}_k)) \in R_{k,n+1} - R_{k,n+1}, (14)$$

where we omit the terms ($\hat{\beta}_k(P_{-k}|\pi_{-k}), P_{-k}|\pi_{-k}$) and ($P_{-k}|\pi_{-k}$) due to space limit. The above equation leads to that, $\forall k \in K$ and $n \in [N_k]$,

$$\hat{J}^*_k = n\hat{b}_k + t_{k,n+1}(R_{k,n+1}-R_{k,n}) + t_{k,n}(R_{k,n+1}-R_{k,n}). (15)$$

The best response price $\hat{b}_k,n$ can then be efficiently solved by standard dynamic programming algorithms such as policy iteration (Bertsekas et al. 1995), the high-level idea of which is to first (randomly) initialize $R_{k,n}$ for all $n \in [N_k]$ and then repeat the following three steps until $\hat{b}_k,n$ for all $n \in [N_k]$ do not change: 1) compute $\hat{b}_k,n$ for all $n \in [N_k]$ by maximizing the right side of Eq.(15); 2) calculate $\hat{J}^*_k$ based on Eq.(12); 3) update $R_{k,n}$ for all $n \in [N_k]$.

4 Structural Properties

In this section, we study some important structural properties for the AE policy and revenue rate. The first one is the monotonicity of $R_{k,n}(P_{-k}|\pi_{-k})$.

Theorem 2 (Monotonicity of $R_{k,n}(P_{-k}|\pi_{-k})$). For all $k \in K$, $P_{-k} \in \Delta_{-k}$ and $\pi_{-k} \in \pi_{-k}$, it holds that, $\forall n \in [N_k - 1]$, $R_{k,n+1}(P_{-k}|\pi_{-k}) \geq R_{k,n}(P_{-k}|\pi_{-k})$.

Proof. Given $P_{-k}$ and $\pi_{-k}$, and consider two copies of $k$'s system. The first one, we refer to as System A, starts from state $n + 1$, and the second one, System B, starts from state $n$. We let B follow the optimal (best-response) policy A uses the same price as B at any time. Thus the total revenue of B is $R_{k,n}(P_{-k}|\pi_{-k})$. Since A and B set the same price all the time, we can assume that they observe the same arrival and departure sequences. There are two special cases. The first one is that A is at state $N_k$ and B is at $N_k - 1$ and new demand arrives. At next time, A stays at $N_k$ and B transits to $N_k$, and from that time on, A and B will always stay in the same state. The analyze for the case where A is at state 1 and B is at state 0 and demand reduces is similar. Hence, the number of running instances in A is always not less than that in B and hence the total revenue of A is not less than $R_{k,n}(P_{-k}|\pi_{-k})$. If we let A use the optimal (best-response) policy $\hat{B}_k(P_{-k}|\pi_{-k})$, it may gain higher total revenue, i.e., $R_{k,n+1}(P_{-k}|\pi_{-k}) \geq R_{k,n}(P_{-k}|\pi_{-k})$.}

Theorem 2 asserts that the maximal total expected revenue ($R_{k,n}(P_{-k}|\pi_{-k})$) increases with the utilization of the system. The next theorem further proves the concavity of $R_{k,n}(P_{-k}|\pi_{-k})$, i.e., $U_{k,n}(P_{-k}|\pi_{-k}) \geq U_{k,n+1}(P_{-k}|\pi_{-k}) \geq 0$, where $U_{k,n}(P_{-k}|\pi_{-k}) = R_{k,n}(P_{-k}|\pi_{-k}) - R_{k,n-1}(P_{-k}|\pi_{-k})$.

Theorem 3 (Concavity of $R_{k,n}(P_{-k}|\pi_{-k})$). For all $k \in K$, $P_{-k} \in \Delta_{-k}$ and $\pi_{-k} \in \pi_{-k}$, it holds that, $\forall n \in [N_k - 1]$, $U_{k,n}(P_{-k}|\pi_{-k}) \geq U_{k,n+1}(P_{-k}|\pi_{-k})$.

Proof. We use mathematical induction for the proof, which includes two main steps. The first one is to prove that

$$U_{k,1}(P_{-k}|\pi_{-k}) \geq U_{k,2}(P_{-k}|\pi_{-k}).$$

We can reformulate Eq.(15) as

$$\hat{J}^*_k(P_{-k}|\pi_{-k}) = \max_{p_{k,n}} n \pi_{min} + t_{k,n+1}(P_{-k}|\pi_{-k})U_{k,n+1}(P_{-k}|\pi_{-k})$$

$$-t_{k,n-1}(P_{-k}|\pi_{-k})U_{k,n}(P_{-k}|\pi_{-k}). (16)$$

Let $\tilde{b}_{k,n}$ represent $\hat{b}_{k,n}$ for simplicity and define

$$g_k(p_{k,n}, P_{-k}, \pi_{-k}) = E_{p_{k,n}} \lambda(\hat{p}_{k,n}, p_{-k})$$

$$\in V_k.$$ (17)

It follows that

$$\hat{J}^*_k(P_{-k}|\pi_{-k}) = 0 \cdot \hat{p}_{k,0} + E_{p_{-k} \in P_{-k}}(\lambda_{p}(\hat{p}_{k,0}, p_{-k}))$$

$$-U_{k,1}(P_{-k}|\pi_{-k}) = U_{k,1}(P_{-k}|\pi_{-k}) - E_{p_{-k} \in P_{-k}}(\lambda(p_{-k})|p_{-k})$$

$$\in V_k.$$ (17)

$$U_{k,2}(P_{-k}|\pi_{-k}) = E_{p_{-k} \in P_{-k}}(\lambda_{p}(\hat{p}_{k,0}, p_{-k}))$$

$$\in V_k.$$ (17)

$$+ U_{k,1}(P_{-k}|\pi_{-k}) = E_{p_{-k} \in P_{-k}}(\lambda_{p}(p_{-k})|p_{-k})$$

$$\in V_k.$$ (17)

Combining the third and last equations in Eq.(17) leads to:

$$E_{p_{-k} \in P_{-k}}(\lambda_{p}(p_{-k})|p_{-k})$$

$$\in V_k.$$ (17)
\{1, 2, \ldots, N_k - 2\},
\hat{J}_k^*(P_{-k}\mid \pi_{-k}) = \min_{P_{-k}} \left\{ \frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\lambda_k(p_k, n, p_{-k})]}{v_k} \right\}
\cdot U_{k,n+1}(P_{-k}\mid \pi_{-k}) \leq (n+1)\hat{p}_{k,n+1}
\frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\mu_k(p_{-k}, n, p_{-k})]}{v_k} U_{k,n+1}(P_{-k}\mid \pi_{-k})
\leq (n+1)\hat{p}_{k,n+1}
\frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\lambda_k(p_{-k}, n, p_{-k})]}{v_k} U_{k,n+1}(P_{-k}\mid \pi_{-k})
\leq \hat{J}_k^*(P_{-k}\mid \pi_{-k}),
\] i.e., \( \hat{J}_k^*(P_{-k}\mid \pi_{-k}) < \hat{J}_k^*(P_{-k}\mid \pi_{-k}) \), which does not hold. Thus, if \( U_{k,n+1}(P_{-k}\mid \pi_{-k}) \geq U_{k,n+1}(P_{-k}\mid \pi_{-k}) \), then \( \hat{p}_{k,n+1}(P_{-k}\mid \pi_{-k}) \geq \hat{p}_{k,n+1}(P_{-k}\mid \pi_{-k}) \), \forall n \in \{1, 2, \ldots, N_k - 2\}.

Based on Theorems 2 and 3, we can derive the monotonicity property of the best-response policy \( \hat{B}_k(P_{-k}\mid \pi_{-k}) \).

**Theorem 4** (Monotonicity of \( \hat{B}_k(P_{-k}\mid \pi_{-k}) \)). For all \( k \in K \), \( P_{-k} \in \Delta_{-k} \) and \( \pi_{-k} \in \times_{i\in C(k)} [0, 1]^{N_i+1} \), it holds that, \forall n \in \{N_k - 1\}, \( \hat{p}_{k,n}(P_{-k}\mid \pi_{-k}) \leq \hat{p}_{k,n+1}(P_{-k}\mid \pi_{-k}) \).

**Proof.** For ease of representation, we use \( \hat{p}_{k,n} \) to denote \( \hat{p}_{k,n}(P_{-k}\mid \pi_{-k}) \). We learn from \text{(15)} that
\[
\hat{J}_k^*(P_{-k}\mid \pi_{-k}) = \min_{P_{-k}} \left\{ \frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\lambda_k(p_k, n, p_{-k})]}{v_k} \right\}
\cdot U_{k,n+1}(P_{-k}\mid \pi_{-k}) \leq \frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\mu_k(p_{-k}, n, p_{-k})]}{v_k} U_{k,n}(P_{-k}\mid \pi_{-k})
\leq \frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\lambda_k(p_{-k}, n-1, p_{-k})]}{v_k} U_{k,n+1}(P_{-k}\mid \pi_{-k})
\leq \frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\mu_k(p_{-k}, n-1, p_{-k})]}{v_k} U_{k,n}(P_{-k}\mid \pi_{-k}).
\]
Similarly, we have that
\[
(n-1)\hat{p}_{k,n-1} + \frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\lambda_k(p_{-k}, n-1, p_{-k})]}{v_k} U_{k,n}(P_{-k}\mid \pi_{-k})
- \frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\mu_k(p_{-k}, n-1, p_{-k})]}{v_k} U_{k,n-1}(P_{-k}\mid \pi_{-k})
\geq (n-1)\hat{p}_{k,n} + \frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\lambda_k(p_{-k}, n, p_{-k})]}{v_k} U_{k,n}(P_{-k}\mid \pi_{-k})
- \frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\mu_k(p_{-k}, n, p_{-k})]}{v_k} U_{k,n-1}(P_{-k}\mid \pi_{-k}).
\]
These two inequalities imply that
\[
\hat{p}_{k,n} - \hat{p}_{k,n-1} \geq (U_{k,n}(P_{-k}\mid \pi_{-k}) - U_{k,n+1}(P_{-k}\mid \pi_{-k}))
\frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\lambda_k(p_{-k}, n, p_{-k})]}{v_k} U_{k,n}(P_{-k}\mid \pi_{-k})
- \frac{\mathbb{E}_{p_{-k}\in P_{-k}}[\mu_k(p_{-k}, n, p_{-k})]}{v_k} U_{k,n+1}(P_{-k}\mid \pi_{-k})
\]


5.1 Runtime Evaluation

We compute $\hat{B}_k(P_{-k} | \pi_{-k})$ by solving Eq.(15) with policy iteration, which repeatedly maximizes the right side of Eq.(15) for all $n \in [N_k]$ and updates $\hat{B}_k(P_{-k} | \pi_{-k})$. We see that the search space of policies is $\delta_k$, which can be reduced using Theorem 4, as demonstrated in Section 4. To show the benefit of this operation, we evaluate the runtime of the calculation of best responses $(B_k(P_{-k} | \pi_{-k})$ for all $k \in K$) in Algorithm 1 with original search space $(R_o)$ and reduced search space $(R_r)$, respectively. The experimental results are depicted in Table 1, where $N_1 = N_2 = N_3 = 6$ and the competitive ratio is calculated using $|R_o - R_r|/R_o$. We see that the performance improvement increases with the capacity, which is consistent with our expectation since more redundant search operations are avoided for larger $N$ when computing $\hat{B}_k(P_{-k} | \pi_{-k})$. Besides, the growth rate of the ratio decreases with $N$, which is because $\hat{B}_{k,n}(P_{-k} | \pi_{-k})$ is decreasing with $N_k$ (similar observations can be found in Figures 1(a) and 3(b)) and hence the reduced search space of it diminishes with $N$. Overall, we can significantly enhance the efficiency of Algorithm 1 by utilizing Theorem 4.

<table>
<thead>
<tr>
<th>$N$</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_o$ (seconds)</td>
<td>2.19</td>
<td>6.52</td>
<td>8.99</td>
<td>11.36</td>
<td>14.51</td>
<td>17.18</td>
</tr>
<tr>
<td>$R_r$ (seconds)</td>
<td>1.68</td>
<td>4.62</td>
<td>5.94</td>
<td>7.12</td>
<td>9.08</td>
<td>10.27</td>
</tr>
<tr>
<td>Ratio (%)</td>
<td>23.1</td>
<td>29.0</td>
<td>34.2</td>
<td>37.3</td>
<td>39.1</td>
<td>40.2</td>
</tr>
</tbody>
</table>

Table 1: Runtime Comparison

5.2 AEs with Different Capacities

We first study the properties of providers’ AE policies with different capacities and let $l_1 = l_2 = l_3 = 1.4$, $u_1 = u_2 = u_3 = 1$ in this subsection. Providers’ capacities are set as $N_1 = 10$, $N_2 = 15$ and $N_3 = 20$, respectively.

We plot the equilibrium policy $\hat{P}^*$ in Figure 1(a) and depict $U_{k,n}(\hat{P}_{-k} | \pi_{-k}(\hat{P}^*))$ in Figure 1(b). The results are consistent with our theorems. Specifically, Figure 1(b) shows that $U_{k,n}(\hat{P}_{-k} | \pi_{-k}(\hat{P}^*)) \geq 0$, which validates the monotonicity of $R_k,n(\hat{P}_{-k} | \pi_{-k}(\hat{P}^*))$ (Theorem 2); besides, $U_{k,n}(\hat{P}_{-k} | \pi_{-k}(\hat{P}^*))$ is decreasing with respect to $n$, which implies that $R_k,n(\hat{P}_{-k} | \pi_{-k}(\hat{P}^*))$ is concave (Theorem 3); furthermore, $\hat{p}_{k,n}$ increases with $n$, just as Theorem 4 declares. We observe that $\hat{p}_{k,n} > \hat{p}_{k,n}$ if $N_k > N_k$; which is reasonable because the provider with more unused instances in inventory needs to set a lower price to attract more users. The maximal revenue rates for the three providers are 7.667, 11.647 and 15.661, respectively. An interesting finding is that

$$\frac{J_k(\hat{P}_{-k} | \pi_{-k}(\hat{P}^*))}{J_k(\hat{P}_{-k} | \pi_{-k}(\hat{P}^*))} \approx \frac{N_k}{N_k},$$

i.e., $J_k(\hat{P}_{-k} | \pi_{-k}(\hat{P}^*))$ is linear with $N_k$ when providers have the same arrival and departure rate functions. Next, we conduct extensive experiments to investigate how the parameters of the demand functions influence providers’ policies and revenue rates.

5.3 AEs with Different Demand Functions

The demand functions for different providers are probably not the same because the qualities of their services are usually different. The difference is reflected by parameters $l_k$ and $u_k$ in our experiments. In this subsection, we investigate how providers’ pricing policies and revenue rates change with respect to these parameters.

We can verify based on Eq.(15) that $\hat{b}_{k,0}(P_{-k} | \pi_{-k}) = p_{k,0}^\text{min}$. It thus follows that $\hat{p}_{k,0} = p_{k,0}^\text{min}$, $\forall k \in K$. The experiments in this part contain two scenarios ($S_1$ and $S_2$) as shown in Table 2, where $\hat{p}_{k,0}$ is omitted. In $S_1$, we observe that $N_1 = N_2 = N_3 = 6$ for ease of representation. In the first one, we assume providers have the same departure rate functions and different arrival rate functions. We see that a larger $l_k$ implies higher prices and $J_k(\hat{P}_{-k} | \pi_{-k}(\hat{P}^*))$, and if we let two providers have the same parameters, we find them to follow the same policy and obtain the same revenue rate. A larger $l_k$ means that the corresponding provider is more attractive to users than others when they set the same price, i.e., the provider has competitive advantages in the market and hence can use a relative higher pricing to gain more profits. The result for the second scenario where providers have different departure functions is similar.

5.4 Revenue Drop of Ignoring Market Competition

To evaluate the benefits of the proposed $\hat{P}^*$, we compare it with the existing optimal dynamic pricing (Elmaghrawy and Keskinocak 2003; Paschalidis and Tsitsiklis 2000; Xu and Li 2012), which maximizes $\sum_{k=1}^{N_k} \pi_k(P^*_k)n_p k,n$ for each provider $k$ without consideration of others’ strategy.
profile \( P_{-k} \). We use the same parameter settings in Table 2 which contains two scenarios. We compute each provider \( k \)'s revenue rate when he/she uses the noncompetitive strategy that maximizes the revenue rate with arrival function \( \lambda_k(p_k) = l_k(1-p_k)^2 \) and departure function \( \mu_k = u_k p_k^2 \), while others use Algorithm 1 to calculate their strategies. The comparison is shown in Figure 2, where the two bars associated with provider \( k \) represent the maximal revenue rate \( k \) can get when 1) all providers follow the \( \hat{P}^* \) and 2) \( k \) resorts to the noncompetitive strategy, respectively. We see that in both scenarios, providers get the highest revenue rates when they use the \( \hat{P}^* \). The noncompetitive strategy will lead to about 10% drop of revenue as compared with \( P^* \). The results indicate that our proposed strategy outperforms existing pricing policies in the real competitive market.

![Figure 2: Strategy Comparison](image)

5.5 Revenue Improvement

In this subsection, we evaluate the providers’ revenue rate under the AE with experiments, the basic setting of which is the same as the first scenario in Table 2. We will check how a provider’s revenue rate change with his/her arrival rate parameter and capacity. We take provider 3 as an example and use Algorithm 1 to compute each provider’s AE policies with different \( l_3 \) and \( N_3 \). Figure 3(a) shows the revenue rates of the three providers, all of which are increasing with respect to \( l_3 \). This observation is reasonable since a larger \( l_3 \) implies that provider 3 becomes more attractive and hence can set a relative higher price for each state, which in return increases (decreases) the arrival (departure) rates of other providers accordingly based on the properties of the demand function. We find from the figure that the revenue rate is a concave function of \( l_3 \) and providers get the same revenue rate when they have the same parameters. In Figure 3(b), we plot provider 3’s AE polices and state probabilities with different capacities \((N_3 = 6, 8, 10)\). The figure indicates that when the capacity is increased, the provider always prefers to decrease his/her prices in order to improve the utilization of his/her instances.

![Figure 3: Revenue rate improvement](image)

5.6 Evaluate the Tightness of \( \epsilon \)

For each parameter setting, we compute \( \epsilon_k \) for all \( k \in K \) using the “fmincon” function with the interior-point algorithm of Matlab 2015a. In fact, \( \epsilon_k \) can be viewed as the maximal additional revenue rate provider \( k \) can get by unilaterally deviating from the AE \( \hat{P}^*_k \). The results are depicted in Table 3. We observe that in all situations, \( \epsilon_k \) is small compared with \( \hat{J}_k^* (\hat{P}^*_k | \pi_{-k}(\hat{P}^*)) \). The ratios indicate that the highest revenue improvement is only 1.66%, which implies that the benefit of deviating from \( \hat{P}^*_k \) is very limited. Thus, it is reasonable to assume providers to use \( \hat{P}^*_k \) – a more realistic equilibrium strategy that can be computed under both full and partial information assumptions.

![Table 3: Tightness of \( \epsilon \)](image)

6 Conclusion and Discussion

We studied the dynamic pricing optimization problem for the service providers selling reusable products and made three main contributions. First, we proposed a comprehensive model that captures the dynamic and competitive features of the market. Second, we formulated providers’ optimal pricing policies as an AE and developed an efficient algorithm to solve it. Third, we derived many useful properties for the model without any further constraints on demand functions. Our experimental results showed that the policy we computed outperforms existing methods in the literature. Our algorithm can be extended to handle the situation where \( \hat{P}^* \) does not exist. Specifically, we can extend the set of best-response edges of the SPG to \( \xi \)-best responses, which are defined as

\[
\{ P_k | P_k \geq \hat{B}_k (P_{-k} | \pi_{-k}(P)) - \xi, k \in K \}.
\]

Accordingly, we change the terminal condition from Eq.(9) to

\[
\hat{J}_k (P_k, P^*_{-k} | \pi_{-k}(\hat{P}^*)) + \xi \geq \hat{J}_k (P_k, P^*_{-k} | \pi_{-k}(\hat{P}^*)�
\]

Then Algorithm 1 always converges if a proper \( \xi \) is set and the resulting policy is an \( \epsilon + \xi \) NE.
Reference


