Pesquisa Operacional (2013) 33(1): 37-61

Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope

DESIGN TO LEARN: CUSTOMIZING SERVICES WHEN THE FUTURE MATTERS

Dan Ariely[1], Gabriel Bitran[2] and Paulo Rocha e Oliveira[3*]

Received April 25, 2012 / Accepted May 10, 2012

ABSTRACT. Internet-based customization tools can be used to design service encounters that maximize customers' utility in the present or explore their tastes to provide more value in the future, where these two goals conflict with each other. Maximizing expected customer satisfaction in the present leads to slow rates of learning that may limit the ability to provide quality in the future. An emphasis on learning can lead to unsatisfied customers that will not only forego purchasing in the current period, but, more seriously, never return if they lose trust in the service provider's ability to meet their needs. This paper describes service design policies that balance the objectives of learning and selling by characterizing customer lifetime value as a function of knowledge. The analysis of the customization problem as a dynamic program yields three results. The first result is the characterization of customization policies that quantify the value of knowledge so as to adequately balance the expected revenue of present and future interactions. The second result is an analysis of the impact of operational decisions on loyalty, learning, and profitability over time. Finally, the quantification of the value of knowing the customer provides a connection between customer acquisition and retention policies, thus enhancing the current understanding of the mechanisms connecting service customization, value creation, and customer lifetime value.

Keywords: Service customization, customer learning, customer lifetime value, dynamic programming.

1 INTRODUCTION

Electronic interfaces for service delivery present unprecedented opportunities for customization. Companies can identify returning customers at almost no cost and present each of them with a unique interface or service offering over a web page, voice interface, or mobile device. Customization is important in these interactions because the variety of products, services and information that companies can offer can be enormous and customers are often not able to find and identify the alternative that best suits their preferences. More generally, the amount of information available over the Internet is overwhelming and the costs to search and evaluate all potentially relevant pieces of information can be prohibitive.

*Corresponding author
[1]Duke University – Fuqua School of Business, 2024 W. Main Street, Durham, NC 27705. E-mail: dandan@duke.edu
[2]MIT – Sloan School of Management, 77 Massachusetts Avenue, Cambridge, MA 02042, USA.
E-mail: gbitran@mit.edu
[3]IESE Business School, Avinguda Pearson 21, 08034 Barcelona, Spain. E-mail: paulo@iese.edu

This situation can lead to interactions of low net utility to customers. Hence the need for software applications generally called intelligent agents (or smart agents) that present customers with customized recommendations from large sets of alternatives. These agents have the potential to create great benefits for both consumers and companies. It is certainly desirable for customers to reduce search costs and by going directly to a company that already knows them and gives them high utility. At the same time, companies that learn about their customers can use this knowledge to provide increasingly better service as the relationship progresses, making it difficult for competitors to "steal" their customers.

In order to fully reap the benefits of customization, companies must overcome two obstacles. First, the strategies that maximize selling (or their customers' utility in the current period) do not coincide with strategies that maximize learning. Second, the cost of making bad recommendations (and learning form them) might be prohibitive because customers rarely forgive and return to service providers that make bad recommendations. Overcoming these obstacles depends on the agents' ability to adequately balance the goals of learning at the expense of selling and selling at the expense of learning.

In addition to being an important tool in the reduction of search costs, smart agents are also cost-efficient. Consequently, as more and more services are delivered through automated interfaces, smart agents have been playing increasingly important roles in everyday decision-making and are gaining considerable more autonomy to act on the consumers' behalf (Maes [1995] documents the early days of this tendency).

The essential task of learning about customers in order to provide better recommendations has been made more difficult in many ways as interfaces change from human-human to human-computer. Human intermediaries can, in many cases, observe their clients and make a number of inferences on the basis of their interactions. The lack of cues typical of the personal encounter diminishes the opportunities for correcting incomplete advice or wrong guesses. Software agents only register click patterns, browsing behavior, and purchase histories. These agents can only learn about customers by asking them questions and by observing their behavior. The issues related to the benefits and limitations of asking questions have been adequately addressed elsewhere: clients may not be willing nor have the patience to answer questions (*e.g.*, Toubia *et al.*, 2003), and their replies are not always truthful for various (and sometimes intentional) reasons (White, 1976). Regardless of how they answer surveys, customers spend their time and money in the activities from which they derive the most utility. In the offline world, Rossi *et al.* (1996) found that the effectiveness of a direct marketing campaign was increased by 56% due to the customization of coupons based on observations of one single purchase occasion. Intelligent agents can learn about customers by observing the purchases they make. Furthermore, firms interacting with customers through automated interfaces can also lean by observing the products that customer reject. Therefore, companies must strive to learn about their customers by observing how they react to the recommended products and services to which they're exposed.

Intelligent agents face a dilemma: they must either sell at the expense of learning or learn at the expense of selling. More precisely, they can either make recommendations that they expect

their customers to like and learn about their customers' preferences at slow rates or they can take more risks by suggesting products their customers might not accept and learn at higher rates. The agent's dilemma captures the essence of the selling versus learning trade-off faced by companies that customize services on the Internet. Observing an action that was already expected to happen only reconfirms something one already knew or suspected to be true. But should an agent recommend a product with no prior knowledge of how their customers would react? By taking this course of action, agents may learn a lot about their customers' responses to recommendations of such products, regardless of whether or not the item is purchased. However, they run the risk of causing an unsuccessful service encounter that may lead to defection.

Customers who receive very bad recommendations can lose trust in the smart agent's ability to help them make better decisions and never return. Research in psychology and marketing has repeatedly shown the importance of trust as the basis for building effective relationships with customers in e-commerce as well as in regular commerce (*e.g.*, Kollock, 1999; Lewicki *et al.*, 1998; Ariely *et al.*, 2004). Doney & Cannon (1997) established an important link between trust and profitability by noting the importance of trust to build loyalty. Low trust leads to low rates of purchase, low customer retention and, consequently, meager profits. Studies on human and non-human agents have extended the findings of previous research by showing that trustworthy agents are more effective, *i.e.*, they are better at engaging in informative conversation and helping clients make better decisions. Urban *et al.* (2000) have also shown that advice is mostly valued by less knowledgeable and less confident buyers, and those are exactly the same customers less likely to trust external agents to act as surrogates on their behalf. The human element involved in company-customer relationships makes it imperative that firms strive to build and maintain trust if they are to maintain profitable customers in the long run. The question is how these principles can be operationalized in the strategies adopted by agents.

Intelligent agents must strive to be trustworthy in two ways. First, customers must trust that agents have their best interest in mind. This will be true whenever the agent's payoffs are proportional to each customer's utility. Second, the customer must trust that the agent is good. Models of online consumer behavior must take into account the fact that if the agent makes a very bad recommendation the consumer loses trust and may never come back. The consequences of giving bad advice are much worse for software agents than for human agents. Ariely *et al.* (2004) conducted a series of experiments to compare trust online and offline and found that even though customers are sometimes willing to forgive a mistake made by a human agent, they rarely do so for a software agent. In one of his experiments, half of the subjects received financial advice from a software agent, and the other half from a human agent. Both agents were manipulated so that they made exactly the same serious mistake. After realizing they had received bad advice, customers of the human agent were much more likely to continue the relationship than customers of the software agent. People forgive people but they don't forgive software. Turning off the computer or clicking on a different website entails much less psychological costs than terminating a personal relationship. Consequently software agents face much higher churn rates than their human counterparts. The levels of tolerance for incomplete, wrong or misleading

advice are lower, yet the electronic agent must rely on much more precarious information and cues regarding consumers' tastes and preferences.

The customization policy used by the agent affects the customer lifetime value (CLV) in two ways. CLV is typically defined as the net present value of the stream of cash flows the firm expects to receive over time from a given customer (*e.g.*, Berger & Nasr, 1998; Jain & Singh, 2002). The customization policy determines expected quality, which in turn influences (1) the customer defection decision and (2) the expected revenue per interaction. The model analyzed in this paper does not make any assumptions about the retention rate and profitability as the relationship progresses. These parameters are determined by the business setting and the manager's decisions. This flexibility is an important contribution of this paper, as it presents a tool that can accomodate situations where profits accelerate over time as well as those when profits do not accelerate over time.

The next section of this paper formalizes the customer behavior model, describes the dynamics of customer/company interactions, and formulates the agent's customization problem as a dynamic program. Section 3 establishes the optimality conditions and solution procedures for the model introduced in Section 2. Section 4 discusses the significance of the results, paying close attention to their meaning in the context of CLV research. Finally, Section 5 offers some directions for further research.

2 MODEL DESCRIPTION

This section is divided into three parts. The first one presents a mathematical choice model that captures the essential behavioral features of the choice process customers go through when interacting with software agents. This is followed by a description of a dynamic system used to model the interactions between customers and companies over time. The third part shows how the company's decision of how to customize products and services can be framed in terms of a Markov Decision Process using the customer behavior model of Section 2.1 and the dynamical system of Section 2.2.

2.1 Customer Behavior

This section describes a customer behavior model that captures the ways in which customers react to recommendations of products and services made by software agents that is consistent with the behavior qualitatively described in the psychology and marketing literature (*e.g.*, Zauberman, 2000; Ariely *et al.*, 2004). Customer observe the true quality of the recommendation, plus or minus an error term. The probability that the customer will accept the recommendation, reject the recommendation, or leave the company is defined through a random utility model (*e.g.*, McFadden, 1981; Meyer & Kahn, 1990). According to random utility theory, when a customer is offered product X the utility that is actually observed has two components: a deterministic component $u(X)$, which corresponds to the true underlying utility and an error term e. The error term accounts for factors such as unobserved attributes, imperfect information, or

measurement errors that make the customer unable to determine the exact utility of a product upon initial examination. Formally, this can be expressed as

$$\hat{u}(X) = u(X) + \varepsilon$$

where $\hat{u}(X)$ is the observed utility.

Once customers evaluate the utility of X, the service they are being offered, they can take one of three actions:

- Accept product X if $\hat{u}(X) > 0$
- Reject product but come back next time if $-c < \hat{u}(X) < 0$
- Leave the company and never come back if $\hat{u}(X) < -c$

Customers who observe a very low utility ($-c$, which can be interpreted as a switching cost) immediately lose trust in the agent as a reliable advisor and never return. In the case of services provided over smartphone apps, this can be interpreted as unistalling the app. The recommendation is accepted if the observed utility exceeds a given threshold, which can be assumed to be 0 without loss of generality. Customers who accept the suggested product or service are satistfied and return the next time they require service. If a recommendation is barely below the acceptance level, customers will purchase elsewhere (or forego purchasing in the current period) but return in the future. The cost of not making a sale by experimenting with different customization policies is captured by the probability that the observed utility will fall in the interval $[-c, 0]$.

The probabilities of accepting, leaving, or rejecting a product X (denoted $p^A(X)$, $p^L(X)$, and $p^R(X)$, respectively) can be computed by defining the distribution of ε. If ε is normally distributed with mean 0 and variance σ^2, the parameter σ can be altered to represent different types of products or business settings, where the customer observes products with more or less error. This model corresponds to the ordered Probit model, and its properties are thoroughly discussed in Greene (2000). The general formulation of the choice probabilities is shown in the first column of (1), and the specific form they take when $\varepsilon \sim N(0, \sigma)$ is shown in the second column (where Φ represents the cumulative normal function). Figure 1 depicts the probabilities taking each of the three possible actions when the customer is offered a recommendation with a deterministic utility component u. In this particular case, u is below the acceptance threshold, but since the observed utility is $u + \varepsilon$ the probability that the product will be accepted is positive.

$$\begin{array}{lll} \text{Action} & \text{Probability of Action} & \text{Probability if } \varepsilon \sim N(0,\sigma) \\ \text{Accept} & p^A(X) = \Pr(u(X) \geq 0) & = \Phi\left(\frac{u(X)}{\sigma}\right) \\ \text{Leave} & p^L(X) = \Pr(u(X) < -c) & = \Phi\left(-\frac{u(X)+c}{\sigma}\right) \\ \text{Reject} & p^R(X) = \left[1 - p^A(X) - p^L(X)\right] & = 1 - \Phi\left(\frac{u(X)}{\sigma}\right) - \Phi\left(-\frac{u(X)+c}{\sigma}\right) \end{array} \quad (1)$$

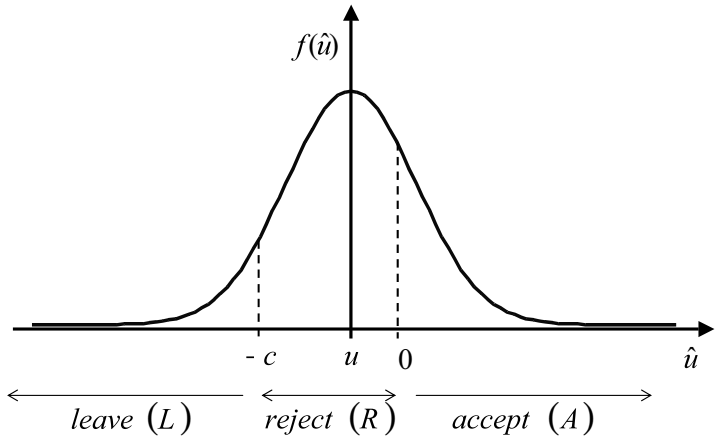

Figure 1 – Probabilities of Accepting, Rejecting and Leaving when offered a product with deterministic utility u.

2.2 Dynamics of Customer-Company Interactions

Figure 2 describes the dynamics of customer/company interactions. Since the interactions take place through an automated interface controled by a smart agent, the words "company" and "agent" are used interchangeably. This system is the basis of the formulation of the agent's dilemma as a Markov decision problem.

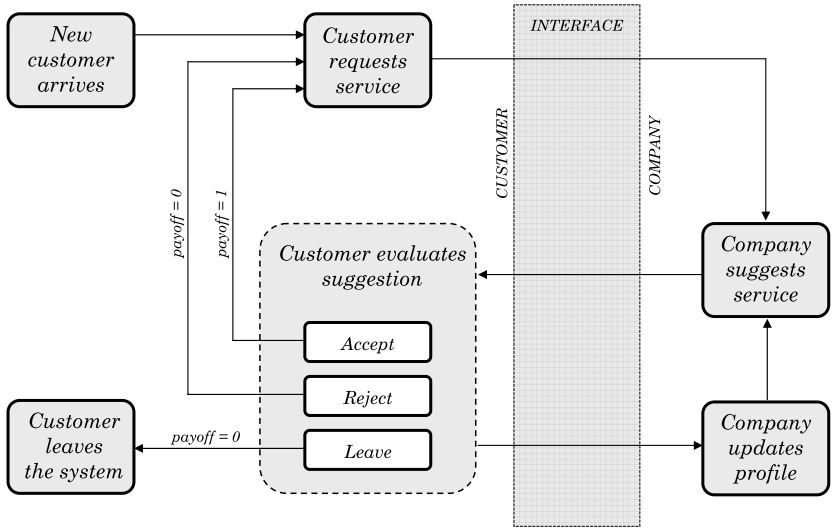

Figure 2 – The dynamics of customer-company interactions.

Each customer belongs to a segment $S_i \in \mathbf{S}$. The company does not know the segment to which the new customer belongs, but has prior beliefs. These beliefs are represented by a $|\mathbf{S}|$-dimensional vector $b^0 \in \mathbf{B}$, with each component corresponding to the probability that the customer belongs to a given segment. If S^* denotes the true segment to which the customer belongs,

the i'th component of the prior belief vector is $b_i^0 = \Pr(S^* = S_i) \in [0, 1]$. Since all customers must belong to some segment, $\sum_{i=1}^{|S|} b_i^0 = 1$. The learning problem faced by the agent is the assignment of an individual customer to one of the | S | segments.

Each segment S_i is defined by a utility function $u_i(X)$ that maps products and services X into real numbers. One way to define these utility functions is by associating each segment S_i with a vector β_i. The components of β_i are the weights a customer of segment i gives to each attribute of the products in $\mathbf{X}$. The corresponding utility function could be defined as $u_i\left(X_j\right) = \left(\beta_i' \cdot X_j\right)$, where $u_i\left(X_j\right)$ denotes the utility that product X_j has to a customer of segment S_i.

When the customer requests service, the company chooses a product from the set $\mathbf{X} = \{X_1, X_2, \ldots, X_{|\mathbf{X}|}\}$. $\mathbf{X}$ is a set of substitute products or services, and can be thought of as the different ways in which a service can be customized. Each $X_i \in \mathbf{X}$ is a vector whose components correspond to different levels of the attributes of the product or service. The recommendation made during the t'th interaction is denoted X^t. The decision problem of which service configuration should be chosen is directly addressed as an optimization problem in Section 2.3.

The customer will either accept the recommendation, reject the recommendation, or leave, as explained in the beginning of this section. The probabilities with which each of these actions take place are given by (1). The company will observe the customer's action as described in Figure 2. The observation made during the t'th interaction is denoted θ^t, and the set of possible observations is $\Theta = \left\{\theta^A, \theta^R, \theta^L\right\}$, corresponding to accepting the recommendation, rejecting the recommendation, and leaving the system. For example, $\theta^5 = \theta^A$ means that the customer accepted the product offered during the fifth interaction.

After each observation, beliefs are updated in light of the previous belief and the action/observation pair of the last interaction. Bertsekas (1995a) proved that the last belief, b^{t-1} is a sufficient statistic for $\left(b^0, X^1, \theta^1, X^2, \theta^2, ..., X^{t-1}, \theta^{t-1}\right)$. Thus, if b_i^t denotes the i'th component of the belief vector after t interactions, then

$$b_i^t = \Pr\left(S^* = S_1 | b^{t-1}, X^t, \theta^t\right).$$

The new belief can be computed using

$$\Pr\left(S^* = S_i | b^{t-1}, X^t, \theta^t\right) = \frac{1}{\Pr\left(\theta^t | b^{t-1}, X^t\right)} \cdot \left[b_i^{t-1} \cdot \Pr\left(\theta^t | S_i, X^t\right)\right]$$

where

$$\Pr\left(\theta^t | b^{t-1}, X^t\right) = \sum_{i=1}^{|\mathbf{S}|} \left[b_i^{t-1} \cdot \Pr\left(\theta^t | S_i, X^t\right)\right] \tag{2}$$

is a normalizing factor. The update function φ can then be defined by

$$\varphi(b, \theta, X) = \frac{1}{\Pr(\theta | b, X)} \cdot \begin{pmatrix} b_1 \cdot \Pr(\theta | S_1, X) \\ b_2 \cdot \Pr(\theta | S_2, X) \\ \ldots \\ b_{|\mathbf{S}|} \cdot \Pr\left(\theta | S_{|S|}, X\right) \end{pmatrix}. \tag{3}$$

2.3 Optimization Problem

The company earns a payoff of 1 if the customer accepts the recommendation and 0 otherwise. Since increasing the utility always increases the probability of purchase, the incentives of the customer and the company are perfectly aligned. It is always in the company's best interest to please the customer, since the more utility the company provides to the customer, the higher the payoff. The expected payoff of suggesting product X_j to a customer from segment S_i is given by

$$R\left(S_i, X_j\right) = 1 \cdot p_i^A\left(X_j\right) + 0 \cdot p_i^R\left(X_j\right) + 0 \cdot p_i^L\left(X_j\right) = p_i^A\left(X_j\right) \tag{4}$$

where $p_i^A\left(X_j\right)$ is the probability of purchase as defined in (1).

If the company knows the customer's true segment, the problem of making recommendations reduces to always suggesting the product with the highest expected utility for that segment. This is a Markov chain model, which can be easily extended to other models such as those of Pfeifer & Carraway (2000) through state augmentation.

The problem of making recommendations is not as simple if the company does not know the customer's segment. The real decision is to find the best recommendation for each possible belief so as to maximize the expected revenue (or customer's utility). This policy, denoted μ^* must solve the equation

$$\mu^* = \arg\max_{X \in \mathbf{X}} \left\{ E\left(\sum_{t=0}^{\infty} \gamma^t r_t \right) \right\} \tag{5}$$

where $\gamma \in (0, 1)$is a discount factor and r_t is the reward at time t. The set of possible beliefs is $[0, 1]^{|\mathbf{S}|}$, which is uncountably infinite. This stands in sharp contrast with customer migration models, where the possible states of the system are defined by a finite (or countably infinite) set.

In the belief-state problem, the payoff function maps belief states and actions to revenues. Therefore, the payoff function $\tilde{R}$ is defined by

$$\tilde{R}\left(b, X\right) = \sum_{i=1}^{|\mathbf{S}|} b_i \cdot R\left(S_i, X_j\right). \tag{6}$$

The three possible transitions from any given belief state are given by

$$\Pr\left(b^t = \hat{b} | b^{t-1}, X^t\right) = \begin{cases} \sum\limits_{i=1}^{|\mathbf{S}|} b_i^{t-1} \cdot p_i^A\left(X^t\right) & \text{if } \hat{b} = \varphi\left(b^{t-1}, \theta^A, X^t\right) \\ \sum\limits_{i=1}^{|\mathbf{S}|} b_i^{t-1} \cdot p_i^R\left(X^t\right) & \text{if } \hat{b} = \varphi\left(b^{t-1}, \theta^R, X^t\right) \\ \sum\limits_{i=1}^{|\mathbf{S}|} b_i^{t-1} \cdot p_i^L\left(X^t\right) & \text{if } \hat{b} = \varphi\left(b^{t-1}, \theta^L, X^t\right) = L \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where L corresponds to the state when the customer has left the system. Figure 3 depicts the dynamics of the belief-state problem for the two-segment case.

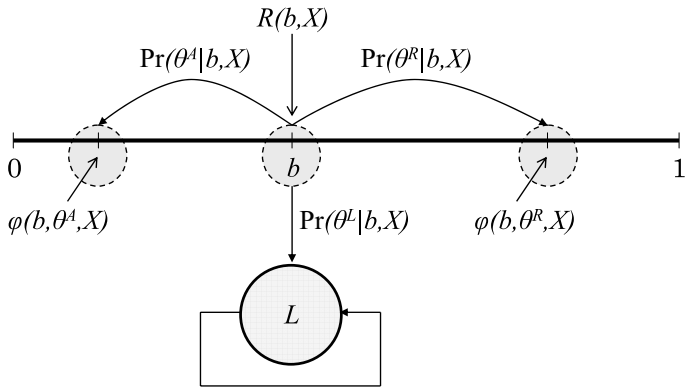

Figure 3 – Transition probabilities and payoffs in the partially observable stochastic shortest path problem.

The optimal value function $V^*(b) = \max E\left(\sum_{t=0}^{\infty} \gamma^t r_t\right)$ satisfies the Bellman equation

$$V^*(b) = \max_{X \in \mathbf{X}} \left\{ \tilde{R}(b, X) + \gamma \sum_{b' \in \mathbf{B}} P\left(b'|b, X\right) V^*\left(b'\right) \right\}. \tag{8}$$

We know from (8) that b' can only take three possible values, namely $\varphi\left(b, \theta^A, X\right)$, $\varphi\left(b, \theta^R, X\right)$, and $\varphi\left(b, \theta^L, X^t\right)$. Furthermore, since customers have no value after they leave $V^*(L) = 0$.

Therefore, (9) can be written as

$$V^*(b) = \max_{X \in \mathbf{X}} \left\{ \tilde{R}(b, X) + \gamma \sum_{\theta \in \{\theta^A, \theta^R\}} P(\theta|b, X) V^*(\varphi(b, \theta, X)) \right\}, \tag{9}$$

The dynamic programming mapping corresponding to the above value function is

$$(TV)(b) = \max_{X \in \mathbf{X}} \left\{ \tilde{R}(b, X) + \gamma \sum_{\theta \in \{\theta^A, \theta^R\}} P(\theta|b, X) V^*(\varphi(b, \theta, X)) \right\} \tag{10}$$

This mapping is a monotone contraction under the supremum metric. Therefore, repeated application of the operator T to a value function V converges to the optimal value function (*e.g.*, Bertsekas, 1995b). This result is the basis for the value iteration algorithm, which consists of repeatedly applying the operator T to the value function V^1. Letting $V^n = T^n V$, *i.e.*, n applications of T to V^1, then the algorithm is stopped when $|V^*(b) - V^n(b)|$ is within a tolerable error margin, which can be expressed as a function of $|V^*(b) - V^{n-1}(b)|$ (this is an application of a standard result found in Littman [1994] or Bertsekas [1995b], for example).

One important problem must be solved before value iteration can be used to find the optimal solution: there is an infinitely large number of beliefs. The continuity of the belief space could

mean that there exists some n for which V^n cannot be represented finitely or that the value function updates cannot be computed in a finite amount of time. The next sections describes how these difficulties can be overcome in order to solve the agent's dilemma.

3 MODEL ANALYSIS AND SOLUTION

The value function after n iterations of the dynamic programming mapping is piecewise linear convex function and can be written as

$$V^n(b) = \max_{a \in A^n} (b \cdot \alpha)$$

where A^n is a finite set of $|\mathbf{S}|$-dimensional vectors. This implies that $V^*(b)$ can be approximated arbitrarily well by a piecewise linear convex function.

Proposition 1. *The value function $V^n(b)$, obtained after n applications of the operator T to the value function $V^1(b)$, is piecewise linear and convex. In particular, there exists a set A^n of $|\mathbf{S}|$-dimensional vectors such that*

$$V^n(b) = \max_{a \in A^n} (b \cdot \alpha)$$

Proof. The proof is by induction, and therefore consists of the verification of the base case and of the induction step.

(1) Base case: $V^1(b) = \max_{\alpha \in A^1} (b \cdot \alpha)$.

(2) Induction step: If

$$\exists A^{n-1} \text{ such that } V^{n-1}(b) = \max_{\alpha \in A^{n-1}} (b \cdot \alpha)$$

then

$$\exists A^n \text{ such that } V^n(b) = \max_{\alpha \in A^n} (b \cdot \alpha).$$

The statement above can be proved by verifying that the elements $\alpha \in A^n$ are of the form

$$\alpha = p^A(X) + \gamma \Gamma^A(\alpha_i) + \gamma \Gamma^R(\alpha_j) \tag{11}$$

where $\alpha_i \in A^{n-1}$, $\alpha_j \in A^{n-1}$ and Γ^A and Γ^R matrices with elements

$$\Gamma^A_{ij}(X) = \begin{cases} p^A_i(X) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

and

$$\Gamma^R_{ij}(X) = \begin{cases} p^R(X) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}.$$

It then follows that

$$V^n(b) = \max_{\alpha \in A^n} (b \cdot \alpha)$$

which can be verified to be true by defining

$$A^n := \bigcup_{X \in \mathbf{X}} \left\{ \bigcup_{\alpha^i \in \mathbf{A}^{n-1}} \left[\bigcup_{\alpha^j \in \mathbf{A}^{n-1}} \left(p^A(X) + \gamma \Gamma^A(X) \cdot \alpha^i + \gamma \Gamma^R(X) \cdot \alpha^j \right) \right] \right\}.$$

The details of the proof are in Appendix A.1. □

Figure 4a shows an example of the convex hull of a hypothetical set A^n consisting of four vectors and Figure 4b depicts the corresponding value function $V^n(b)$. Each vector corresponds to a different customization policy. On a more intuitive level, each component of the vector is the expected payoff the agent would receive given a sequence of recommendations and a customer segment. The optimal policy will be a function of the belief (represented by the slope of the dotted line in Figure 4a) and the set of feasible policies. Each piecewise-linear segment of the value function 4b corresponds to a different policy.

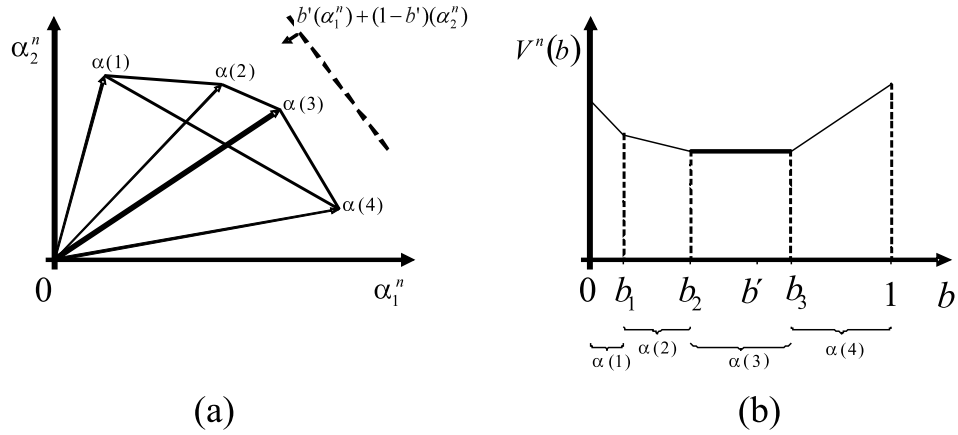

Figure 4 – A set of A^n vectors and their corresponding value function.

Theorem 1 provides the missing link to ensure that the value iteration algorithm is certain to arrive at the optimal solution for the finite-horizon problem and an arbitrarily good approximation for the infinite horizon problem in a finite amount of time. Unfortunately, this finite amount of time can be very large and often impractical. The difficulty lies in the efficient computation of the sets A^n since the size of these sets grows very fast with each step of the value iteration algorithm. In each iteration of the algorithm there is one new α vector for each possible product recommendation X for each permutation of size 2 of the vectors in A^{n-1}. Consequently, $|A^n| = |\mathbf{X}|^{2^n-1}$.

The way in which the sets A^n are generated make it impossible to determine the optimal value function in closed-form, and the rate at which they grow can make numerical solutions impractical. Figure 5 show the convex hull of a sequence of sets A^n for increasingly large n. Fortunately,

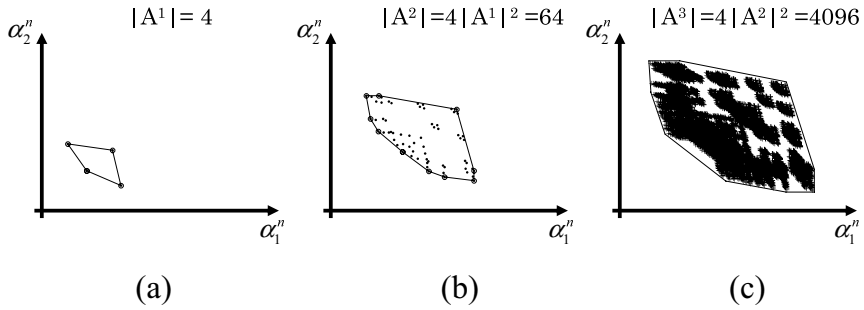

Figure 5 – Generating A^n sets from A^{n-1}.

this property is shared by all partially observable Markov Decision Problems (POMDPs), and the important developments in the literature have generated methods to (i) make the sets A^n as small as possible so as to minimize the size of A^{n+1} or (ii) find ways of quickly generating the sets A^n from sets A^{n-1}. Sondik (1978) made significant theoretical contributions to the understanding of the POMDP model and described the first algorithm for finding the optimal solution. Monahan (1982) formulated a linear program that finds the smallest possible set A^n that contains all the vectors a used in the optimal solution. Littman's (1994) Witness algorithm is one of the most important contributions to the generation of A^n from A^{n-1}. Applications in robot navigation resulted in a recent increase of interest in POMDPs in the artificial intelligence community (*e.g.*, Kaelbling *et al.*, 1996) which led to very important contributions in computational methods. Littman *et al.* (1995), for example, describe methods to solve POMDPs with up to nearly one hundred states. Tobin (2008) and Ross *et al.* (2008) provide a recent review of the state of the art in this field.

Figure 6 illustrates the results of numerical studies comparing the optimal policy μ^*, defined in (6) and computed using the Witness algorithm, and the myopic policy μ^M, which satisfies

$$\mu^M(b) = \arg\max_{X \in \mathbf{X}} \left\{ \tilde{R}(b, X) \right\}. \tag{12}$$

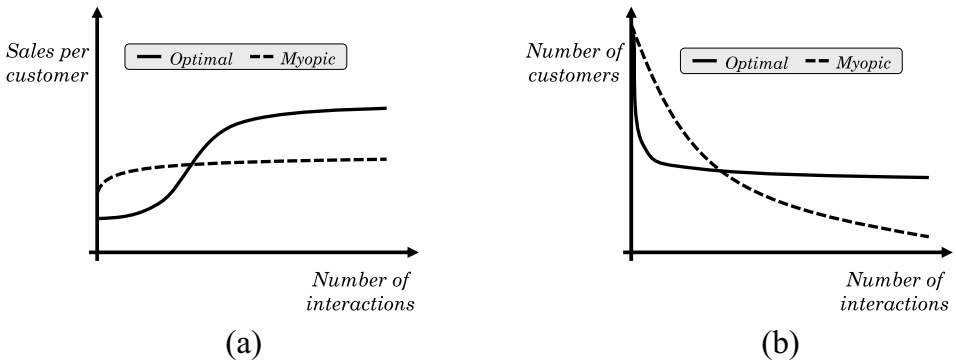

Figure 6 – Comparison of optimal and myopic customization policies.

These results were obtained from simulations where companies started out with the same number of customers and gained no new customers over time. The graphs are typical of simulations done with varying numbers of customer segments and products. Graph 5(a) shows that the myopic policy is more likely to generate profits in the short-run, while the optimal policy does better in the long-run. The agent that acts optimally sacrifices sales in the beginning to learn about the customers to serve them better in the future. Graph 5(b) shows the cost of learning in terms of the customer base. The optimal policy leads to more customer defections due to low utility in the short-run, but the customers who stay receive consistently high utility in later periods and defect at much slower rates than customers being served by a myopic agent.

In spite of significant recent advances in computational methods, the optimal policy cannot always be computed in a reasonable amount of time. Hence the need for approximate solution methods that produce good policies in an efficient manner. If $V^*(b)$ can be approximated by $\tilde{V}(b)$, then the corresponding policy $\tilde{\mu}(b)$ satisfies

$$\tilde{\mu} = \arg\max_{X \in \mathbf{X}} \left\{ \tilde{R}(b, X) + \gamma \sum_{\theta \in \Theta} P(\theta | b, X) \, \tilde{V}(\varphi(b, \theta, X)) \right\}. \tag{13}$$

There are several ways to approximating $V^*(b)$. Bertsekas (1995b) discusses traditional approaches, and Bertsekas & Tsitsiklis (1996) suggest alternative methods based on neural networks and simulations. The paragraphs that follow describe the Limited Learning (LL) approximation. This approximation is of interest because its corresponding customization policy yields good results and can be stated in simple terms which give insights into relevant managerial settings.

Limited learning policies are derived by assuming that the agent will stop learning after a fixed number of interactions. This restriction in the agent's actions implies that the same product will always be recommended after a given point. These approximations are very good in situations where most of the learning occurs in the early stages.

Proposition 2 derives an expression for the policy $\mu^{LL}(b)$, which maximizes profits when agents are not allowed to change their beliefs between interactions.

Proposition 2. *If agents are constrained to recommend the same product in all interactions with customers, then the infinite-horizon value function is given by*

$$V^{LL}(b) = \max_{X \in \mathbf{X}} \left\{ \sum_{i=1}^{|\mathbf{S}|} b_i \cdot \left(\frac{p_i^A(X)}{1 - \gamma \left(1 - p_i^L(X)\right)} \right) \right\}. \tag{14}$$

and the corresponding policy by

$$\mu^{LL}(b) = \arg\max_{X \in \mathbf{X}} \left\{ \sum_{i=1}^{|\mathbf{S}|} b_i \cdot \left(\frac{p_i^A(X)}{1 - \gamma \left(1 - p_i^L(X)\right)} \right) \right\}. \tag{15}$$

Proof. See Appendix A.2. □

The $\mu^{LL}(b)$ policy can be re-written as

$$\mu^{LL}(b) = \arg\max_{X \in \mathbf{X}} \left\{ b' \cdot \Lambda(X) \cdot p^{A}(X) \right\}$$

where $\Lambda(X)$ is a diagonal matrix that can be interpreted as the inverse of the risk associated with each recommendation for each segment. There are many ways to choose the elements of $\Lambda(X)$. In the particular case of (18), the risk is fully captured by the probability of leaving. In a real world application, this would be an appropriate place to incorporate the knowledge of managers into the control problem. If all the diagonal elements of $\Lambda(X)$ are the same, μ^{LL} reduces to the myopic policy. The matrix $\Lambda(X)$ scales the terms $p_i^A(X)$ by a risk factor unique to each product/segment combination. Therefore, under μ^{LL} products that are very good (or very safe) for only a few segments are more likely to be chosen than products that are average for all segments. The opposite would be true with the myopic policy. Consequently, agents acting according to μ^{LL} observe more extreme reactions and learn faster.

The $\mu^{LL}(b)$ policy can be improved upon by allowing the agent to learn between interactions. In other words, the agent acts according to $\mu^{LL}(b)$ but updates the beliefs after each interaction. This idea can be generalized to a class of policies where during each period the agent solves a n- period problem and chooses the action that is optimal for the first period, in a rolling-horizon manner. More precisely,

$$\begin{aligned} \mu^{LL(n)} &= \arg\max_{X \in \mathbf{X}} \left\{ E\left(\sum_{t=0}^{n} \gamma^t r_t \right) \right\} \\ & \quad s.t. \\ r_n(b) &= V^{LL}(b) \end{aligned}$$

In the specific case where $n = 1$ the corresponding policy is given by

$$\mu^{LL(1)} = \arg\max_{X \in \mathbf{X}} \left\{ \tilde{R}(b, X) + \gamma \sum_{\theta \in \Theta} P(\theta | b, X) V^{LL}(\varphi(b, \theta, X)) \right\}. \tag{16}$$

Figure 7 shows the results of a numerical experiment designed to compare the performance of policies $\mu^{LL(1)}$ and μ^{M}, the myopic policy. This experiment consisted of simulating purchase experiences for two sets of 500 customers, each being exposed to one of the two control policies. Figure 7a shows the customer a base as a function of the number of interactions. The myopic policy takes less risks and performs better in the short-run, but agents using $\mu^{LL(1)}$ learn faster and are able to retain a higher percentage of customers after 4 or so interactions. The net result is that policy $\mu^{LL(1)}$ leads to a consistently higher customer base after 9 interactions. Figure 7b shows that the initial investment in learning by $\mu^{LL(1)}$ pays off in the long-run in the form of higher cumulative profits.

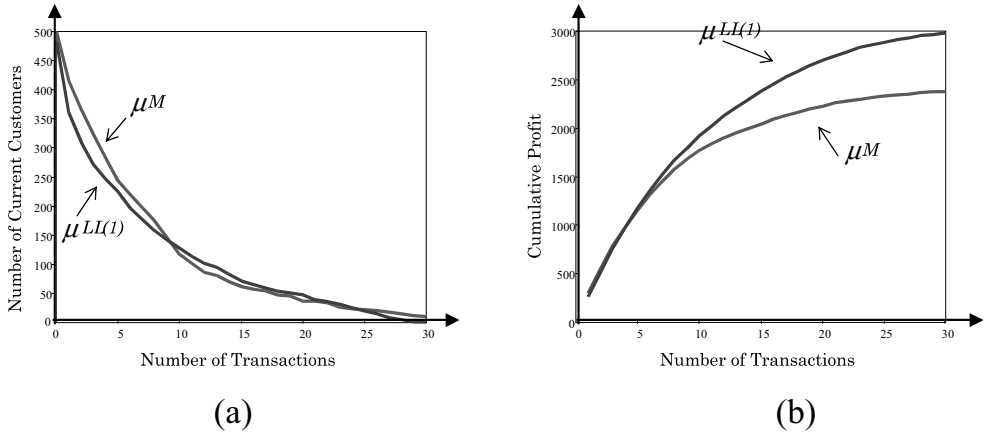

Figure 7 – Comparison of the "limited learning" and "myopic" policies.

The policy $\mu^{LL(N)}$ consistently yields good results in empirical tests. The expression for $V^{LL}(b)$ can be combined with an upper bound based on the assumption of perfect information to compute performance bounds as a function of the belief state:

$$\max_{X\in\mathbf{X}}\left\{\sum_{i=1}^{|\mathbf{S}|} b_i\cdot\left(\frac{p_i^A(X)}{1-\gamma\left(1-p_i^L(X)\right)}\right)\right\}\leq V^*(b)\leq\left\{\sum_{i=1}^{|\mathbf{S}|} b_i\cdot\max_{X\in\mathbf{X}}\left(\frac{p_i^A(X)}{1-\gamma\left(1-p_i^L(X)\right)}\right)\right\}. \quad (17)$$

Note, first of all, that the upper and lower bounds are very close to each other in the extreme points of the belief space. The worst-case scenario happens when the company knows very little about the customer. In this case, however, we would expect the upper bound to be very loose, since it is based on the assumption of perfect information. This suggests that using the lower bound value function as a basis for control may yield good results even if the upper and lower bounds are not very close.

Figure 8 discretizes the continua of customer heterogeneity and product variety to explain why $\mu^{LL(1)}$ performed well in our simulations and is likely to perform well in real-world scenarios. The first column refers to the situation when there are few customer segments and they are all very similar. Customization is of little value in this situation because one standardized product should suit all customer segments equally well. The cases described by the first row are equally uninteresting. If all products are very similar, the agent will learn very little by observing customers over time. However, profits will not be affected because the limited product variety would prevent the agent from catering to specific tastes in the first place. Cell (4) accurately captures the situation where online customization is most valuable (and most commonly used): different customer segments and different products. Since customers differ in their reactions to recommendations, early observations are very informative. This is precisely the situation when the policy performs well, because it observes "extreme" reactions from different segments and learns fast, moving to one of the regions in the belief space where $\mu^{LL(1)}$ is more likely to coincide with the optimal policy.

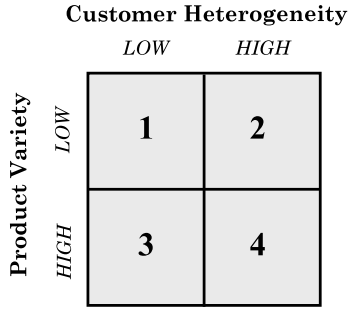

Figure 8 – Possible configurations of customer segments and products.

We conclude this section listing five important properties of the $\mu^{LL(1)}$ policy.

1. Convergence to the optimal policy

As the belief state approaches the extreme points of the belief space, the upper and lower bounds are very close to each other. This implies that over time, as the company learns about the customer, the policy $\mu^{(LL)}(b)$ will be the same as the optimal policy.

2. Lower the probability that the customer will leave after each interaction due to a bad recommendation.

This can only be guaranteed probabilistically because the observation error could theoretically be large enough to cause the customer to behave suboptimally. The likelihood of this event happening decreases as the utility of the recommendation increases, and the utility of the recommendations will increase over time.

3. Better revenues than the myopic policy.

When the company already knows the customer well, $\mu^{LL(1)}$ the myopic policy, and the optimal policy perform equally well. In situations where tthe agent needs to learn, $\mu^{LL(1)}$ loses to the myopic policy in terms of short-run revenue, wins in terms of expected revenues over time.

4. Learn faster than the myopic policy.

The myopic policy is more likely to suggest products that have similar utilities across all customer segments when the agent is not sure about the customer's true segment. This situation is likely to happen at the beginning of the relationship, and suggesting such products leads to little, if any, learning.

5. Ease of computation (or, better yet, closed-form solution).

The value function $V^{LL}(b)$ can be described in closed-form. Therefore, policies $\mu^{LL(n)}$ are very easy to compute and implement for small n.

4 DISCUSSION

4.1 Solving of the Agent's Dilemma

The main question of this paper is how companies should customize their products and services in settings such as the Internet. The analysis unambiguously shows that the pre-dominant paradigm of maximizing the expected utility of the current transaction is not the best strategy. Optimal policies must take into account future profits and balance the value of learning with the risk of losing the customer. This paper describes two ways of obtaining good recommendation policies: (i) applying POMDP algorithms to compute the exact optimal solution, (ii) approximating the optimal value function. The heuristic we provide worked well on the tests we performed, but we do not claim this heuristic to be the best one for all possible applications. Indeed, there are better heuristics for the solution of POMDPs with certain characteristics and the development of such heuristics for specific managerial settings is a topic of great interest to current researchers.

The bounds analyzed in the previous section make explicit why agents acting according to a myopic policy perform so poorly. These agents fail to recommend products that can be highly rewarding for some segments because they fear the negative impact that these recommendations could have on other segments. This fear is legitimate – companies can lose sales and customers can defect – but it can be quantified through a matrix $\Lambda(X)$ that captures the risk associated with each possible recommendation for each segment. Optimally-behaving agents make riskier suggestions than myopic agents, learn faster, and earn higher payoffs in the long-run.

4.2 Learning and the Value of Loyalty

The more quality (utility) a company provides the customer, the more likely the customer is to come back. The better the company knows the customer, the more quality it can provide. Yet, learning about the customer can mean providing disutility (less quality) in the short-run. The interaction policy described in this paper reconciles these two apparently contradictory statements. The model suggests that the concept of loyalty must be expanded to incorporate learning. The extent to which repeated purchases lead to high value and loyalty depends on the company's ability to learn about their customers as they interact.

The benefits of loyalty to profitability have been abundantly discussed in the managerial literature. Reichheld (1996) gives the following examples of how loyalty leads to profits:

- In an industrial laundry service, loyal customers were more profitable because drivers knew shortcuts that made delivery routes shorter and led to cost reductions.
- In auto-service, if the company knows a customer's car, it has a better idea of the type of problem the car will have and will be able to identify and fix the problem more efficiently.
- A shop that sells to the same group of people year after year saves on inventory costs because it can stock only the items its customers like.

These examples suggest an intimate relationship between learning and loyalty. In the laundry example, the fact that the customers have made a number of purchases from the same service provider has no direct effect on cost reduction. If a particular customer were to leave the company and its neighbor became a new customer, the cost savings would be the same in spite of the newness of the new customer. What seems to matter is knowledge, not length of tenure or number of previous purchases. Changes in profitability and retention rate over time depend on the firm's ability to learn from past interactions and customize its services so as to provide customers with more quality.

The observations of the previous paragraph are corroborated by the analysis of Section 3. Figure 9 reproduces and labels the graph of Figure 6b. When the curves intersect, both firms have a customer base of size N who have been customers for T interactions. Even though both customer bases might be assigned the same value by a number of metrics (*e.g.*, size of customer base, length of tenure) the difference in knowledge acquired during the first T interactions makes an enourmous difference in future profits. A metric such as RFM (recency, frequency, monetary value) might even assign a higher value to customers of the myopic agent. The ability to offer a large variety of products to a heterogeneous customer population (cell 4 in Figure 8) is not sufficient to generate rising revenues over time. It is also necessary to learn over time and customize services accordingly. Reinartz & Kumar (2000) make no mention of learning about customer preferences and customized services in their study, and their conclusion is that profitability per customer does not increase over time. Reichheld & Schefter (2000) conduct their study in the context of the Internet, where customer information is easily gathered and customization is commonplace. Not surprisingly, they find that profits per customer rapidly increase over time and that the benefits of loyalty on the online world far exceed those of the brick-and-mortar world.

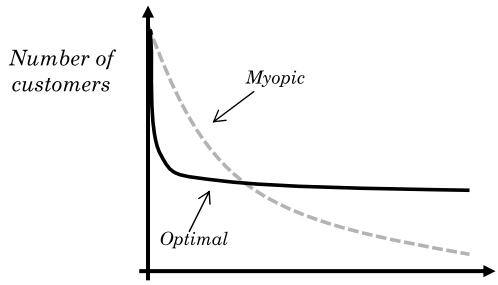

Figure 9 – The effect of knowledge of profits.

4.3 The Value of Knowing the Customer

The model presented in this paper shows how to quantify the value of learning about cus-tomers. Therefore, the optimal value function $V^*(b)$ can be interpreted as the lifetime value of a customer as a function of how well the company knows that customer. The fact that $V^*(b)$ is convex (proved in Theorem 1) has interesting managerial implications.

Consider, for example, a company operating in an industry where customers of Segment 1 are half as profitable as customers in Segment 2. Which marketing campaign would be most profitable: (i) one that only brings in customers from Segment 1 or (ii) one that brings customers from both segments with equal probability? The answer is that it depends on the shape of the value function. Figure 10 shows three possible value functions. If $V^*(b) = V_1^*(b)$, then campaign (ii) is better; if $V^*(b) = V_3^*(b)$, then campaign (i) is better. Both campaigns are equally profitable if $V^*(b) = V_2^*(b)$. The intuition behind this result is that sometimes it can be costly to learn about one's own customers. In some situations the company is better off knowing for sure that they are providing excellent value to the least desirable segment. The lower the switching cost and the higher the variance in the utility observation error, the more companies should invest in targeted advertising. This result provides an important connection between a firm's acquisition and retention policies: the V_i^* curves, which guide the acquisition policy, are determined by the customization policy.

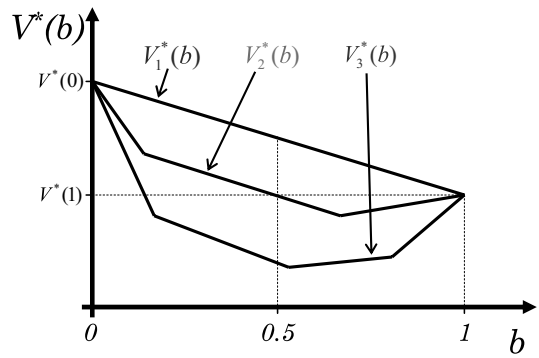

Figure 10 – The value of customers as a function of how well the company knows them.

The more a company knows the preferences of a particular customer, the more it should want to learn even more about that customer in order to transform knowledge into cus-tomized, highly valued services. Intuitively, one may think the information is most valuable when the company doesn't know anything about the customer. The convexity of the value function implies the opposite. In qualitative terms, the change in the value of the customer as the company's level of knowledge changes from "very good" to "excellent" is greater than the change in value as knowledge goes from "average" to "good". This finding provides a theoretical justification for the empirical finding that the difference in loyalty between "very satisfied" and "satisfied" customers is much greater than the difference between "satisfied" and "slightly dissatisfied" customers (*e.g.*, Heskett *et al.*, 1994). If a company knows a customer well, that customer will receive consistently good recommendations and be very satisfied.

5 DIRECTIONS FOR FUTURE RESEARCH

The results obtained in this paper drew on previous research in Management Science and Consumer Behavior. Future research in this topic can be pursued in both these directions. From the

Management Science perspective, the issue of the best way to implement web-based customization policies remains open. The value function approximations derived in this paper lead to a number of insights, but exact numerical methods generate better control policies. More work needs to be done to understand which approximation methods work best in situations where the problem is too large to compute the optimal policy. A second open question is how to recommend sets of products. In some situations agents are asked to recommend more than one product. One obvious way of addressing this issue is to consider each possible set as a single product and reduce the problem to the one solved in this paper. This method can be inefficient if the set of possible products is large, because of the exponential increase in the number of potential suggestions. Therefore, it is worthwhile to explore other ways to generate good policies.

From the Consumer Behavior perspective, there are two topics of interest. First, the de-velopment of efficient perception management devices to reduce the probability of defection. In some situations, it may be possible to have a "suggestion" to explore tastes as well as a "recommendation" to maximize utility. It has been shown that customers do not forgive bad recommendations, but could they forgive bad suggestions? Second, it is important to investigate how customers aggregate their experiences with agents over time. The model in this paper assumes that customer behavior is solely determined by the current product offering. In reality, the customers' perceptions of the agent's quality evolve over time. But how exactly does this evolution take place? On one hand, there is a stream of research which suggests that recency (the satisfaction derived in the last few interactions) is the most important determinant of current satisfaction (for a review see Ariely & Carmon, 2000). On the other hand, first impressions are important and it can be very difficult for a customer to change his or her initial perception of the agent's quality (*e.g.*, Hogarth & Einhorn, 1992). These two views are in direct opposition, and it is not clear which is more important in web-based settings. Further research should also investigate how customers' expectations change as a function of the quality of the advice they receive.

REFERENCES

[1] ARIELY D & CARMON Z. 2000. Gestalt characteristics of experienced profiles. *Journal of Behavioral Decision Making*, **13**: 219–232.

[2] ARIELY D, LYNCH JG & APARICIO M. 2004. Learning by collaborative and individual-based recommendation agents. *Journal of Consumer Psychology*, **14**(1&2): 81–94.

[3] BERGER PD & NASR NI. 1998. Customer lifetime value: Marketing models and applications. *Journal of Interactive Marketing*, **12**: 17–30, (Winter).

[4] BERTSEKAS D. 1995. Dynamic Programming and Optimal Control, Volume 1. Athena Scientific.

[5] BERTSEKAS D. 1995. Dynamic Programming and Optimal Control, Volume 2. Athena Scientific.

[6] BERTSEKAS D & TSITSIKLIS J. 1996. Neuro-Dynamic Programming. Athena Scientific, Belmont, MA.

[7] DONEY PM & CANNON JP. 1997. An examination of the nature of trust in buyer-seller relationships. *Journal of Marketing*, **61**: 35–51, (April).

[8] GREENE WH. 2000. Econometric Analysis, 4th edition. Prentice Hall, New Jersey, U.S.A.

[9] HESKETT JL, JONES TO, LOVEMAN GW, SASSER WE JR & SCHLESINGER LA. 1994. Putting the service-profit chain to work. Harvard Business Review. March-April, 164–174.

[10] HOGARTH RM & EINHORN HJ. 1992. Order effects in belief updating: The belief adjustment model. *Cognitive Psychology*, **24**: 1–55, (January).

[11] JAIN D & SINGH SS. 2002. Customer lifetime value research in marketing: A review and future directions. *Journal of Interactive Marketing*, **16**(2): 34–46.

[12] KAELBLING LP, LITTMAN ML & MOORE AW. 1996. Reinforcement learning: A survey. *Journal of Artificial Intelligence*, **4**: 237–285.

[13] KOLLOCK P. 1999. The production of trust in online markets. *Advances in Group Processes*, **16**: 99–123.

[14] LEWICKI RJ, MCALLISTER DJ & BIES RJ. 1998. Trust and distrust: New relationships and realities. *Academy of Management Review*, **23**, (July).

[15] LITTMAN ML. 1994. The witness algorithm: Solving partially observable markov decision processes. Department of computer science, technical report, Brown University, Providence, R.I.

[16] LITTMAN ML, CASSANDRA AR & KAEBLING LP. 1995. Learning policies for partially observable environments: Scaling up. In: PRIEDITIS A & RUSSELL S (Eds), Proceedings of the Twelfth International Conference on Machine Learning, 362–370. Morgan Kaufmann, San Francisco, CA.

[17] MAES P. 1995. Intelligent software. *Scientific American*, **273**(9): 84–86.

[18] MCFADDEN D. 1981. Econometric models of probabilistic choice. In: Structural Analysis of Discrete Data with Econometric Applications, MANSKY CF & MCFADDEN D (Eds), MIT Press.

[19] MEYER RJ & KAHN BE. 1990. Probabilistic models of consumer choice behavior. Handbook of Consumer Theory and Research, ROBERTSON T & KASSARJIAN H (Eds), Prentice-Hall.

[20] MONAHAN G. 1982. A survey of partially observable markov decision processes: Theory, models, and algorithms. *Management Science*, **28**(1): 1–15.

[21] PFEIFER PE & CARRAWAY RI. 2000. Modeling customer relationships as Markov chains. *Journal of Interactive Marketing*, **12**(2): 43–55.

[22] REICHHELD F. 1996. The Loyalty Effect. Harvard Business School Press.

[23] REICHHELD F & SCHEFTER P. 2000. E- loyalty: Your secret weapon on the web. *Harvard Business Review*, **78**: 105–113, (July-August).

[24] REINARZ WJ & KUMAR V. 2000. On the profitability of long lifetime customers: An empirical investigation and implications for marketing. *Journal of Marketing*, **64**: 17–35.

[25] ROSS S, PINEAU J, PAQUET S & CHAIB-DRAA B. 2008. Online Planning Algorithms for POMDPs. *Journal of Artificial Intelligence Research*, **32**: 663–704.

[26] ROSSI P, MCCULLOCH R & ALLENBY G. 1996. The value of purchase history data in target marketing. *Marketing Science*, **15**(4): 321–340.

[27] SONDIK EJ. 1978. The optimal control of a partially observable markov decision process over the infinite horizon: Discounted costs. *Operations Research*, **26**: 282–304.

[28] TOBIN. 2008. A Stochastic Point-Based Algorithm for Partially Observable Markov Decision Processes. M. Sc. Thesis, Faculté de Sciences et Génie, Université Laval, Québec.

[29] TOUBIA O, SIMESTER DR & HAUSER JR. 2003. Fast polyhedral adaptive conjoint estimation. *Marketing Science*, **22**(3): 273–303.

[30] URBAN GL, SULTAN F & QUALLS WJ. 2000. Placing trust at the center of your internet strategy. *Sloan Management Review*, **42**(1): 39–48.

[31] WHITE III CC. 1976. Optimal diagnostic questionnaires which allow less than truthful responses. *Information and Control*, **32**: 61–74.

[32] ZAUBERMAN G. 2000. Consumer choice over time: The perpetuated impact of search costs on decision processes in information abundant environments. Ph.D. Dissertation, Duke University, Durham, NC.

A Appendix

A.1 Proof of Proposition 1

The proof is by induction, and therefore consists of the verification of a base case and the induction step.

(1) Base case: To prove that there exists a set of vectors A^1 *such that*

$$V^1(b) = \max_{\alpha \in A^1} (b \cdot \alpha).$$

The optimal policy with 1 step to go is to act myopically with respect to immediate payoffs:

$$V^1(b) = \max_{X \in \mathbf{X}} \left\{ \tilde{R}(b, X) \right\} = \max_{\alpha \in A^1} (b \cdot \alpha)$$

where the last equality holds by letting $A^1 = \bigcup_{X \in \mathbf{X}} p_i^A(X)$.

(2) Induction step

If

$$\exists A^{n-1} \text{ such that } V^{n-1}(b) = \max_{\alpha \in A^{n-1}} (b \cdot \alpha)$$

then

$$\exists A^n \text{ such that } V^n(b) = \max_{\alpha \in A^n} (b \cdot \alpha)$$

From the induction hypothesis, we can write, for $\theta \in \left\{\theta^A, \theta^R\right\}$,

$$V^{n-1}(\varphi(b, \theta, X)) = \max_{\alpha \in A^{n-1}} (\varphi(b, \theta, X) \cdot \alpha) = \varphi(b, \theta, X) \cdot \alpha^{n-1}(b, \theta, X) \tag{18}$$

where

$$\alpha^{n-1}(b, \theta, X) = \arg\max_{\alpha \in A^{n-1}} (\varphi(b, \theta, X) \cdot \alpha).$$

Since $V^n = TV^{n-1}$, we obtain $V^n(b)$ by applying the mapping $(TV)(b)$ to the V^{n-1} function above:

$$V^n(b) = \max_{X \in \mathbf{X}} \left\{ \tilde{R}(b, X) + \gamma \sum_{\theta \in \left\{\theta^A, \theta^R\right\}} \Pr(\theta | b, X) \left[\varphi(b, \theta, X) \cdot \alpha^{n-1}(b, \theta, X) \right] \right\}. \tag{19}$$

Recall the definition of $\varphi(b, \theta, X)$ and note that $\varphi\left(b, \theta^A, X\right)$ and $\varphi\left(b, \theta^R, X\right)$ can be written as

$$\varphi\left(b, \theta^A, X\right) = \frac{b \cdot \Gamma^A(X)}{\Pr\left(\theta^A | b, X\right)}$$

and

$$\varphi\left(b, \theta^R, X\right) = \frac{b \cdot \Gamma^R(X)}{\Pr\left(\theta^R | b, X\right)}.$$

If we substitute $\tilde{R}(b, X) = \sum_{i=1}^{|\mathbf{S}|} b_i \cdot R\left(S_i, X_j\right)$ and the expressions for $\varphi(b, \theta, X)$ above into the expression for $V^n(b)$ we obtain

$$V^n(b) = \max_{X \in \mathbf{X}} \Big\{ b \cdot p^A(X) + \gamma \cdot b \cdot \Gamma^A(X) \cdot \alpha^{n-1}\left(b, \theta^A, X\right) + \gamma \cdot b \cdot \Gamma^R(X) \cdot \alpha^{n-1}\left(b, \theta^R, X\right) \Big\}$$

by cancelling out the terms $\Pr\left(\theta^A | b, X\right)$ and $\Pr\left(\theta^R | b, X\right)$.

Factoring out the vector b and defining the set A^n as

$$A^n = \bigcup_{X \in \mathbf{X}} \left\{ \bigcup_{\alpha^i \in \mathbf{A}^{n-1}} \left[\bigcup_{\alpha^j \in \mathbf{A}^{n-1}} \left(p^A(X) + \gamma \Gamma^A(X) \cdot \alpha^i + \gamma \Gamma^R(X) \cdot \alpha^j \right) \right] \right\}$$

we conclude that

$$V^n(b) = \max_{\alpha \in A^n} (b \cdot \alpha)$$

A.2 Proof of Proposition 2

If agents are not allowed to learn, then the value function that maximizes future expected profits for the $n-$period problem is given by

$$\hat{V}^n(b) = \max_{\alpha \in \hat{A}^n} (b \cdot \alpha).$$

where $\hat{A}^1 = A^1$ and the sequence of sets $\left\{\hat{A}^1, \hat{A}^2, ..., \hat{A}^n\right\}$ is defined by the recursion

$$\hat{A}^n = \bigcup_{X \in \mathbf{X}} \left(\bigcup_{\alpha^i \in \hat{A}^n} \left[p^A(X) + \Gamma(X) \cdot \alpha^i \right] \right)$$

where $\Gamma(X) = \Gamma^A(X) + \Gamma^R(X)$.

The value functions $\hat{V}^n(b)$ can be explicitly written as maximizations over $\alpha \in \hat{A}^1$ as follows:

$$\hat{V}^1(b) = \max_{\alpha \in \hat{A}^1} (b \cdot \alpha) = \max_{X \in \mathbf{X}} \left(b \cdot p^A(X) \right)$$

$$\begin{aligned}
\hat{V}^2(b) &= \max_{\alpha\in\hat{A}^2}(b\cdot\alpha) = \max_{X\in\mathbf{X}}\left(b\cdot\left[p^A(X)+\Gamma(X)\cdot p^A(X)\right]\right)\\
&\vdots\\
\hat{V}^n(b) &= \max_{\alpha\in\hat{A}^n}(b\cdot\alpha)\\
&= \max_{X\in\mathbf{X}}\left(b\cdot\left[p^A(X)+\Gamma(X)\left[p^A(X)+\Gamma(X)\left[p^A(X)+\cdots\right]\cdots\right]\cdot p^A(X)\right)\right.\\
&= \max_{X\in\mathbf{X}} b\cdot p^A(X)+b\cdot\Gamma(X)\cdot p^A(X)+\cdots+b\cdot[\Gamma(X)]^{n-1}\cdot p^A(X)\\
&= \max_{X\in\mathbf{X}} b\left[\mathbf{I}+\sum_{i=1}^{n-1}[\Gamma(X)]^i\right]\cdot p^A(X)
\end{aligned}$$

The sequence of functions $\hat{V}^n(b)$ converges:

$$\begin{aligned}
\lim_{n\to\infty}\ &\left(\hat{V}^{n+1}(b)-\hat{V}^n(b)\right) = \qquad (20)\\
&= \lim_{n\to\infty}\left(\max_{X\in\mathbf{X}} b\left[\mathbf{I}+\sum_{i=1}^{n-1}[\Gamma(X)]^i\right]\cdot p^A(X)-\max_{X\in\mathbf{X}} b\left[\mathbf{I}+\sum_{i=1}^{n-2}[\Gamma(X)]^i\right]\cdot p^A(X)\right)\\
&= \lim_{n\to\infty}\left(\max_{X\in\mathbf{X}}[\Gamma(X)]^n\cdot p^A(X)\right)\\
&= 0
\end{aligned}$$

The last step follows from the fact that all the elements in $\Gamma(X)$ are in $(0,1)$, and therefore $\lim_{n\to\infty}[\Gamma(X)]^n$ is a matrix of zeroes. Now consider the difference

$$D^n(b) = b\cdot\hat{\alpha}-V^n(b)$$

where

$$\hat{\alpha} = \arg\max_{X\in\mathbf{X}}\left(b\cdot[\mathbf{I}-\Gamma(X)]^{-1}\cdot\left[p^A(X)\right]\right).$$

Substituting the full expression for $V^n(b)$ yields

$$D^n(b) = \max_{X\in\mathbf{X}}\left(b\cdot[\mathbf{I}-\Gamma(X)]^{-1}\cdot\left[p^A(X)\right]\right)-\left(\max_{X\in\mathbf{X}} b\cdot\left[\mathbf{I}+\sum_{i=1}^{n-1}[\Gamma(X)]^i\right]\cdot p^A(X)\right)$$

Since all the elements of $\Gamma(X)$ are in $(0,1)$, the expression $[\mathbf{I}-\Gamma(X)]^{-1}$ is well-defined and can be expanded to

$$[\mathbf{I}-\Gamma(X)]^{-1} = \mathbf{I}+\sum_{i=1}^{\infty}[\Gamma(X)]^i.$$

Substituting the expansion above into the expression for D^n yields

$$D^n(b) = \max_{X \in \mathbf{X}} \left(b \cdot \left[\sum_{i=n}^{\infty} [\Gamma(X)]^i \right] \cdot p^A(X) \right).$$

It then follows that

$$\lim_{n \to \infty} D^n(b) = 0, \forall b \tag{21}$$

The result from (20) ensures that the sequence $V^n(b)$ converges to a limit $V^*(b)$. Equation (21) asserts that $V^n(b)$ also converges to $b \cdot \hat{\alpha}$. Since the dynamic programming mapping is a monotone contraction, the limit is unique, so $\hat{\alpha}$ corresponds to the policy that maximizes the infinite horizon value function.

A.3 Simulation parameters

Parameters used for simulation in Figure 7:

$c = 2.5$

$\varepsilon \sim Normal(0, 3)$

$u_1(1) = -2 \quad u_2(1) = 5$

$u_1(2) = 5 \quad u_2(2) = -2$

$u_1(3) = 0 \quad u_2(3) = 0$

$u_1(4) = 3 \quad u_2(4) = 3$