Information Bundling in a Dynamic Environment

Christopher H. Brooks* Rajarshi Das† Jeffrey O. Kephart†
Jeffrey K. MacKie-Mason‡ Robert S. Gazzale§ Edmund H. Durfee*

Abstract

Digital information goods potentially provide information producers with a new set of strategies, or price schedules, for offering these goods to a consumer population. If consumer preferences are known, then a producer can choose from the available schedules according to the profits they are able to extract. However, if the population is not known, then a producer must weigh a schedule's complexity against its potential profitability. In addition, the producer must consider the learnability of the environment; is the consumer population changing, and how much information does it have about the nature of this change?

In this work, we present a model of a monopolist producer learning the parameters of a number of price schedules when the consumer population is nonstationary. We adapt amoeba, an optimization algorithm, to effectively handle nonstationary profit landscapes and compare the performance of different price schedules as the rate of consumer change and knowledge about consumer preferences is varied. We find that two-part tariff, a two-parameter price schedule, performs well, due to its low complexity and its robustness to change in the consumer population.

1 Introduction

Digital information goods typically have negligible incremental reproduction and distribution costs once the good is produced. An information goods producer can almost costlessly package these goods in a wide variety of configurations, opening the possibility for more complex product and pricing configurations. With multiple items (say, articles) to sell, a producer might offer a single price per article (linear pricing), sell the whole lot for a subscription price (pure bundling), or any number of other variants.

In an economy populated by computational agents, price schedules may be changed quickly and easily. Software agents can evaluate and select from among many complex pricing schemes more easily than a human, without suffering from the confusion or frustration that humans experience when faced with complicated pricing schemes.

Prior authors [6, 2, 15] have analyzed the benefit of bundling and other pricing schemes based on the assumption that the producer knows the distribution of consumer preferences. In practice, producers are unlikely to have complete knowledge of all relevant parameters of this distribution. Thus, if there is more than one period of activity, there will be two sources of value from any price schedule: the profit received and the information obtained about consumer preferences. The pricing problem is inherently dynamic as information acquired in one period affects a producer's actions in later periods. Since the price that yields the most valuable information about consumer preferences typically will not be the price that maximizes current profits, there is a tradeoff between *exploration* and *exploitation*. Exploitation will yield a known profit, but give no new information about consumer preferences or the existence of better prices. Exploration can yield this information, but carries the risk of lower current profits.

The many pricing schedules available to an information goods producer make the pricing problem especially acute. Two different schedules may have different *potential profitability*, defined as the profits available

*Artificial Intelligence Laboratory, University of Michigan, Ann Arbor, MI, 48109, {chbrooks, edurfee}@umich.edu
†Institute for Advanced Commerce, IBM Research, P.O. Box 704, Yorktown Heights, NY 10598, {rajarshi, kephart}@watson.ibm.com
‡School of Information and Department of Economics, University of Michigan, Ann Arbor, MI 48109, jmm@umich.edu, (734) 647-4856
§Department of Economics, University of Michigan, Ann Arbor, MI 48109, rgazzale@umich.edu

from that schedule if all of the relevant consumer preference parameters were known with certainty. Likewise, two pricing schedules may differ in their *complexity*, defined as how difficult it is for the firm to learn their optimal parameters. Thus, to select from the many possible pricing schedules, a producer must assess the tradeoff between profitability and complexity for each, compute their expected economic performance, and select the best.

In prior work we have studied the profitability/complexity tradeoff for a variety of information good pricing schedules and compared their economic performance [5]. We did so for a *stationary* environment: a single producer offered price schedules to a static population of consumers. Thus, given enough time, the producer could eventually find the maximal point on the profit surface induced by any of the price schedules. However, the best price schedule was not generally the one with the highest potential profit, since the producer's objective is to maximize accumulated total profit and low profits in the learning phase are costly. We found that when the profit potential for a relatively simple (low-dimensional) price schedule was not too low relative to the schedule with the maximal profit potential (always the most complex), the simpler schedule performed better due to its lower learning cost.

Of course, the value of exploration depends on a producer's ability to use this knowledge in later periods. The durability of current learning depends in some sense on the *stationarity* of the environment. The environment facing an information goods producer can change for several reasons. The preferences of individuals participating in the market may change; for this and other reasons potential customers can enter or exit the market, changing the distribution of preferences. Likewise, if competitors' price and product offerings change, the set of consumers interested in a given producer's goods will change, thereby changing the the residual distribution of preferences.

Nonstationarity can alter the tradeoff between exploration and exploitation. In this paper we study the economic performance of information goods price schedules in the face of nonstationarity. We characterize the learnability of the environment, by which we mean the ease of learning the underlying consumer preferences. We identify three factors that affect learnability in our problems: the quality of the current position, the frequency of environmental changes, and the magnitude of environmental changes. Our results suggest that the lower learning costs of less complex price schedules have an advantage in less learnable environments, to be considered against the cost of their lower potential profit.

We also observe some important implications of nonstationarity for the design of the learning algorithm used by the producer. Most standard machine learning algorithms are designed for stationary environments. We focus on two issues: First, the algorithm must have a procedure for (tentatively) detecting changes in the environment. Second, the algorithm should be designed with a procedure that allows it to adapt to environmental change once detected. Since these tasks must be solved before the learning characteristics of different price schedules become relevant in a nonstationary environment, we address the design of the learning algorithm before analyzing the performance of different price schedules below.

1.1 Related work

Economists have previously studied the use of prices to learn about consumer preferences. Grossman, Kihlstrom and Mirman [9] were among the first to identify the desirability of deviating from the myopically optimal prices in order to increase learning. The existing literature almost universally assumes that the preferences to be learned are stationary. For example, Aghion et al. [1] determine conditions under which "complete" learning will occur in the long run in stationary environments. However, Keller and Rady [10] consider a monopolist that faces a nonstationary demand curve that alternates between two states. They find that for a given level of noise and a given discount rate, low rates of state changes will lead to "extreme" experimentation where the monopolist tracks the true state rather well, while higher rates of state changes will lead to "moderate" experimentation where the monopolist chooses actions that do not provide much information. We study provider price learning in environments that exhibit less restrictive forms of nonstationarity. Second, because we focus on information goods, we allow the provider to consider a variety of pricing schemes.

Some research in machine learning addresses nonstationary environments. Early examples include reinforcement algorithms such as Q-learning and TD-learning [12], and signal-processing algorithms such as Parzen windows [7], which can nonparametrically learn a changing probability distribution. However, these methods require a large amount of data, which makes them unsuitable for our problem. Some other strategies (e.g. [13] and [8]) rely on a non-economic definition of optimality or more restrictive assumptions about the nature of the problem to be optimized.

1.2 Problem statement

We study a single producer that has control over N information items each period. We treat these items as proprietary (e.g., copyrighted), and ignore competition from any other firms that might offer different (imperfectly substitutable) items. In each period the producer may costlessly change the parameters of its price schedule. Each consumer purchases the set of goods that maximizes its surplus in that period. The producer has no reproduction or distribution costs, so its profits equal the total revenues collected from the consumers.

The producer observes consumer purchases, but is not able to identify individual consumers from period to period, so the only information obtained is about the distribution of preferences in the set of currently active consumers. In each subsequent period the producer can change its price schedule parameters, which might change the period profit it obtains and provide it new information about the profit landscape. The producer's objective is to maximize its cumulative profit.

One of our primary objectives is to study the performance of different families of price schedules when the environment is unpredictably (from the producer's perspective) changing over time. Our design objective is to implement an information economy in which the producer faces substantial uncertainty about unobservables (such as the private preference information of consumers), and in which the environment changes sufficiently that there is at least the potential for substantial performance improvement by attempting to adapt and search for the new profit-maximizing price schedule parameters.

While it is important to us that the environment be sufficiently nonstationary that adaptive learning is potentially worthwhile, it is not important to us to represent a specific story about where the nonstationarity comes from. It could be that consumers randomly enter and exit the active buying population; it could be that exogenous world events lead to changes in preferences for different types of information; it could be that exogenous changes in the information content offered by or technologies available to competitors changes the preferences or composition of consumers for the firm in question. As long as there is sufficient nonstationarity in our environment, we are not concerned with which of these (or other) stories is most "realistic" for a particular application.

2 Modeling

In this section we describe consumer preferences and optimizing behavior, the specification of the provider's problem, and the ways in which the population changes over time.

2.1 Consumer Model

Each period, consumers are faced with the producer's offering, in the form of a price schedule (see section 3.1). Each consumer might value different articles differently. Further, a given article might be valued differently by different consumers. Duplicate copies of a particular article have no value. We model consumers using a simple two-parameter model introduced by Chuang and Sirbu [6]. This formulation has the advantage of being analytically tractable, which allows us to generate the optimal results shown in table 1, while still providing significant nonlinearities in aggregate consumer demand when consumers are heterogeneous.

The two parameters are w, a consumer's value for its most-preferred article, and k, the fraction of the N articles available for which the consumer has a positive valuation. The valuation $V_j(i)$ of the ith most-preferred article by consumer j is a linear function of these variables, expressed by:

$$V_j(i) = \begin{cases} w_j(1 - \frac{i-1}{k_j N}) & \text{if } i-1 \leq k_j N \\ 0 & \text{if } i-1 > k_j N. \end{cases} \tag{1}$$

We consider price schedules in which the total payment $P(n)$ depends only on the number of articles purchased. Consumer j's surplus from reading the n^* most preferred articles is thus the aggregate value less any payments made to the producer, $S_j = \sum_{i=1}^{n_j^*} V_j(i) - P(n_j^*)$. Consumers are risk-neutral expected utility maximizers, and thus each chooses n_j^* to maximize S_j. We assume that a consumer always has sufficient computational power to compute its optimal purchase.

In this work, we limit consumer heterogeneity by assuming that w is the same for all consumers. Consumers differ in their values of k_j, which are distributed uniformly between 0 and $\overline{k}$.

2.2 Producer Model

A monopolist producer produces N articles in each period. The producer's performance is measured as the undiscounted sum of profits over the run of the simulation.[1]

As noted above, we assume that the producer can only model the consumer population as an aggregate. This implies that a producer may freely choose from the family of price schedules described in Section 3.1, but must offer all consumers the same schedule.

We assume that the producer does not know the distribution of w and k across the consumer population, and in fact does not even know that preferences take the form given in (1), above. In a given period the producer announces a price schedule and receives a profit. The price schedule announced in the next period will incorporate the information learned from this period's price and profits.

Note that we do not assume that the producer has any knowledge about the source of consumer nonstationarity or the periodicity of population change. The producer must detect that the population has changed by sampling the profit landscape, and then adjust its learning to this new landscape.

2.3 Population Model

We introduce nonstationarity by allowing the composition of the consumer population to vary over time. The producer must thus be concerned about transitional performance as it will have a limited time to recoup learning costs.

We consider a simple model of nonstationarity in which a consumer population is repeatedly resampled from a fixed population with each individual having $w = 10$ and k drawn from $U[0, 0.7]$. Thus, the underlying population distribution of preferences is stationary, but the realized sample of preferences is randomly sampled from that distribution and thus fluctuates.[2]

Resampling happens at discrete time points. At a given time, the entire consumer population is replaced by a new set of consumers. We refer to this as a *shock*. By varying the amount of time between shocks, we can control the amount of volatility in the consumer population and the fraction of time a producer must spend learning about the population.

We have also found that the producer's performance can depend upon where in the pricing space it begins its search for the optimal price. In order to quantify this effect, we give the producer a startup time, in which it interacts with an unchanging population. After this startup time, the shocks begin. By varying the startup time, we move from a situation in which a producer begins interacting with an unknown and changing population to one in which a previously stable population suddenly begins to change, perhaps due to the entrance of other producers or a change in consumer preferences.

It is important that each shock change the consumer population enough to make re-learning about the new population useful. In other words, the variance between realized populations must be sufficiently high. We implement this by using a small (size 10) consumer population, in which each 'consumer' may represent a large number of identical consumers. Just as with our other implementation details, this small number of consumers should not be overinterpreted: it is simply a convenient way to ensure that there is a sufficient amount of variation in the environment over time to justify investing effort in learning.

3 Learning and Model Complexity

In this section, we describe the price schedules a producer can offer, emphasizing the tradeoff between a schedule's complexity and the profit it can extract when the consumer preferences are known exactly. We then

[1]Introducing reasonable discount rates interior to $(0, 1)$ would not change our qualitative results.

[2]We have also run most of the simulations reported in this paper using a model in which nonstationarity is generated by a random walk or drift in the underlying distribution of preferences. Our results are qualitatively identical and so are left out to conserve space.

Pricing Schedule	**Description**	$\Pi_{\textbf{opt}}(\infty, \infty)$	$\Pi_{\textbf{opt}}(100, 100)$	$\Delta\Pi_{\textbf{opt}}(100, 100)$
Pure Bundling	Consumers pay a fixed price b for access to all N articles.	0.875	0.9497 (0.0878)	0.0666
Linear Pricing	Consumers pay a fixed price p for each article purchased.	0.875	0.9044 (0.0515)	0.0042
Two Part Tariff	Consumers pay a subscription fee f, along with a fixed price p for each article	1.037	1.1043 (0.0777)	0.0464
Mixed Bundling	Consumers have a choice between a per-article price p and a bundle price b	1.037	1.1071 (0.0766)	0.0470
Block Pricing	Consumers pay a price p_1 for the first j articles, and a price p_2 for remaining articles.	1.094	1.158 (0.0769)	0.0383
Nonlinear Pricing	Consumers pay a different price p_i for each article i.	1.152	1.2564 (0.0842)	0.0463

Table 1: This table presents characteristics and optimal profits for a series of price schedules, when applied to the following population: $w = 10$, $N = 100$, and k drawn from $U[0, 0.7]$. Column 3 gives the optimal profit per good sold that can be earned by each schedule, with a bundle size $N \to \infty$ and a population size $n \to \infty$. Column 4 gives the corresponding results for a bundle size $N = 100$ and a population size $n = 100$. The standard deviation across different realizations of the landscape are given in parentheses. Column 5 indicates the average additional profit that can be obtained by re-optimizing a landscape after redrawing a new population, assuming bundle size $N = 100$ and population size $n = 100$.

describe the amoeba optimization algorithm and our modifications for nonstationary environments. Finally, we present a set of experiments comparing the performance of each schedule as both the frequency of shocks and the amount of startup time a producer has is varied.

3.1 Schedules

When deciding how to price a bundle of information goods, a producer must trade off the potential profitability of a schedule against the difficulty of accurately learning that schedule. A price schedule simply indicates how much a consumer pays for each article purchased. Simple schedules, such as linear pricing, have only one parameter, making them easy to learn, but are able to extract less profit once the optimal parameters are learned. In contrast, nonlinear pricing has N parameters, allowing it to fit aggregate demand exactly, but potentially requiring a great deal of data to accurately learn all the parameters. [3] This can be visualized as fitting a nonlinear curve; more parameters allow the curve to be fit more closely, but increase the problem's complexity.

Table 1 describes the six schedules we have examined and their expected surplus as a function of the bundle size N and the population size n. In the limit of $N \to \infty$ and $n \to \infty$, an exact analysis of the schedules is possible [5]. These values, scaled by dividing by N, are given in column 3 of Table 1.

Column 4 of Table 1 gives the peak profits for finite consumer populations, averaged over 10,000 landscapes, each of which was generated by choosing 100 random consumers from the given distribution. The peaks were determined by running the amoeba optimization algorithm 100 times on each landscape and choosing the highest peak found over the 100 runs. Previous experimentation had established that this was much more than sufficient to find the global peak for the two-part tariff landscape. Taking the two-part tariff case as an example, the average peak profit of 1.1043 is measurably higher than the peak of the ideal landscape, which is 1.0710. This comes about because a realized landscape is bound to have portions which are both lower and higher than the ideal, and we are selecting the highest point on each landscape. The numbers in parentheses represent the standard deviation across different realizations of the landscape.

In an environment in which the profit landscape changes but the consumers are drawn from a fixed distribution, it is of interest to characterize how these shifts affect the overall profit. To do this, for each successive landscape we evaluated both the peak profit for that landscape and the profit that would have been attained at

[3] A producer may not need to constrain itself to the choice of a single schedule. Some of our other work [3] has explored how a producer can choose a series of schedules. In this work, we will assume that a producer picks one schedule and stays with it.

the price parameters for the previous landscape. The resultant difference is reported in column 5 of Table 1. This difference provides a sense of the amount of gain there is to be had from trying to track a changing but statistically stationary consumer population, as opposed to simply riding out the statistical fluctuations. The actual amount of gain may be less if the optimization algorithm fails to find the new global optimum in the landscape whenever the landscape shifts, either due to getting stuck at a local optimum or having insufficient time before the landscape changes again.

For smaller consumer populations, we have verified that the amount of potential gain is larger. This is to be expected, as the relative size of the bumps in the landscape is larger.

3.2 Amoeba

Amoeba is a variant of the *simplex algorithm* [11] for nonlinear unconstrained optimization problems. (This simplex algorithm should not be confused with the simplex algorithm for linear programming.) The amoeba algorithm maintains at each iteration a nondegenerate simplex, which is a geometric figure in n dimensions of nonzero volume that is the convex hull of $n + 1$ vertices, $x_0, x_1, ..., x_n$, and their respective function values. Amoeba alternately reflects, contracts, and expands the simplex in an attempt to locate an optimum in a function's landscape. In each iteration, a new point (in this case, prices) is chosen and the corresponding value (profit) is determined.[4]

Especially relevant for our work is the fact that amoeba does not make use of any gradient information when finding optima. This makes it a particularly suitable choice for this problem, as the profit landscape the producer must search is dotted with peaks and discontinuities.

Amoeba was designed for searching a fixed, deterministic landscape. In order to contend with a changing profit landscape, a producer must be able to detect changes in the environment when they occur. Also, once changes are detected, the producer must be able to search the new profit landscape. Standard amoeba fails to meet both these criteria. In order to detect a change, a producer using amoeba must be able to observe that a previously visited point now yields a different profit. Since amoeba is essentially a "memoryless" algorithm - the simplex only contains the last $n+1$ observed points - it cannot detect changes even if it revisits previously sampled points. Additionally, our experiments show that amoeba is quick to exploit a high-quality solution by rapidly shrinking the n dimensional simplex around it. While this characteristic potentially helps a producer who is using amoeba to locate optima in static environments, it also makes amoeba sluggish in dynamic situations. Even if amoeba can detect changes in the profit landscape, it first has to spend several steps expanding a highly contracted simplex before it can effectively explore the new environment.

We address these limitations in amoeba with some straightforward extensions. In order to detect that a change has occurred, we allow amoeba to keep a hash table of points (i.e., price settings) and their corresponding profits. When a set of price parameters is selected, amoeba can compare its resulting profit to previous profits for these prices. If they differ, the consumer population is assumed to have changed. When amoeba determines that the landscape has changed, it expands the existing simplex by discarding the n lowest-profit points in the simplex and generating new, random points uniformly sampled from a cloud of radius r around the current best solution. The size of this radius can affect the performance of amoeba; if r is chosen to be too small, then all the new points are sampled from the immediate vicinity of the best point in the simplex, and amoeba will tend to stay near the previous optimum. On the other hand, a very large value of r can over-expand the simplex, requiring amoeba to go through several unnecessary contractions. Thus one would expect that as the resampling radius r is increased from zero, the performance of modified amoeba would improve up to a critical point beyond which amoeba's performance would degrade. The proper choice of r will therefore depend upon the magnitude of change in the consumer population and robustness of the chosen price schedule to change.

Table 2 compares the performance of our modified amoeba for different settings of the resampling radius r. Each element in the table shows the average profit per article per customer per iteration for a price schedule. The results are averaged over 100 runs, where each run consists of 1000 iterations with shocks occurring every 50 iterations. The table shows that for each of the three price schedules, the modified amoeba works best for intermediate values of r. Additionally, these values vary across price schedules. A larger r indicates that the optimum has moved farther between shocks, and so more adaptation is needed. Note that when r is set to zero, the modified amoeba is emulating the original amoeba, and thus, the first column in table 2 serves as a

[4]For a detailed survey on amoeba refer to Walters [14].

	Resampling radius r				
Pricing Schedule	0.0	0.5	1.0	2.0	5.0
Linear Pricing	0.7840	0.8818	0.8669	0.8620	0.8109
Pure Bundling	0.5989	0.8174	0.8146	0.8103	0.7414
Two-part Tariff	0.7758	1.0033	1.0061	0.9313	0.9047

Table 2: Average profit per article per customer per iteration (over 1000 iterations) for three price schedules as the resampling radius r is varied. In each run, shocks occur every 50 iterations and the producer has 50 iterations of ‘start-up’ time. The results are averaged over 100 runs.

benchmark to comparing the additional surplus a seller will receive when employing modified amoeba with different values of r to that gained with standard amoeba.

3.3 Experimental Results

In our previous work [5], we examined the problem of a monopolist producer learning the preferences of a static consumer population. We assumed that the producer had a limited number of chances to interact with this population and asked what price schedule this producer should choose, given that it wanted to maximize its aggregate profit. We found that two-part tariff tended to perform best; it was fairly easy to learn, and has a high potential profitability.

Implicit in these experiments was the idea that, after some period of time, the consumer population was going to drastically change, and so learning costs could not be absorbed into long-run profits. In this work, we make that assumption explicit and examine the profits accrued by a producer as the population changes.

As described in section 2.3, we model consumer change as a series of periodic shocks in which the consumer population is replaced. By varying the amount of time between each shock, we can alter the amount of time a producer has to learn about a new population. As described above, each consumer has a fixed $w = 10$, and k is drawn from $U[0, 0.7]$. We considered situations in which shocks occurred every 10, 50 and 250 iterations.

In conducting our experiments, we found that a producer had an easier time adapting to change if it was already ‘near’ an optimum. While the location of optima in the profit landscape will move between shocks, the large-scale structure of the landscape remains relatively constant, giving an advantage to a producer who has already had some time to interact with a population before it begins to change. To quantify this advantage, we tested each shock rate when a producer had 0, 50 and 500 iterations to interact with a static population *before* the shocks began. The results of these experiments (averaged over 100 runs) are presented in tables 2-4.

From observing the results of these experiments, two trends are immediately obvious. First, as predicted, every schedule extracts more profit as the time between shocks grows and the environment becomes more learnable. This is not surprising; fewer shocks means more time for a producer to exploit learned knowledge about the consumer population. Second, giving the producer a starting time to learn about a population before shocks begin improves performance. This gives amoeba time to find the approximate area of the profit landscape where optima tend to lie. This trend is not absolute; in fact, it varies depending upon the schedule. Giving linear pricing any sort of startup time improves its performance and makes it resistant to shocks; this implies that, after a shock, it is easy to move to the next linear price optimum, given that you were at the previous optimum.

With two-part tariff, on the other hand, there is not a clear benefit to having a startup time. One explanation for this is that the two-part tariff optima are easily found from a large number of locations in the profit landscape (this would also explain the fact that it extracts the highest average profit) and so a startup time adds no additional benefit.

Nonlinear pricing also seems to perform better when it had a long startup time. Empirically, we have seen that it can take nonlinear pricing at least 500 iterations to begin showing improvement when $N = 100$, so the startup time gives it a chance to locate favorable regions of the profit landscape.

Specifically measuring the magnitude of environmental change is a difficult task; developing precise metrics for this will require further work. Nevertheless, we can see that the change we have introduced in these experiments is sufficient to produce a qualititative difference from the results previously reported for

		Shock Rate	
Pricing Schedule	10	50	250
Linear Pricing	0.7033	0.8696	0.8782
Pure Bundling	0.7536	0.8566	0.9762
Two-part Tariff	0.9220	0.9934	1.0491
Mixed Bundling	0.8245	0.9001	0.9664
Block Pricing	0.7809	0.9060	1.0231
Nonlinear Pricing	0.3279	0.4155	0.5279

Table 3: Average profit per article per customer per iteration (over 1000 iterations) for each price schedule when shocks occur every 10, 50 and 250 iterations, respectively. The producer has 0 iterations of ‘start-up’ time in this case.

		Shock Rate	
Pricing Schedule	10	50	250
Linear Pricing	0.8474	0.8685	0.8871
Pure Bundling	0.7650	0.8713	0.9687
Two-part Tariff	0.9144	1.0203	1.0533
Mixed Bundling	0.8239	0.9210	0.9628
Block Pricing	0.7911	0.9275	1.0046
Nonlinear Pricing	0.3735	0.4296	0.5736

Table 4: Average profit per article per customer per iteration (over 1000 iterations) for each price schedule when shocks occur every 10, 50 and 250 iterations, respectively. The producer has 50 iterations of ‘start-up’ time in this case.

		Shock Rate	
Pricing Schedule	10	50	250
Linear Pricing	0.8492	0.8716	0.8866
Pure Bundling	0.7526	0.8730	0.9661
Two-part Tariff	0.9247	1.0023	1.0959
Mixed Bundling	0.8444	0.9110	0.9692
Block Pricing	0.8127	0.9029	1.0338
Nonlinear Pricing	0.4127	0.4696	0.6245

Table 5: Average profit per article per customer per iteration (over 1000 iterations) for each price schedule when shocks occur every 10, 50 and 250 iterations, respectively. The producer has 500 iterations of ‘start-up’ time in this case.

a static population [5]. In that experiment, nonlinear pricing would *eventually* dominate once all schedules were learned perfectly; it just took a very long time to make up for profits foregone due to learning. In these experiments, nonlinear pricing will never dominate; the world changes too much and too frequently. This tells us that the magnitude of change is sufficient to make overly complex schedules unattractive to a learning producer.

We can also use these experiments to compare the performance of the different schedules. As noted above, two-part tariff yields the highest average profit, since it is easily learned and captures a large percentage of the available profit. When the population changes rapidly, linear pricing outperforms pure bundling, but as the shocks slow and the environment becomes more learnable, pure bundling overtakes linear pricing. The reason for this is that the pure bundling landscape contains a number of widely spaced local optima. Amoeba requires several iterations to move through these. Finally, mixed bundling performs similarly to pure bundling, even though theoretically it should extract the same profit as two-part tariff. This is due to the presence of optima in which all consumers purchase the bundle or they all purchase per-article. When the population changes, if amoeba finds itself near one of these optima, it will converge to it and not necessarily explore the rest of the landscape. In contrast, the two-part tariff landscape has a single hill with optima located on a 'mesa' at the top of this hill.

4 Conclusion

In this paper, we have explored a simple model of consumer nonstationarity and compared the profit extracted by different price schedules in this nonstationary environment. We have emphasized the tradeoff between exploration and exploitation, and shown how the complexity of different pricing schedules and the learnability of the profit landscape can both affect a producer's aggregate profit. Additionally, we have shown that the point(s) at which a producer begins searching for optima can have a large effect on the cumulative profit; if a producer has a 'startup' time to find regions of the profit landscape which have a nonzero gradient, the performance when using amoeba improves. As in our previous work [5], two-part tariff has been shown to be an attractive choice of price schedule, due to its low complexity and robustness to changing consumer populations.

This work has only scratched the surface of learning in a nonstationary environment. We plan to extend it in several ways. First, we would like to explore more complex models of nonstationarity, including continuous change, as well as evolutionary models in which consumer replacement is tied to their satisfaction. This also gives us an avenue to introduce learning on the part of the consumers.

Competition is an essential next step. Our previous work [4] examined a scenario in which producers learn to discover niches in an unknown but static population. Extending this to examine niche formation and discovery in a dynamic world is a promising extension.

Finally, we would like to extend the current model of consumer preferences. While the Chuang-Sirbu model is useful for its tractability, it does not have the richness one might like. For example, it doesn't consider substitutes or complements, and it assumes that the producer can show a set of N goods to the consumer, who then picks the ones she likes. If, instead, the producer must select these goods based on a prediction of the consumer tastes, the problem becomes much more complex.

References

[1] Philippe Aghion, Patrick Bolton, Christopher Harris, and Bruno Jullien. Optimal learning by experimentation. *Rev. Econ. Stud.*, 58(4):621–654, 1991.

[2] Y. Bakos and E. Brynjolfsson. Bundling information goods: Pricing, profits and efficiency. In D. Hurley, B. Kahin, and H. Varian, editors, *The Economics of Digital Information Goods*. MIT Press, Cambridge, Massachusetts, 1999.

[3] C. H. Brooks and E. H. Durfee. Decision-theoretic learning of agent models in an information economy. In *Proceedings of the 2001 AAAI Spring Symposium on Game-theoretic and Decision-theoretic Agents*, 2001.

[4] C. H. Brooks, E. H. Durfee, and R. Das. Price wars and niche discovery in an information economy. In *Proceedings of ACM Conference on Electronic Commerce (EC-00)*, Minneapolis, MN, October 2000.

[5] C. H. Brooks, S. Fay, R. Das, J. K. MacKie-Mason, J. O. Kephart, and E. H. Durfee. Automated strategy searches in an electronic goods market: Learning and complex price schedules. In *Proceedings of ACM EC-99*, pages 31–40, 1999.

[6] J. C. Chuang and M. A. Sirbu. Network delivery of information goods: Optimal pricing of articles and subscriptions. In D. Hurley, B. Kahin, and H. Varian, editors, *The Economics of Digital Information Goods*. MIT Press, Cambridge, Massachusetts, 1999.

[7] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.

[8] A. A. Fe'ldbaum. *Optimal Control Systems*. Academic Press, New York, 1965.

[9] Sanford J. Grossman, Richard E. Kihlstrom, and Leonard J. Mirman. A Bayesian approach to the production of information and learning by doing. *Rev. Econ. Stud.*, 44(3):533–547, 1977.

[10] Godfrey Keller and Sven Rady. Optimal experimentation in a changing environment. *Rev. Econ. Stud.*, 66:475–507, 1999.

[11] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[12] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[13] Sebastian B. Thrun and Knut Møller. Active exploration in dynamic environments. In J.E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*. Morgan Kaufmann, San Mateo, CA, 1992.

[14] F. H. Walters, L. R. Parker, S. L. Morgan, and S. N. Deming. *Sequential Simplex Optimization*. CRC Press, Boca Raton, Florida, 1991.

[15] Robert B. Wilson. *Nonlinear Pricing*. Oxford University Press, 1993.