RANA, R. and OLIVEIRA, F.S. 2015. Dynamic pricing policies for interdependent perishable products or services using reinforcement learning. *Expert systems with applications* [online], 42(1), pages 426-436. Available from: https://doi.org/10.1016/j.eswa.2014.07.007

Dynamic pricing policies for interdependent perishable products or services using reinforcement learning.

RANA, R. and OLIVEIRA, F.S.

2015

Dynamic Pricing Policies for Interdependent Perishable Products or Services using Reinforcement Learning

Abstract: Many businesses offer multiple products or services that are interdependent, in which the demand for one is often affected by the prices of others. This article considers a revenue management problem of multiple interdependent products, in which price set is dynamically adjusted over a finite sales horizon to maximize expected revenue, given an initial inventory for each product. The main contribution of this article is to use reinforcement learning to model the optimal pricing of perishable interdependent products when demand is stochastic and its functional form unknown. We show that reinforcement learning can be used to price interdependent products. Moreover, we analyze the performance of the Q-learning with eligibility traces algorithm under different conditions. We illustrate our analysis with the pricing of services.

Keywords: Dynamic pricing; Reinforcement learning; Revenue management; Service Management; Simulation.

1 Introduction

The history of the development of expert-systems is a very reach one, throughout the years several important applications have been proposed in which there is an attempt to transfer expertise form humans to computers by using artificial intelligence methods, see Eom (1996) and Liao (2005) for a complete survey in the area: expert systems have been applied in Accounting and Finance, Human resource management, Marketing, Logistics, and Manufacting planning, among other areas. In the context of pricing problems we find: customized pricing in which the producer charges a different price to different consumers (e.g., Lee et al., 2012); automobile pricing using artificial neural networks (Iseri and Karlik, 2009); pricing and promotion strategies for online shoping (Chan et

al., 2011); smart meetering (e.g., Chakraborty et al., 2014); and pricing of mobile phones (Sohn et al., 2009), among others.

In this article we address the issue of dynamic pricing interdependent products and services, which can be defined as those whose demand is affected by the prices of other products and services. The dynamic pricing of interdependent and perishable products or services requires a strategy that considers these demand interdependencies. Indeed, the generic problem of pricing perishable interdependent products or services arises in a variety of industries, including fashion, or seasonal retail, and the travel and leisure industries. For example, in the retail industry it may take as long as six to eight months to produce an item which would typically be expected to be sold in as little as nine weeks (Gallego and Van Ryzin, 1994). In such a case, reordering stock is not possible and old stock must be cleared before the arrival of new stock. Many retail products influence demand for other products or services. For example, changes in the price of a pair of jeans might affect the demand for a matching belt or other related brand preferences. Other examples include flights to the same destination at different times of the day or week, the delivery of services at different times, and various types of rooms in a hotel. Ignoring the effects of demand substitution on inventory and pricing decisions can have significant implications for profit (Bitran et al., 2004). This interdependency is especially important as the need to understand purchasing behavior of customers becomes increasingly complex as the number of variables increases with interdependent products.

For simplicity, most studies in dynamic pricing of interdependent services or products assume that the functional relationship between demand and price is known to the decision maker. For example, Oliveira (2008) uses Lemke's algorithm to analyze the dynamic pricing of interdependent products both within a week and for the management of the products' life cycle. This author's assumption was that, indeed, the firm knows the demand functions both in the short-term and in the long-term. Also, Besbes and Zeevi (2009) show that for a single product dynamic pricing problem, the use of parametric approaches in nonparametric environments can result in significant revenue loss. The assumption of full information, especially with multiple interdependent products, makes the problem more tractable. While allowing for the fast solution of complex problems (Oliveira, 2008), this endows the decision maker with knowledge that he or she does not possess in practice. These assumptions regard not only the estimates for demand and cost parameters but also the functional forms of the demand and cost functions. These functional forms are very

difficult to estimate and in most cases are unknown.

The pricing of services and products is usually influenced by many factors such as competitors prices of substitutable services and stochastic demand, which makes it a complex large scale stochastic problem. For this reason a simplified model is usually analyzed, due to computational tractability, as the most complex models are too difficult for managers to implement in real time, because of the large number of calculations involved. A method that allows the analysis of a complex problem, such as the pricing of interdependent products, requires the ability to both implicitly learn demand behavior and to optimize the pricing policies of the different products. Reinforcement learning meets these requirements (e.g., Sutton and Barto, 1998; Kaelbling et al., 1996; Mabu et al., 2012; Peteiro-Barral et al., 2013; dos Santos et al., 2014; Oliveira, 2014) as the optimal policy is learn implicitly (without requirement the knowledge of the actual demand function) and without requirement knowledge of the transition probabilities between states; moreover, as reinforcement learning is based on using Monte-Carlo simulation being able to handle very large problems.

Reinforcement learning offers the advantage of formulation of a mathematical model based on multiple variables without any pre-definition of non-linear structure of the model, (Jiang and Sheng, 2009, Dorca et al, 2013). Applications of reinforcement learning in the context of expert systems include, among others, goal-regulation in manufacturing systems (Shin et al, 2012), real time rescheduling (Palombarini and Martinez, 2012), inventory control in supply chain management (Kwon et al., 2008; Jiang and Sheng, 2009), and real-time dynamic packaging for e-commerce (Cheng, 2009). Our research similarly uses the advantages of using a model-free approach offered by reinforcement learning algorithm but is applied in a different domain i.e, the dynamic pricing of multiple interdependent products. The major contribution of this article lies in the use of the Q-learning with eligibility traces algorithm to model the dynamic pricing of interdependent services. The use of this algorithm allows the joint learning of the pricing strategies for different services without explicitly modeling consumer behavior. Using a model-free environment (whereby the transition probabilities between states follow an unknown distribution) enables many influencing factors to be included implicitly in the pricing decisions.

The remainder of this paper is therefore structured as follows. First, we review relevant literature and present the contribution of this article. Second, we discuss how the model is formulated and analyze the dynamic pricing model with interdependent products. Third, we evaluate the

performance of the interdependent learning algorithm, using simulation experiments, and provide an understanding of the dynamics of the theorems proved in the article. Finally, we summarize our conclusions.

2 Relevant Literature

The two main areas of research that are most relevant to this study are dynamic pricing and reinforcement learning. Dynamic pricing of perishable assets has been researched extensively see, for example, Gallego and van Rzyin (1994), Mcgill and van Ryzin (1999), Anjos et al. (2004, 2005), Currie et al. (2008) and Zhao and Zheng (2000), who each address a single product problem. Reviews of these articles can be found in Elmaghraby and Keskinocak (2003), Bitran and Caldentey (2003) and Talluri and van Ryzin (2005).

There is limited literature on the dynamic pricing of interdependent products or services. Gallego and van Ryzin (1997) consider dynamic pricing problems where the demand for each product depends on a vector of prices of all the products. They assume that demand is Markovian for the current price and that the relationship between all prices and arrival rates is known. Bitran et al. (2004) combine the multinomial logit model with a utility maximization function to describe the demand for substitutable products. These authors use heuristic algorithms to approximate an optimal solution. Maglaras and Meissner (2006) show that when customers choose between multiple products, the dynamic pricing problem can be reduced to an equivalent one-dimensional problem. They propose several heuristics to solve the optimization problem. Cooper et al. (2006) show that neglecting substitution across products can lead to a downward spiral effect, in which the performance of the capacity allocation policy worsens systematically as the forecasting-optimization process continues. Zhang and Cooper (2006) develop a Markov decision process formulation of dynamic pricing for multiple substitutable flights between the same origin and destination, taking into account customer choice among flights. Netessine et al. (2006) consider cross-selling by offering customers a choice between their requested product and a package containing multiple products which include the requested one. They recognize the complexity of this problem and demonstrate that, in a setting where the number of products is three or more, the choice of the best packaging complements is non-trivial. Oliveira (2008) uses Lemke's algorithm to analyze dynamic pricing issues in the daily and life-cycle dynamic pricing of services. Asdemir

et al. (2009) investigate optimal dynamic pricing of multiple home delivery options using dynamic programming. Their analysis shows that substitution effects are significant on an optimal pricing policy and on the resulting revenue gained. The joint dynamic pricing of multiple perishable products under a consumer choice model was investigated by Akcay et al. (2010), who formulate the problem as a stochastic dynamic program where consumer behavior depends on the nature of product differentiation. Kim and Bell (2011) study the impact of price-driven substitution on a firms' pricing and production capacity decisions for a single period during which the firm sells to multiple segments.

The papers listed above have assumed knowledge of model parameters. In this article we develop methods for learning the demand response functions over time. However, it could be argued that the real-world demand model is more complex, given that parameters are unknown and, therefore, modeling errors may arise through assumptions that are made for the purpose of analytical tractability (Lim and Shanthikumar, 2007). Estimating the demand for services is difficult, especially when faced with increasing numbers of interdependent services, and dynamic pricing models in the aforementioned literature have therefore had to make assumptions regarding customer behavior. The possibility of substitution across products and services has a significant impact on both on the probability distribution of demand and the total revenue gained.

Given the complexity of dynamic programming, instead, when modelling real-world problems a good approach is to use of heuristics (e.g., Burkart et al., 2012; Sen, 2013) or reinforcement-learning. In this article we are going to explore the use of reinforcement learning as this is an ideal method for solving the pricing problem in situations when both the probability distributions of demand and the expected revenue gain for taking a pricing action are unknown. Reinforcement learning does not require any assumptions regarding market demand such as customer arrival rates, customer reservation price or the probabilities of demand substitution, but rather one learns how customers act at each time-step before the service expires, and subsequently this information is used to solve the maximization problem. The revenue management literature using reinforcement learning is limited (e.g., Gosavi et al., 2002; Raju et al., 2006). Rana and Oliveira (2013) use reinforcement learning, Q-learning and Q-learning with eligibility traces algorithms, to learn and optimize pricing strategies for a single product in a non-stationary finite selling horizon. They present an approach that avoids optimization errors caused by the underlying model and compare a model-free approach with a parameterized structure approach. The literature concerning reinforcement learning and

multiple product pricing is even sparser. For example, Cheng (2007) investigates a Q-learning approach to determining dynamic pricing for multiple products in an e-retailing setting. These papers do not consider the dynamic nature of demand, how the demand between the products interacts and the effects on the revenue gained.

The majority of articles (e.g., Maglaras and Meissner, 2006; Cheng, 2007) do not explicitly model the dependencies between demand and history of pricing since this is a complex task (especially with interdependent products or services); it requires estimations of all inter-related model parameters and this in turn requires a large amount of data. Also, this may not be computationally tractable as the number of interdependent products or services increases.

3 Modeling Dynamic Pricing using Reinforcement Learning

Reinforcement learning originated in the areas of cybernetics, psychology, neuroscience, and computer science and has since attracted increasing interest in artificial intelligence and machine learning (e.g., Sutton and Barto 1998; Gosavi 2009; dos Santos et al., 2014; Oliveira, 2014). Reinforcement learning is an approach to sequential decision making in an unknown environment that is based upon learning from past experience. Reinforcement learning algorithms apply directly to the agent's experience, changing the policy in real time. The first advantage of using reinforcement learning is that it does not require a pre-specified model of the environment on which to base the action selections. Instead, the relationship between states, actions and rewards is learned through dynamic interaction with the environment. The second advantage is that it is adaptive, in the sense that reinforcement learning is capable of responding to a dynamically changing environment through on going learning and adaption.

The revenue management problem of interdependent products in a finite selling horizon is considered in this article. For example, consider a dynamic pricing problem for multiple substitutable flights between the same origin and destination, in which the airline's objective is to maximize the total expected revenue from customer bookings over the finite selling horizon by setting the prices of the flights. Customers choose among the flights or decide not to purchase based upon their own preferences and the prices of all the flights offered. This problem is particularly relevant to low-cost airlines which have multiple-flights scheduled for the same day between each pair and

they sell many tickets on the Internet, hence making price comparisons easily accessible. Another example is that of a delivery company which operates from Monday to Friday with a fixed number of drivers. The customers choose from multiple delivery times. The company can use dynamic pricing as a mechanism to control uncertainty created by consumers choice of delivery time, with objective to maximise the expected revenue of the entire week.

In this article we use the following concepts.

A) The selling horizon is the period of time during which the product or the service is sold. For example an airline company may start selling tickets from 6 months before the flight departs. The last selling period may be 48 hours before departure, if there are still seats available on that flight. This entire period of time is the selling horizon. The selling horizon is split into discrete times at which prices for the services are updated. The selling horizon is separated into m discrete decision times. Let $t = 1, 2, .., m$ denote the index of decision times. m is the last time a price can be changed and at the end of this period there is no salvage value for the unsold services.

B) The state set $\mathbf{X}$ is a set of all possible capacity states. Capacity state is the amount of inventory available for each product or service. For the multiple substitutable flights example, the capacity state would reflect the number of seats available on each flight, where each flight would be categorised as a different service. A dynamic pricing decision is made at the beginning of every decision time. The state of current capacity is the reference factor for future pricing decisions, which consists of the current decision time index and capacity level of each interdependent products. Let $X_t = (x_t(1), x_t(2), , x_t(n))$ denote the current state, the capacity level for all n products at time t. Where, $1, .., n$ denotes the set of products or services.

C) The pricing action set $A(X_t)$ is a set of actions available in state X_t. At decision time t, a set of prices $A_t = (a_t(1), a_t(2), .., a_t(n))$ for all products is set for this period, where, $a_t(1)$ is the price for product 1 at time t. e.g., each flight will have a price for its seat, and the set of prices for all the multiple flights A_t is called a pricing action.

D) The state transition probabilities specify how the actions affect the transition from one state to the next. Let $P_t(X_{t+1}|X_t, A_t)$ be the probability that the state is X_{t+1} after it was currently at state X_t and A_t is performed. For the multiple flights example, it is the probability of having X_{t+1} amount of seats remaining for each flight at time $t + 1$, when the seats remaining for each flight is X_t and the pricing action A_t is selected, at time t. The transition probabilities are only required here for explanation and proof purposes, but the learning algorithm does not require them

to derive a pricing policy.

E) The policy is a function $\pi : X_t \rightarrow A_t$ which specifies the prices that should be set for all the products or services given the remaining capacities at time t.

F) The revenue function computes, for every state X_t and pricing action A_t, the immediate revenue gained, $r(X_t, A_t, X_{t+1}) = \sum_{j=1}^{n} r(x_t(j), a_t(j), x_{t+1}(j))$.

The objective function to be maximized is the finite total expected revenue,

$$V_1^{\pi}(X_1) = E^{\pi}[r(X_1, A_1) + r(X_2, A_2) + ... + r(X_m, A_m)|X_1, \pi] \tag{1}$$

for $X \in \mathbf{X}$ and E_{π} is the expected value given the policy π.

The Q-learning with eligibility traces Q(λ) algorithm has been proposed to approximately solve large scale Markov Decision Process (MDP) problems. In an MDP framework, $V_t^{\pi}(X_t)$ denotes the expected total reward when starting at state X_t and following a policy π; $Q_t^{\pi}(X_t, A_t)$ denotes the discounted expected total reward when starting at state X, taking action A at time t and following policy π, i.e., Q_t^{π} is the (X_t, A_t) value function for policy π (i.e., $A_t = \pi(x_t)$ where $\pi(s)$ denotes the action chosen in state s when policy π is pursued). Eq. (2) represents the relationship of $Q_t^{\pi}(X_t, A_t)$ and V_t^{π},

$$Q_t^{\pi}(X_t, A_t) = \sum_{X_t \in X} P_t(X_{t+1}|A_t, X_t)[r(X_t, A_t, X_{t+1}) + \eta V_{t+1}^{\pi}(X_{t+1})] \tag{2}$$

where η is the discount factor, $0 < \eta < 1$. In terms of the Bellman optimality function, Eq. (3) holds for arbitrary $X_t \in X$, where $Q_t^*(X_t, A_t)$ is the optimal value function for each state-action pair. For all $t = 1, 2, ...m$

$$Q_t^*(X_t, A_t) = \sum_{X_t \in X} P_t(X_{t+1}|A_t, X_t)[r(X_t, A_t, X_{t+1}) + \eta \max_{A_{t+1}} Q_{t+1}^*(X_{t+1}, A_{t+1})] \tag{3}$$

In the Q(λ) paradigm, the decision maker interacts with the environment by executing a set of actions. The environment is then modified and the agent receives a new state and a reward signal at each decision time. Over the course of the learning process, the Q-value of every state-action pair, $Q_t(X_t, A_t)$, is stored and updated. Let k denote the episode (selling horizon) and let each selling horizon be split into m decision times. The episode refers to multiple instances of the dynamic pricing problem in consecutive time horizons and the transition probabilities are the same for different episodes (this makes them stationary across different episodes) but non-stationary within

each episode. The algorithms consist of updating $Q_{t,k}$, the Q-values at every selling horizon k, for time t, which is a representation the estimation of Q_t^*, the optimal Q-values, from the current observed transitions and reward $< X_{t,k},\ A_{t,k},\ X_{t+1,k},\ r_{t,k} >$, where $X_{t,k},\ A_{t,k}, r_{t,k}$ is the remaining capacity, price action, current observed reward at time t, in episode k, respectively and $X_{t+1,k}$ is the new remaining capacity at time $t+1$. Note that $r_{t,k}$ would depend on $X_{t,k},\ A_{t,k}$ and $X_{t+1,k}$ i.e., $r_{t,k} \equiv r(X_{t,k}, A_{t,k}, X_{t+1,k})$.

The Q-values estimates of all state-action pairs are updated in proportion to their eligibility. The use of eligibility traces helps a reinforcement learning based system to solve the temporal-credit assignment problem, i.e., to calculate how to punish or reward a state-action choice, when it might have far-reaching effects. Additionally, reinforcement learning with eligibility traces has the important ability of learning in non-stationary selling horizons; refer to Rana and Oliveira (2013), for a more detailed discussion. The eligibility traces function, at episode k, at time t, is denoted by $e_{t,k}$. On observing $< X_{t,k},\ A_{t,k},\ X_{t+1,k},\ r_{t,k} >$ the following updates are performed to the eligibilities trace:

$e_{t,k}(X_{t,k}, A_{t,k}) = 1,$

$\forall\ i < t\ e_{t,k}(X_i, A_i) = \lambda e_{t-1,k}(X_i, A_i)$ if $Q_{t,k}(X_{t,k}, A_{t,k}) \ = \max_{A_t} Q_{t,k}(X_{t,k}, A_t),$

otherwise $e_{t,k}(X_i, A_i) = 0.$

The eligibility trace decays for all state-action pairs at a rate λ , except for the last state-action visited where the eligibility trace is incremented by one unit. The traces can be updated in two ways. If a greedy selection is made, then all the traces of the state-action pairs visited in the episode decay at a parameter λ. The decay parameter determines how different state-action pairs are assigned a certain prediction error. If an exploration action was taken then the eligibility traces are set to zero. At episode k, for each decision time-step t, we look at the current error,

$\delta_{t,k} = r_{t,k} + \eta \max_{A_{t+1}} Q_{t,k}(X_{t+1,k}, A_{t+1}) - Q_{t,k}(X_{t,k}, A_{t,k})$, and assign it backward to each prior state-action pair visited in episode k according to their eligibility trace. The objective is to learn the optimal pricing policy. The Q(λ) algorithm iteratively computes the optimal value function: for each state X at every time t, the optimal value $Q_t^*(X_t, A_t)$ of each action A_t is estimated on the basis of simulated transitions. When all these values have been correctly estimated, the optimal policy can be derived through:

$$\forall t \leq m,\ \forall X \in \mathbf{X},\ \pi^*(X_t) = \max_{A_t} Q_t^*(X_t, A_t). \tag{4}$$

Before discussing the $Q(\lambda)$ algorithm in procedural form (presented in Table 1), two important elements must be introduced: the exploration rate and the learning rate. The absence of perfect prior information concerning the demand model introduces a new component into the dynamic optimization problem; the trade-off between exploration (attempting non-optimal decisions in order to improve the current policy) and exploitation (choosing the best policy so far in order to maximize the expected profit). The longer one spends learning the demand, the less time is spent exploiting prices to increase revenue.

The objective of the algorithm is to obtain an accurate estimate of the optimal policy based on observations during the exploration phase while, at the same time, keeping the exploration rate small in order to limit revenue loss over this learning phase. The exploration rate (ϵ) chosen is a ϵ-greedy policy at a rate $1/k$ so that learning progresses as the exploration rate decreases. The result of this assumption is that, as the decision-maker gains more knowledge, sub-optimal prices are explored less often. The learning rate is set in a similar manner and also decreases with time. The learning rate for each state-action pair is denoted by $\alpha_k(X_t, A_t)$ and equals $1/n_k(X_t, A_t)$, where $n_k(X_t, A_t)$is the number of times the state-action pair (X_t, A_t) was visited by the process $(X_{t,k}, A_{t,k})$ before time k, plus one.

Table 1: Q(λ) An online policy TD dynamic pricing algorithm for interdependent products in finite selling horizon

Initialize $k = 1$ and $\forall\ t\ Q_t^1(X_t^1, A_t^1)$;	
Repeat for each episode $k \to \infty$	
Step 1: All traces are set to zero at the beginning of the sales horizon	
	$\forall\ t,\ X_t,\ A_t,\ e_{t,k}(X_t, A_t) = 0,$;
Step 2:	We initialize the initial capacities and price set $X_{t,k},\ A_{t,k}$
Step 3:	Repeat $t = 1$ to m (for each decision step in the selling horizon)
Step 3.1:	Take price $A_{t,k}$ observe the reward $r_{t,k}$ and remaining capacities $X_{t+1,k}$
Step 3.2:	We choose a price $A_{t+1,k}$ for state $X_{t+1,k}$ using using policy π $(\epsilon = 1/k)$
	Greedy price $A_{t+1}^* \leftarrow argmax_{A_{t+1}} Q_{t+1,k}(X_{t+1,k}, A_{t+1})$
Step 3.3:	TD error is $\delta_{t,k} \leftarrow r_{t,k} + \eta Q_{t+1,k}(X_{t+1}^k, A_{t+1}^*) - Q_{t,k}(X_{t,k}, A_{t,k})$
Step 4.:	Update the $(X_{t,k}, A_{t,k})$ pair's trace $e_{t,k}(X_{t,k}, A_{t,k}) \leftarrow 1$
Step 4.1:	$\forall\ i \leq t \in T, \forall\ X_i,\ A_i$ Update all Q-values according to their eligibility traces
Step 4.2:	$Q_{t,k+1}(X_i, A_i) \leftarrow Q_{i,k}(X_i, A_i) + \alpha(X_i^k, A_{i,k})\delta_{t,k} e_{t,k}(X_i, A_i)$
Step 4.3:	If $A_{t+1,k} = A_{t+1}^*$, then $e_{t+1,k}(X_i, A_i) \leftarrow \lambda e_{t,k}(X_i, A_i)$
	else $e_{t+1}^k(X_i, A_i) \leftarrow 0$
Step 5:	$X_{t,k} \leftarrow X_{t+1,k},\ A_{t,k} \leftarrow A_{t+1,k},\ k \leftarrow k + 1$

The algorithm in Table 1 starts by initializing the Q-values. We repeat the following steps for

each episode k. All the eligibility traces for all state-action pairs are set to zero. We choose the starting state $X_{t,k}$, a vector of the total number of inventory and $A_{t,k}$ a vector of prices for each product. After taking the pricing action we observe the next state $X_{t+1,k}$ and immediate revenue gained $r_{t,k}$. We choose the pricing action $A_{t+1,k}$ in the next state using the (non-greedy) policy π. The error (δ) is calculated and the eligibility trace $e_{t,k}(X_t, A_t)$ is updated to 1. Next, all the Q-values of the state, action pairs are updated using their eligibility traces. All eligibility traces are updated: if the pricing $A_{t+1,k}$ is a greedy (optimal) action then all traces are multiplied by a parameter λ. If $A_{t+1,k}$ is an exploration action then all traces are set to zero. Then the next state $X_{t+1,k}$ at decision time-step $t+1$ and action $A_{t+1,k}$ becomes the new $X_{t,k}$ and $A_{t,k}$. The process is repeated until the last decision time m, for each episode. The Q-values converge with probability one as long as all state-action pairs are visited an infinite number of times (Rana and Oliveira, 2013).

We proceed with the analysis of dynamic pricing policies for interdependent products, to address model misspecification risk and simplified assumptions of demand, especially on the pricing policy, of pricing interdependent products individually or independent of each other.

4 Analysis of the dynamic pricing model with interdependent products

This section considers how different products (services) and their demand patterns determine the learning dynamics. The analysis first investigates how the Q-values are estimated for interdependent products when their pricing policies have been learnt independently of each other. Consider a set of interdependent products or services, all sold by the same decision maker, but whose pricing policies are derived/learnt independently of each other. Let $A_{(t,-j)} = [a_t(1), .., a_t(j-1), a_t(j+1), .., a_t(n)]$ be the vector of prices of all interdependent products, excluding product j, and let $w_t[x_t(j), a_t(j), A_{(t,-j)}]$ be a weight assigned to state $x_t(j)$, price $a_t(j)$ and prices of interdependent products $A_{(t,-j)}$ tuple at time t. The weight $w_t[x_t(j), a_t(j), A_{(t,-j)}]$ is equal to the number of times the price $A_{(t,-j)}$ is chosen at a state-action pair $(x_t(j), a_t(j))$ divided by the total number of times $(x_t(j), a_t(j))$ is visited. Since the decision maker prices the interdependent products independently, the price of other products is not considered when selecting the optimal policy for product j. However, the price of interdependent products will affect their demand and therefore

the price acts as a stochastic element of demand. Thus the Q-value of a state-action pair will be the weighted average of the Q-values of the state, action and the prices of the interdependent products ($Q^*_{t+1}(x_{t+1}(j), a_{t+1}(j))$).

Theorem 1.1. *The Q-values for any product or service j are equal to the weighted sum of the Q-values of its state x_t, action a_t and prices of the interdependent products A_{t-j}, as described in equation (5):*

$$\begin{aligned} Q^*_t(x_t(j), a_t(j)) &= \sum_{A_{(t,-j)}} w_t[x_t(j), a_t(j), A_{(t,-j)}] Q^*_t(x_t(j), a_t(j), A_{(t,-j)}) \\ &= \sum_{A_{(t,-j)}} w_t[x_t(j), a_t(j), A_{(t,-j)}][r(x_t(j), a_t(j), A_{(t,-j)}) + \\ &\quad p_t(x_{t+1}(j) \mid (x_t(j), a_t(j), A_{(t,-j)}))\eta \max_{a_{t+1}} Q^*_{t+1}(x_{t+1}(j), a_{t+1}(j))]. \end{aligned} \tag{5}$$

[Proof in Appendix]

Deriving a pricing policy for interdependent products individually leads to the prices of the independent products increasing the stochastic element of demand. More specifically, the Q-values calculated using the Q(λ) algorithm are a weighted sum of the prices of the interdependent products. Knowledge of the (part) cause of the stochastic element of demand, especially as the decision maker can control it, can help to price strategically.

The next theorem highlights the advantages of aggregation information of products which follow the same demand to arrive at an optimal policy. Let X_t be the capacity remaining to sell for the first set of products 1 to n; X'_t be the capacity remaining to sell for a different, second set of products from 1 to n; A_t be the prices for the first set of products; A'_t be the prices for the second set of products.

Theorem 1.2. *If we have two sets of n products or services each following the same demand patterns then $r(X_t, A_t) = r(X'_t, A'_t)$ for all $X_t = X'_t$, $A_t = A'_t$ and $\pi^*(X_t, A_t) = \pi^*(X'_t, A'_t)$ and the dynamic policy used for the two sets of products is the same. Aggregating the observations of both products accelerates the learning process of the policy.* [Proof in Appendix]

Theorem 1.2 shows that the more information the decision maker has regarding the demand of a set of interdependent products, the faster the algorithm will converge to the optimal policy.

Next, we discuss how oversimplifying the model changes the learning dynamics.

Theorem 1.3. *If we have two sets of n products or services following different demand patterns* $r(X_t, A_t) \neq r(X'_t, A'_t)$ *for all* $X_t = X'_t$, $A_t = A'_t$ *then aggregating the observations of each would produce a less profitable pricing policy than if the policies were to be derived separately.* [Proof in Appendix]

Theorem 1.3 illustrates how oversimplifying the model can change the dynamics of how the pricing policy is developed. The revenue gain is likely to be significantly lower if the pricing policy uses information of interdependent products that do not follow the same demand pattern. Theorem 1.4 highlights the importance of using interdependent learning.

Theorem 1.4. *Let there be n interdependent perishable products. The total expected revenue for jointly optimising pricing actions at each state, using interdependent learning, is greater than the sum of the individual total expected revenue of the optimal pricing actions in these states:*

$$\max_{A_t} Q^*_t(X_t, A_t) \geq \sum_{j=1}^{n} \max_{a_t(j)} Q^*_t(x_t(j), a_t(j)). \tag{6}$$

[Proof in Appendix]

Theorem 1.4 does not only show that pricing interdependent products independently of each other generates less revenue, but also that simplified assumptions or misspecified models of demand can have significant effects on the revenue gained. This theorem shows the risks of model misspecification, which changes the dynamics of how the pricing policy is learnt. Under an individual learning policy the products may be competing with others, whilst when interdependent products are priced jointly, the effects on demand are taken into account in order to maximise revenue.

Finally, an important issue when using learning models is their tractability in real-world situations. The speed of convergence to the optimal pricing policy as the number of interdependent products increases is analysed in Theorem 1.5.

Theorem 1.5. *Let* u, v *denote the number of interdependent products, if* $u > v$ *then the pricing policy of v interdependent products will converge to its optimal policy exponentially faster than that of u interdependent products.* [Proof in Appendix]

5 Numerical Results and Qualitative Insights

To evaluate the performance of the interdependent learning algorithm, and to understand the dynamics of the theorems presented in the previous section, two examples are considered. In the first example, customers have the choice of two options (Peak or Off-peak) for when to receive a service and in the second example interdependent pricing of five different services with different demand patterns will be offered over the course of a week (weekly pricing).

5.1 Example 1: Peak Service

Consider a business which offers a particular service. It is assumed that on a given day there are a fixed number of workers deployed to meet the overall demand for the service, but that demand varies during the day. There are periods of excess demand, i.e. during Peak hours, such as early morning, lunch time and evening time, where there are insufficient workers to meet all service requests, and periods of low demand, i.e. during Off-peak hours (mid-morning and after lunch), where there are more workers than strictly required. Now, consider the problem where customers have two options of when they receive the service: during Peak hours and Off-Peak hours; it will be assumed that there are equal numbers of time slots in both periods. The objective is to maximise the total revenue over a finite horizon. The characteristics of customer behaviour (assumed in the simulation but not known by the decision maker) are as follows:

1. Customer Arrival Pattern: a Poisson distribution with discrete decision time for the mean arrival $\mu(t)$. The customer arrival rate is drawn randomly from a uniform distribution [60, 100].

2. Customer Reservation Price (i.e., how much the customer is willing to pay): increases exponentially as the decision expiry time approaches, $e^{-\beta a \zeta(t)}$, where a is the price of the service and β is the sensitivity parameter of the customers to the price. As function $\zeta(t)$ increases with time, the customers are willing to pay more for a service that is performed sooner rather than later.

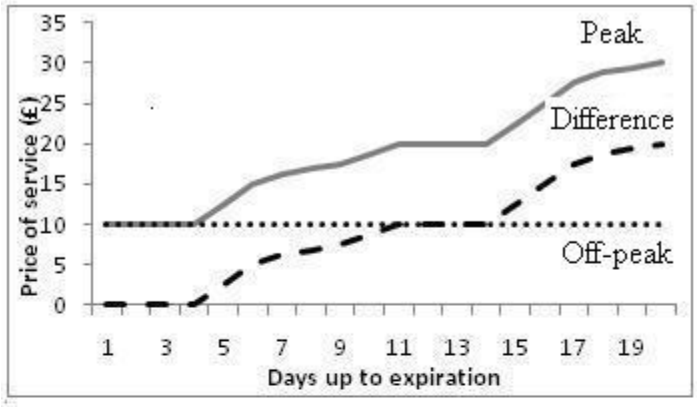
3. Customer Preference: most customers prefer Peak to the Off-peak option; if the price of the service is the same for both Peak and Off-peak times, the probability of a customer purchasing the Peak option is higher.

This model is simulated and studied by considering a discrete price set [10, 35] and a discrete stock set [0,100]. It is assumed that both of the service options expire at the same time. Moreover, a time-horizon of 20 days with daily price change is considered. The first decision time is day 1 and the last time the price can be changed is at the start of day 20, i.e., just before the service expires. The optimal policy is derived by averaging 1000 simulation runs.

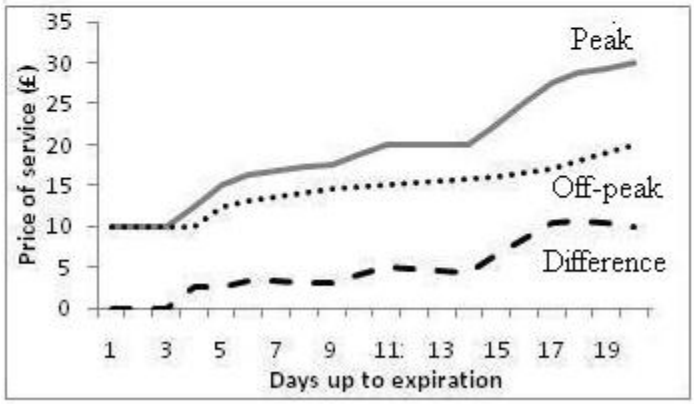
To provide qualitative insight into the optimal policies derived from the independent and interdependent learning algorithms, three experiments have been designed. The first experiment compares two different approaches to determining the optimal pricing policy, i.e., individual learning and interdependent learning. The second experiment compares optimal prices over time, derived from individual learning, when there are different numbers of slots available at Peak times. The third experiment compares optimal prices over time, derived from interdependent learning, when there are different numbers of slots available at Peak times.

In the first experiment, it is assumed that there are 100 time slots in each of the Peak and Off-peak periods. Figure 1 shows the optimal pricing policy for both the individual learning (Figure 1a) and interdependent learning (Figure 1b) approaches. It can be seen that the policies differ over the 20-day time-course and that they differ from each other. It can be observed that the optimal price for Peak is always higher than Off-peak; this is expected as, by definition, Peak is the more popular option and price is demand-driven. The price for both the services increases as time gets closer to expiration; this is due to customers being willing to pay more to receive a service sooner rather than later. In the individual learning scenario, it is interesting to note the Off-peak price remains constant at £10, whereas in the interdependent learning scenario the Off-peak price increases as well as the Peak price. The optimal prices for Peak and Off-peak are closer when the interdependent learning algorithm is used as, by increasing the Off-peak price, demand increases for the more expensive and profitable Peak service. This insight is very specific to this example, and aims to illustrate how interdependent learning leads to different (better) pricing policies. The average total expected revenue gained using individual learning is £2670 and £1657 for Peak and Off-peak, respectively. The average total expected revenue gained using interdependent learning, maximising revenue jointly is £4699. Thus, the interdependent learning algorithm generates 8.6% (£372) more revenue.

The second experiment compares optimal prices derived from individual learning when there are different available slots at Peak times. In the first scenario, the number of slots available in



a. Individual Learning

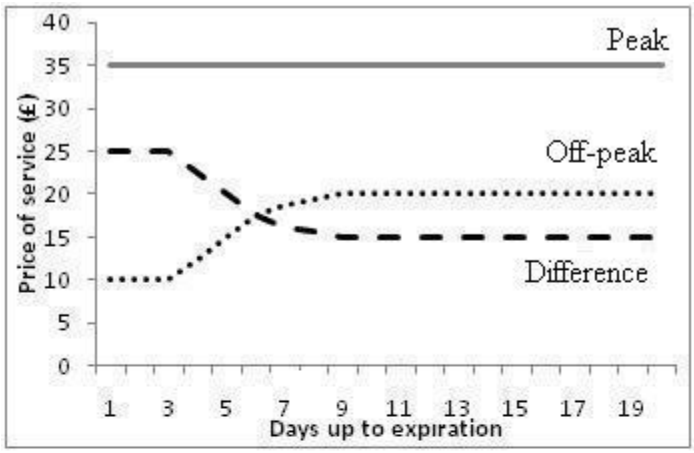


b. Interdependent Learning

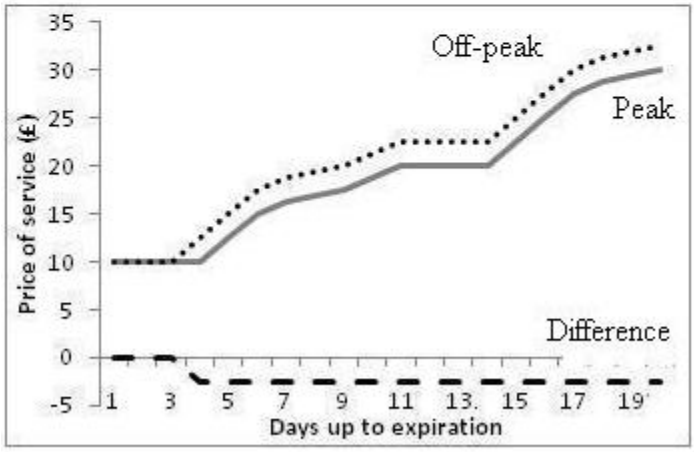
Figure 1: Prices over time with 100 available slots for Peak and Off-peak

the Peak period is 20 and the number available during Off-peak is 40; in the second scenario, the number of slots available during Off-peak hours remains the same but 80 slots are made available during the Peak period. Figure 2 illustrates the optimal prices over time using individual learning for Peak and Off-peak separately. One can observe how the optimal prices of the Peak and Off-peak services changes when the number of available slots for the more popular Peak service is increased. When the availability of Peak slots is low, one observes (Figure 2a) that the price for the Peak service is much higher than that of the Off-peak price; Peak service is priced at £35 across all 20 days. In contrast, Figure 2b shows that when the number of available slots for the Peak service is increased to 80, the price of the Off-peak service is actually higher than Peak service, and remains higher as time progresses. The results suggest that the difference in optimal policy is caused by an interesting substitution effect: the popular Peak service will be filled later in the selling horizon

whereas the less popular Off-peak service will be filled in earlier in the selling horizon. The effect is that the popular Peak service becomes less important as the time in the selling horizon decreases. The total expected revenue gained by the Off-peak service can be greater than the total expected revenue gained by the Peak service.



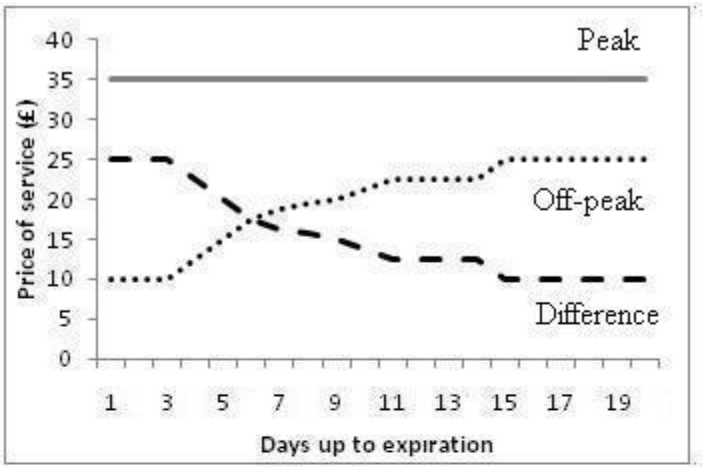
a. x = (20,40)



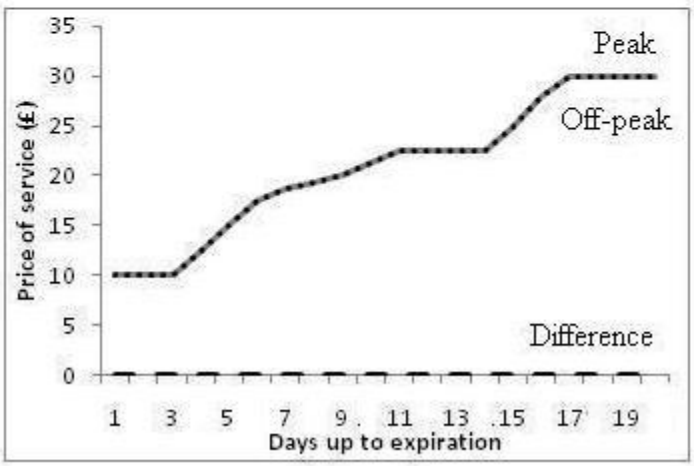
b. x = (80,40)

Figure 2: Prices over time for the different number of available slots using individual learning

The third experiment, designed to investigate Peak service, is similar in design to the previous experiment but determines the optimal policy through interdependent learning. The same two scenarios are examined, i.e. there are 40 Off-peak slots in both scenarios whilst the Peak slots increase from 20 to 80. Figure 3 shows the optimal prices over time for two scenarios. In the scenario where Peak availability is low (Figure 3a), the price of the Peak service remains constant over time (£35) and is always higher than that of the Off-peak service. Whilst the Off-peak price



a. x = (20,40)



b. x = (80,40)

Figure 3: Prices over time for the different available slots using interdependent learning

does increase over time, the difference in the two services remains at £10 until the end of the 20 days. When compared to individual learning policy in Figure 2, the optimal price of Off-peak is greater when using the interdependent learning primarily because the policy takes into account the effect that the price change in one service has on the demand for the other. Figure 3b shows the policy when the availability of the Peak service is high and results suggest that both of the services should be priced equally, even though there are fewer Off-peak slots.

5.2 The Weekly Pricing example

The weekly pricing example attempts to improve the management of demand by using dynamic pricing. Consider a service company which operates from Monday to Friday with a fixed number

of contracted workers. Service requests can arrive at any time during the week, including on weekends. The problem faced by the company is that service requests logged during the weekend have to wait until at least Monday before they can be carried out. It is reasonable to assume that all service requests have to be completed within a certain lead time in order for the company to maintain acceptable levels of quality; this can result in the company having employees work beyond their contracted hours at the start of the working week. Furthermore, it can be assumed that the demand for the service varies during the working week and it is possible for there to be an excess of workers during periods of low demand. If the excess demand seen at the start of the working week (say Monday and Tuesday) can be shifted to the rest of the week then there would be no need for workers to work overtime and therefore it is possible to reduce costs significantly.

This experiment is designed to determine whether dynamic pricing can be used as a mechanism to decrease the cost of having engineers working overtime and to keep the lead time to an acceptable level of quality. If, by introducing a dynamic pricing policy, it is possible to provide the right incentives to price-sensitive customers to choose a period of lower demand, it is possible to reduce the need for overtime working. It may also be possible to increase revenue by charging a higher price for a service to customers who are willing to pay more to receive the service at a convenient time. The objective is, therefore twofold: to improve the allocation of resources in order to increase the expected revenue gained, and to meet pre-determined levels of quality.

To define the problem, consider the five days of the week (Monday to Friday) as five different services. The company has the same capacity each day, i.e., a fixed number of workers, with which to meet demand. Most importantly, it is assumed that the price of the service on one day can affect the demand for the service on another. It is also assumed that the customer can only see the price of the service over the next five days, which reflects the selling horizon $t = 1, 2, 3, 4, 5$, i.e., the length of time over which the services are sold before they expire. At the start of each day of the week the company reviews the stock of each day and is able to change the price of each service. The available slots in each day come from a discrete set between [0, 50] and the price set [35, 50].

The primary objective is to maximise the expected revenue gain during the entire selling horizon, from all services, and therefore the objective function to be maximised is
$\sum_{t=1}^{5} \sum_{d=1}^{5} r(x_t(d), a_t(d))$ where Monday is d=1, Tuesday is d=2, etc.

The demand is characterised by the following equations and parameters, and the realised de-

mand is observed over time. Neither the equations, parameters nor the underlying functional relationship between the price set of multiple products and the demand function that governs these observations are known to the decision maker. The demand for service at day d at price $a_t(d)$ is $D(a_t(d)) = \mu(t) - \zeta_d(a(d)) - \zeta_{d+1}(a(d+1)) - \zeta_{d+2}(a(d+2)) - \zeta_{d+3}(a(d+3))$ $-\zeta_{d+4}(a(d+4)) - \zeta_{d+5}(a(d+5))$. The demand for $d+i$ at price $a_t(d+i)$ is $D(a_t(d+i)) = \zeta_{d+i}(a(d+i))$. The arrival rate $\mu(t)$ is given by $\mu(t) = \theta - 5t$, where θ is the initial customer arrival rate drawn randomly from a uniform distribution $[60, 100]$. As the selling horizon is a moving window, the experiment was repeated 1000 times and the optimal pricing policy learnt by repeating different pricing actions over numerous weeks.

It is assumed that customers are willing to wait to receive the service on a later day if the price is cheaper than on all previous days. As the number of days the customer has to wait increases, the number of customers willing to wait reduces. If customers can purchase the service more cheaply or at the same price on an earlier day they will then choose not to purchase the service on a later day. Both interdependent learning (Theorem 4) and individual learning (Theorem 1) are used in this experiment to generate optimal policies for the five interdependent services. Tables 2 and 3 show the results of a single simulation run, when there are 50 slots available at the start of each day using interdependent and individual learning respectively. Each table shows the optimal prices for the service on each day for the entire week. The rows represent the day of the week (t) on which the decision maker acts, and the columns show the prices for that day and for the next four days. For example, if the decision maker is acting on Tuesday, the table shows the price of the service on the Tuesday (50) and for the following four days, i.e., Wednesday (47), Thursday (40), Friday (37) and Monday (50).

Table 2: Prices for interdependent learning algorithm

	Mon	Tue	Wed	Thur	Fri	Mon	Tue	Wed	Thur
Monday	50	50	46	39	35				
Tuesday	--	50	47	40	37	50			
Wednesday	--	--	47	42	39	50	37		
Thursday	--	--	--	45	42	50	39	35	
Friday	--	--	--	--	45	50	42	40	35

It can be observed that the optimal prices for the services using interdependent learning are, in general, higher when compared to those derived using individual learning. The price for the

Table 3: Prices for individual learning algorithm

	Mon	Tue	Wed	Thur	Fri	Mon	Tue	Wed	Thur
Monday	45	45	37	35	35				
Tuesday	--	45	39	35	35	45			
Wednesday	--	--	40	35	35	45	38		
Thursday	--	--	--	37	35	45	39	35	
Friday	--	--	--	--	38	45	40	35	35

service on Monday is always set to be high by both algorithms, with the interdependent learning algorithm setting it at five units higher. The optimal prices using the individual learning are lower as the prices for services on each day compete with one another; in interdependent learning the services work together to maximise their revenue jointly. The revenue generated using the individual learning is 9680, with a fixed optimal price of 42, is 9828 (1.53% improvement on the individual learning), and using interdependent learning is 10264 (6.03% improvement on the individual learning). From these results it can be seen that pricing the services individually generates less revenue than keeping the policy at a fixed price because the individual learning creates competition between the different days of the week. Further to the experiment described above, the effect of assuming that the demands for each of the days of the week are the same, when in fact the underlying demands are different for each days, is analyzed. The optimal dynamic pricing policy, assuming an initial availability of 50 slots per day, is shown in Table 4. As the observations of all the days are combined to a single pricing policy, all the days of the week will have the same pricing policy.

Table 4: Prices for the combined distribution of demand

	Mon	Tue	Wed	Thur	Fri	Mon	Tue	Wed	Thur
Monday	41	39	38	37	36				
Tuesday	--	41	39	38	37	36			
Wednesday	--	--	41	39	38	37	36		
Thursday	--	--	--	41	39	38	37	36	
Friday	--	--	--	--	41	39	38	37	36

The revenue generated using the pricing policy of the combined demands is 8752 (14.73% less revenue compared to the interdependent learning algorithm). This clearly shows that combining the observations of the days of the week when they have different demands generates less revenue,

illustrating that the simplified model assumption can have large impacts on the total revenue gained.

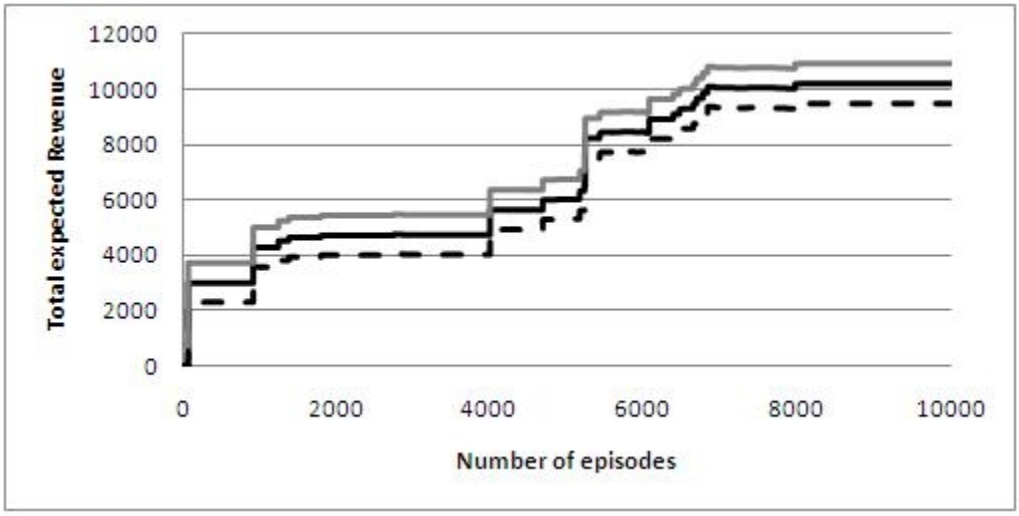


Figure 4: Convergence to the true expected revenue for five services

For practitioners it is important to know how long it takes to learn the optimal policy; here this is addressed by observing the time it takes for five interdependent products to converge to its optimal policy for a given initial state. An experiment was run to determine the convergence of the total expected revenue over the number of episodes. This experiment considered the five services as described in our pricing example. This experiment was run 10000 times and repeated for 50 samples. Figure 1 shows the convergence to the total expected revenue and their respective 95% confidence intervals. In Figure 1 shows discontinuity in the graph; with a larger state and action space it takes longer to explore all possible pricing actions.

In this study we have set the Q-values to zero (without any prior training) to test the worst case situation. Even in the worst case with a large state-space problem with five interdependent services, a reasonable policy can be arrived within 6000 episodes. Companies applying dynamic pricing tend to have vast amounts of data available; they have the same scenarios running daily and, therefore, the actual policy can be initialised with the Q-values set to the their current practice. The only difference between using this approach and the Q(λ) (as seen in Table 1) is that the Q-values are initialised using an estimated demand simulation model as opposed to being set to zeros (without priori training). Rana and Oliveira (2013) have showed that using such an approach leads to higher returns than assuming no prior knowledge and convergence to the near-optimal policy is quicker. Hence companies can start with using their current practice, when

selecting pricing actions, and move toward the new policy derived from the reinforcement learning algorithm approach faster.

6 Conclusions and Discussion

The pricing of interdependent products is a very complex problem due to the exponential growth of the policy space with the number of products considered and because of the requirement to account for the interactions between the pricing of these products. In practice this is a very important problem. It is well-known that the network problem in dynamic pricing is very hard to handle and these interactions are usually ignored and the products are priced independently, or heuristics are required to address this problem instead. It is for these reasons that the major contribution of this article is the presentation of a dynamic pricing model for interdependent products. Unlike in traditional dynamic pricing models, no functional relationship between price and demand was assumed; instead the model learns the relationship explicitly using the observation of realized demand to derive an optimal dynamic pricing policy. Indeed, the main advantage of using a Q-learning with eligibility traces algorithm is that the decision maker can learn how to make pricing decisions without any explicit knowledge of customer buying behavior.

Moreover, in this article we emphasize the advantages of using interdependent dynamic pricing, as opposed to individual pricing of products or services. Firstly, this allows for the better approximation of the optimal policy. By using interdependent pricing, the algorithm is able to use the information from the different products or services to improve the overall profit received from pricing all the items in a consistent way. Secondly, this allows for faster learning when the demand for the different products or services is strongly related. However, as with dynamic programming, the algorithm still suffers from the curse of dimensionality. As the number of interdependent products increases, the speed of convergence decreases exponentially. As the number of interdependent products becomes too large, the decision maker may consider grouping products together if they follow the same demand pattern, as proven in Theorem 2, or jointly pricing highly correlated interdependent products, so that he can arrive at optimal policies in real-time. Alternatively, the decision maker could use function approximation such as regression or Bellman error (Tsitsiklis and Roy, 1997; Gosavi, 2009).

In conclusion, we propose a new formalization of the problem of dynamically pricing interde-

pendent products that is able to improve the quality of the solutions derived when compared with independent pricing without requiring a formal model of customer behavior.

As future research in the area there are several avenues both regarding possible evolutions on the algorithms used to solve the pricing problem and on the applicability of this tool to the solution of complex real word problems. A) We can relax the approach of fixed resources and consider problems where there is flexibility in resources, for example, the repair service problem considered in this paper can be extended to have the option of overtime workers i.e., flexibility in resources. B) Reinforcement learning algorithms with large state-spaces and action spaces face the problem of the slow speed of convergence, e.g., Mabu et al. (2012). The use of function approximators and grouping of correlated products is an area of future research that may improve the speed of convergence of the algorithms, when this is a concern. C) Dynamic pricing is becoming more prevalent in many industries, for example, in electricity and natural gas where real time meetering is used (e.g., Chakraborty et al., 2014); this is a case in which reinforcement learning can play an important role, as it enables the retailers to best learn how to price their products dynamically by learning the optimal pricing policy for different types of consumers. D) Another business area where dynamic pricing usining reinforcement learning can be used is markdown-pricing (in which a firm decides to plan a systematic decrease of prices in order to sell an wanted inventory); in most cases this is a mixed-integer non-linear optimization problem in which the firm needs to decide when and by how much to reduce the price. In the case of some products, such as perishable food (fresh vegetables and meat, for example), this price reductions are repeated daily, allowing the collection of the information needed to use reinforcement learning, when oprimization techniques are hard to apply. E) Finally, hotel management can slso benefit from the algorithm proposed in this article; in this case, the online pricing of the different types of rooms (single vs. double; with different locations in the hotel - sea view vs. city view, for example) can be priced taking into account available capacity; as this is a daily pricing activity the data collected can be used to learn to price online the different types of room with interdependent demands.

7 References

Akcay, Y., H. P. Natarajan, S. H. Xu. 2010. Joint dynamic pricing of multiple perishable products under consumer choice. *Management Sci.* **56**(8): 1345-1361.

Anjos, R., R. C. H. Cheng, C. S. M. Currie. 2004. Maximizing revenue in the airline industry under one-way pricing *J. Oper. Res. Soc.* **55**: 535-541.

Anjos, R., R. C. H. Cheng, C. S. M. Currie. 2005. Optimal pricing policies for perishable products. *Eur. J. Oper. Res.* **166**: 246-254.

Asdemir, K., V. Jacob, R. Krishnan. 2009. Dynamic pricing of multiple home delivery options. *Eur. J. Oper. Res.***196**: 246-257.

Besbes, O., A. Zeevi. 2009. Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Oper. Res.* **57**(6):1407-1420.

Bitran, G. R., R. Caldentey. 2003. An overview of pricing models for revenue management. *Manufacturing & Service Oper. Management.* **53**(3): 203-229.

Bitran, G. R., R. Caldentey, R. Vial. 2004. Pricing policies for perishable products with demand substitution. Working paper.

Burkart, W. R., R. Klein, S. Mayer. 2012. Product line pricing for services with capacity constraints and dynamic substitution. *Eur. J. Oper. Res.***219**:347–359.

Chakraborty, S., T. Ito, and T. Senjyu. 2014. Smart Pricing Scheme: A Multi-Layered Scoring Rule Application. *Expert Systems with Applications***41**: 3726-3735.

Chan, C.-C. H., C.-B. Cheng, and W.-C. Hsien. 2011. Pricing and Promotion Strategies of an Online Shop Based on Customer Segmentation and Multiple Decision Making. *Expert Systems with Applications***38**: 14585-14591.

Cheng, Y. 2007. Dynamic pricing for multi-product in e-retailing. *Intern. Confer. Wireless Comm. Networking Mobile Comput.* 5476-5479.

Cheng, Y. 2009. Dynamic Packaging in e-Retailing with Stochastic Demand over Finite Horizons: A Q-Learning Approach. *Expert Systems with Applications***36**: 472-480.

Cooper, W. L., T. Homem-de-Mello, A. J. Kleywegt. 2006. Models of the spiral-down effect in revenue management. *Oper. Res.* **54**(5): 968-987.

Currie, C. S. M., R. C. H. Cheng, H. K. Smith. 2008. Dynamic pricing of airline tickets with competition. *J. Oper. Res. Soc.* **59**: 1026-1037.

Dorca, F. A., L. V. Lima, M. A. Fernandes, C. R. Lopes. 2013. Comparing strategies for modeling students learning styles through reinforcement learning in adaptive and intelligent educational systems: An experimental analysis. . *Expert Systems with Applications.* **40**: 2092-2101

dos Santos, J.P.Q., J. D. de Melo, A. D. D. Neto, and D. Aloise. 2014. Reactive Search

Strategies using Reinforcement Learning, Local Search Algorithms and Variable Neighborhood Search. *Expert Systems with Applications.* bf1: 4939-4949.

Eom, S. B. 1996. A Survey of Operational Expert Systems in Business (1980-1993). *Interfaces***26** 5: 50-70.

Gallego, G., G. J. van Ryzin. 1994. Optimal dynamic pricing of inventories with stochastic demand over finite horizon. *Management Sci.* **40**: 999-1020.

Gallego, G., G. J. van Ryzin 1997. A multi-product dynamic pricing problem and its applications to network yield management. *Oper. Res.* **45**: 24-41.

Gosavi, A. 2009. Reinforcement Learning: A Tutorial Survey and Recent Advances. *INFORMS J. Comput.* **21**(2): 178-192.

Gosavi, A. N., N. Bandla, T. K. Das. 2002. A reinforcement learning approach to a single leg airline revenue management problem with multiple fare classes and overbooking. *IIE Trans.* **34**: 729-752.

Iseri, A., and B. Karlik. 2009. An Artificial Neural Networks Approach on Automotive Pricing. *Expert Systems with Applications***36**: 2155-2160.

Jiang, C., Z. Sheng. 2009. Case-based reinforcement learning for dynamic inventory control in a multi-agent supply-chain system. *Expert Systems with Applications* **36**(3): 6520-6526

Kaelbling, L. P., M. L. Littman, A. W. Moore. 1996. Reinforcement learning: A survey. *J. Artificial Intelligence Res* **4**: 237-285.

Kim, S. W., P. C. Bell. 2011. Optimal pricing and production decisions in the presence of symmetrical and asymmetrical substitution. *Omega* **39**: 528-538.

Kunnumkal, S., and H. Topalogula 2008. Exploiting the Structual Properties of the Underlying Markov Decision Problem in the Q-Learning Algorithm. *INFORMS J. Comput.*, **20**(2): 288-301.

Kwon, I.-H., C. O. Kim, J. Jun, and J. H. Lee. 2008. Case-based Myopic Reinforcement Learning for Satisfying Target Service Level in Supply Chain. *Expert Systems with Applications***35**: 389-397.

Lee, K. C., H. Lee and N. Lee. 2012. Agent Based Mobile Negotiation for Personalized Pricing of Last Minute Theatre Tickets. *Expert Systems with Applications***39**: 9255-9263.

Liao, S.-H. 2005. Expert System Methodologies and Applications - A Decade Review from 1995 to 2004. *Expert Systems with Applications***28**: 93-103.

Lim, A. E. B., J. G. Shanthikumar. 2007. Relative entropy, exponential utility, and robust

dynamic pricing. *Oper. Res.* **55**: 198-214.

Mabu, S., A. Tjahjadi, K. Hirasawa. 2012. Adaptability analysis of genetic network programming with reinforcement learning in dynamically changing environments. *Expert Systems with Applications*, **39** (16): 12349-12357

Maglaras, C., J. Meissner. 2006. Dynamic pricing strategies for multi-product revenue management problems. *Manufacturing Service Oper. Management*, **8**(2): 136-148.

McGill, J. I., G. J. van Ryzin. 1999. Revenue management: Research overview and prospects. *Transportation Sci.*, **33**(2): 233-256.

Netessine, S., S. Savin, W. Q. Xiao. 2006. Revenue management through dynamic cross-selling in e-commerce retailing. *Oper. Res.*, **54**(5): 893-913.

Oliveira, F. S. 2008. A Constraint Logic Programming Algorithm for Modeling Dynamic Pricing. *INFORMS J. Comput.* **20**(1): 69-77.

Oliveira, F. S. 2014. Reinforcement Learning for Business Modeling. In *Encyclopedia of Business Analytics and Optimization*, J. Wang (Ed.), 2010-2019.

Palombarini, J., and E. Martinez. 2012. SmartGantt - An Intelligent System for Real Time Rescheduling Based on Relational Reinforcement Learning. *Expert Systems with Applications***39**: 10251-10268.

Peng, J., R. J. Williams. 1996. Incremental multi-step Q-learning. *Machine Learn.* **22**:283-290.

Peteiro-Barral, D., B. Guijarro-Berdias, B. Prez-Snchez, O. Fontenla-Romero. 2013. A comparative study of the scalability of a sensitivity-based learning algorithm for artificial neural networks. *Expert Systems with Applications* **40** (10): 3900-3905.

Puterman, M. L. 1994. Markov Decision Processes: Discrete Stochastic Dynamic Programming. John Wiley & Sons, Inc.

Raju C., Y. Narahari, K. Ravikuma. 2006. Learning dynamic prices in electronic retail markets with customer segmentation. *Ann. Oper. Res.* 59-75.

Rana R., F. S. Oliveira. 2013. Real-Time Dynamic Pricing in a Non-Stationary Environment Using Model-Free Reinforcement. *Omega* http://dx.doi.org/10.1016/j.omega.2013.10.004

Sen, A. 2013. A comparison of fixed and dynamic pricing policies in revenue management. *Omega* **41**: 586–597.

Shin, M., K. Ryu, and M. Jung. 2012. Reinforcement Learning Approach to Goal-Regulation in a Self-Evolutionary Manufacturing System. *Expert Systems with Applications***39**: 8736-8743.

Sohn, S. Y., T. H. Moon, and K. J. Seok. 2009. Optimal Pricing for Mobile Manufacturers in Competitive Market Using Genetic Algorithm. *Expert Systems with Applications***36**: 3448-3453.

Sutton, R., A. G. Barto. 1998. Reinforcement Learning. The MIT Press, Cambridge, Massachusetts.

Talluri, K. T., G. J. van Ryzin. 2005. The Theory and Practice of Revenue Management. Springer. New York, NY.

Tsitsiklis, J. N., B. Van Roy. 1997. An analysis of temporal-difference learnign with function approximation. *IEEE Trans.* **42**(5): 674-690.

Zhang, D., W. L. Cooper. 2006. Revenue management for parallel flights with customer choice behavior. *Oper. Res.*, **53**(3): 415-431.

Zhao, W., Y. S. Zheng. 2000. Optimal dynamic pricing for perishable assets with nonhomogeneous demand. *Management Sci.*, **46**(3): 375-388.

Appendix

Theorem 1.1 - Proof

The Q-values for interdependent products are formulated by $Q_t^*(X_t, A_t) = r(X_t, A_t) + \sum_{X_{t+1}} p_t(X_{t+1} \mid X_t, A_t)\eta \max_{A_{t+1}} Q_{t+1}^*(X_{t+1}, A_{t+1})$. If in vector A_t only $a_t(j)$ is controlled by the decision maker and all other decisions are not observed, and the states of the other variables are not observed within them, then $r(X_t, A_t) = r(x_t(j), a_t(j))$, where $r(x_t(j), a_t(j)) =$
$\sum_{A_{(t,-j)}} w_t[x_t(j), a_t(j), A_{(t,-j)}] r(x_t(j), a_t(j), A_{(t,-j)})$ and $p_t(X_{t+1} \mid X_t, A_t) = p_t(x_{t+1}(j) \mid x_t(j), a_t(j))$ hold. The decision maker does not know the state of the interdependent products. $X_{t,-j}$ does not directly influence customers buying behaviour but determines the prices of the interdependent products. The decision maker only observes transition probabilities and these are the weighted average implied by the prices and state of the other interdependent products as follows $p_t(x_{t+1}(j) \mid x_t(j), a_t(j)) = \sum_{A_{(t,-j)}} w_t[x_t(j), a_t(j), A_{(t,-j)}] p_t(x_{t+1}(j) \mid x_t(j), a_t(j), A_{(t,-j)})$. Then $Q_t^*(X_t, A_t) =$
$\sum_{A_{(t,-j)}} w_t[x_t(j), a_t(j), A_{(t,-j)}][r(x_t(j), a_t(j), A_{(t,-j)}) +$
$\sum_{x_{t+1}} p_t(x_{t+1}(j) \mid (x_t(j), a_t(j), A_{(t,-j)}))\eta \max_{A_{t+1}} Q_{t+1}^*(x_{t+1}(j), A_{t+1})]$. Finally as we only control $a_t(j)$ and observe $x_{t+1}(j)$ we get that $A_{t+1} = a_{t+1}(j)$ and $Q_t^*(X_t, A_t) = Q_t^*(x_t(j), a_t(j))$,
$Q_{t+1}^*(X_{t+1}, A_{t+1}) = Q_{t+1}^*(x_{t+1}(j), a_{t+1}(j))$, and $Q_t^*(x_t(j), a_t(j)) =$
$\sum_{A_{(t,-j)}} w_t[x_t(j), a_t(j), A_{(t,-j)}][r(x_t(j), a_t(j), A_{(t,-j)}) +$

$\sum_{x_{t+1}} p_t(x_{t+1}(j) \mid (x_t(j), a_t(j), A_{(t,-j)}))\eta \max_{a_{t+1}} Q^*_{t+1}(x_{t+1}(j), a_{t+1}(j))]$
$= \sum_{A_{(t,-j)}} w_t[x_t(j), a_t(j), A_{(t,-j)}]Q^*_t(x_t(j), a_t(j), A_{(t,-j)})$. QED

Theorem 1.2 - Proof

If the demands are the same then $r(X_t, A_t) = r(X'_t, A'_t)\ \forall X_t, X'_t \in \mathbf{X}, A_t, A'_t \in \mathbf{A}(\mathbf{X_t})$, $X_t = X'_t$, $A_t = A'_t$ and the optimal pricing policy for the first set of n products will be the same as the second set of n products, $\pi^*(X_t) = \pi^*(X'_t)$. Then we can put the observations for the products together $< X_{t,k}, A_{t,k}, X_{t-1,k}, r_{t-1,k}, ...X'_{t,k}, A'_{t,k}, X'_{t-1,k}, r'_{t-1,k}, .., X'_{1,1} >$ and use them to update the Q-values. It follows $\forall\ (X_t, A_t) = (X'_t, A'_t)$, $\alpha'_k(X_t, A_t) \leq \alpha_k(X_t, A_t))$ and $\alpha'_k(X_t, A_t) \leq \alpha_k(X'_t, A'_t)$ where, $\alpha'_k(X_t, A_t)$, $\alpha_k(X_t, A_t)$, $\alpha_k(X'_t, A'_t)$ are the learning rates of the combined observations of the products, for the first set of n products and the second set of n products, at episode k, respectively. The learning rate $\alpha_k(X_t, A_t)$ is equal to $1/n_k(X_t, A_t)$, where $n_k(X_t, A_t)$ is the number of times the state-action (X_t, A_t) was visited by the process $(X_{t,k}, A_{t,k})$ before episode k, plus 1.

Consider Q(λ) in a finite MDP where the sequence is $< X_{t,k}, A_{t,k}, X_{t+1,k}, r_{t,k} >$. Assume that the learning rate sequence $\alpha_k(X_t, A_t)$ satisfies the following:
1. $0 \leq \alpha_k(X_t, A_t)$, $\sum_{t=0}^{\infty} \alpha_k(X_t, A_t) = \infty$, $\sum_{t=0}^{\infty} \alpha_k^2(X_t, A_t) < \infty$ are uniformly and hold w.p.1
2. $\alpha_k(X_t, A_t) = 0\ if\ \alpha_k(X_t, A_t) \neq \alpha_k(X_{t,k}, A_{t,k})$ w.p.1. Then the values calculated by $Q(\lambda)$ algorithm converges to Q^*.

Then $0 \leq \alpha_k(X_t, A_t) \leq 1$, $t \geq 0$, and $\sum_{k=0}^{\infty} \alpha_k(X_t, A_t)$ converges uniformly in X as $k \longrightarrow \infty$ is a condition for Q_k converges to Q^*. The smaller $\alpha_k(X_t, A_t)$ the closer Q_k is to Q^*. Then it follows that $Q'_{t,k}(X_t, A_t)$ converges Q^*_t faster than $Q_{t,k}(X_t, A_t)$ and $Q_{t,k}(X'_t, A_t)$. QED

Theorem 1.3 - Proof

If all the observations from the first set of n products and the second set of n products are put together $< X_{t,k}, A_{t,k}, X_{t-1,k}, r_{t-1,k}, ..., X'_{(t,k)}, A'_{t,k}, X'_{t-1,k}, r'_{t-1,k}, .., X'_{1,1} >$ and used to update the Q-values, the optimal Q-values of the combined observation are calculated $\forall\ X_t = X'_t$ by
$\pi^*(X_t) = \max Q^*_t(X_t, A_t) = \max_{A_t}\{r(X_t, A_t) + \sum_{X_{t+1}} p_t(X_{t+1} \mid X_t, A_t)\eta \max_{A_{t+1}} Q^*_{t+1}(X_{t+1}, A_{t+1})\}$
as we have assumed that $\pi^*(X_t) = \pi^*(X'_t)$ and $Q^*_t(X_t, A_t) = Q^*_t(X'_t, A_t)$ holds, then $\forall\ X_t = X'_t$,
$\max_{A_t}\{r(X_t, A_t) + \sum_{X_{t+1}} p_t(X_{t+1} \mid X_t, A_t)\eta \max_{A_{t+1}} Q^*_{t+1}(X_{t+1}, A_{t+1})\}$
$= \max_{A'_t}\{r(X'_t, A'_t) + \sum_{X_{t+1}} p_t(X'_{t+1} \mid X'_t, A'_t)\eta \max_{A'_{t+1}} Q^*_{t+1}(X_{t+1}, A'_{t+1})\}$. Then $r(X_t, A_t) = r(X'_t, A'_t)$ is a contradiction therefore $\pi^*(X_t) \neq \pi^*(X'_t)\ \forall\ \mathrm{X}_t = X'_t$. Hence combining the observation gives a

non-optimal policy and in this case it is better to learn the policy for the products separately. QED

Theorem 1.4 - Proof

The Q-values for n interdependent products or services is calculated by $Q_t^*(X_t, A_t) = r(X_t, A_t) + \sum_{X_{t+1}} p_t(X_{t+1} \mid X_t, A_t)\eta \max_{A_{t+1}} Q_{t+1}^*(X_{t+1}, A_{t+1})$. The Q-values for each product, for all product $j = 1, .., n$ is calculated by

$Q_t^*(x_t(j), a_t(j)) = r_t(x_t(j), a_t(j)) + \sum_{x_{t+1}} p_t(x_{t+1}(j) \mid x_t(j), a_t(j))\eta \max_{a_{t+1}} Q_{t+1}^*(x_{t+1}(j), a_{t+1}(j))$

$= \sum_{A_{(t,-j)}} w_t(x_t(j), a_t(j), A_{(t,-j)}) Q_t^*(x_t(j), a_t(j), A_{(t,-j)})$.

(proved in Theorem 1.1). When calculating the $\max_{A_t} Q_t^*(X_t, A_t)$ we choose $a_t(1)$, $a_t(2)$.. and $a_t(n)$ at the same time and hence the weights do not sum over the other actions, say all possible $A_{t,-j}$, but you use the optimal A_t, say A_t^*. By definition of optimality it follows $\max_{A_t} Q_t^*(X_t, A_t) \geq$

$\sum_{j=1}^{n} \max_{a_t(j)} \sum_{A_{(t,-j)}} w_t(x_t(j), a_t(j), A_{(t,-j)}) Q_t^*(x_t(j), a_t(j), A_{(t,-j)})$

$= \sum_{j=1}^{n} \max_{a_t(j)} Q_t^*(x_t(j), a_t(j))$ holds. QED.

Theorem 1.5 - Proof

The state space of u and v independent products at time t is denoted by $\mathbf{X_t^u}$ and $\mathbf{X_t^v}$ respectively, and let $\{1, ..., n\}$ be a set of all the possible states of a single product. Then the combination of all possible states for u products is greater than v products. $\forall\ t$, $\mathbf{X_t^u} = \{1, ..., n\} \times ..\{1, ..., n\}^u > \{1, ..., n\} \times ..\{1, ..., n\}^v = \mathbf{X_t^v}$. The same holds for the action space $\mathbf{A_t^u}$ and $\mathbf{A_t^v}$, $\mathbf{A_t^u} = \{1, ..., L\} \times ..\{1, ..., L\}^u > \{1, ..., L\} \times ..\{1, ..., L\}^v = \mathbf{A_t^v}$. where, $\{1, ..., L\}$ is the set of all the possible pricing actions of a single product. Then it follows that the state-space of u products is greater than v products. $\forall t\ \mathbf{X_t^u} \times \mathbf{A_t^u} > \mathbf{X_t^v} \times \mathbf{A_t^v}$ and then the $\#\alpha(X_t^u, A_t^u) = n^u L^u > \#\alpha(X_t^v, A_t^v) = n^v L^v$ $\forall X_t^u \in \mathbf{X_t^u}$, $X_t^v \in \mathbf{X_t^v}$ $A_t^u \in \mathbf{A_t^u}$, $A_t^v \in \mathbf{A_t^v}$. All estimated Q-values $(Q_{t,k})$ will converge to their optimal Q-values (Q_t^*), if all $\alpha_k(X_t, A_t)$ satisfy the condition that $\sum_{k=0}^{\infty} \alpha_k(X_t, A_t)$ converges uniformly as $k \to \infty$. Then since u products have a greater number of states-action space and therefore a greater number of learning rates, the rate of its convergences will be $1/(n^{(v-u)} L^{(v-u)})$ slower than v number of products, to its optimal policy. QED