Data-driven Direct Marketing via Approximate Dynamic Programming

Jesper Slik and Sandjai Bhulai

Vrije Universiteit Amsterdam
Faculty of Science, Department of Mathematics
Email: jesper.slik@pon.com and s.bhulai@vu.nl

***Abstract*—Email marketing is a widely used business tool that is in danger of being overrun by unwanted commercial email. Therefore, direct marketing via email is usually seen as notoriously difficult. One needs to decide which email to send at what time to which customer in order to maximize the email interaction rate. Two main perspectives can be distinguished: scoring the relevancy of each email and sending the most relevant, or seeing the problem as a sequential decision problem and sending emails according to a multi-stage strategy. In this paper, we adopt the second approach and model the problem as a Markov Decision Problem (MDP). The advantage of this approach is that it can balance short- and long-term rewards and allows for complex strategies. We illustrate how the problem can be modeled such that the MDP remains tractable for large datasets. Furthermore, we numerically demonstrate by using real data that the optimal strategy has a high interaction probability, which is much higher than a greedy strategy or a random strategy. Therefore, the model leads to better relevancy to the customer and thereby generates more revenue for the company.**

***Keywords*—email marketing; Markov decision processes; approximate dynamic programming; recommendation systems.**

I. INTRODUCTION

Customer communication is crucial to the long-term success of any business. Research has shown communication effectiveness to be the single most powerful determinant of relationship commitment [1]. Companies can choose from multiple channels in reaching their customers. The recent rise of social media has expanded the possibilities immensely. Most research focuses on email communication though, because it is relatively easy to collect data of every email sent and every interaction resulting from the email on a customer level. Therefore, a thorough analysis of email communication effectiveness is possible.

Currently, in most companies, domain experts determine the email strategy. Customers are selected for emails based on business rules. These rules can be deterministic, such as matching the language or gender of the email with those of the customer, or stochastic, such as matching the (browsing) activity categories of a customer to the category of the email. Measurements suggest that a large fraction of the emails are unopened, a larger portion of the emails do not even direct customers to the company's website, and almost all emails are not related to direct sales. An increase in the interaction probability, therefore, directly leads to additional revenue. This probability can be increased by a better recommendation process of deciding which email to send at what time to which customer.

The challenge faced in this research can be classified within the research field of recommender systems. A recommender system has as purpose to generate meaningful recommendations of items (articles, advertisements, books, etc.) to users. It does so based on the interests and needs of the users. Such systems solve the problem of information overload. Users might have access to millions of choices but are only interested in accessing a fraction of them. For example, Amazon, YouTube, Netflix, Tripadvisor, and IMDb use recommender systems to display contents on their web pages [2]. Similarly, one can use recommender systems to recommend certain emails to users, thus, to determine when to send which email to which user.

Recommender systems have traditionally been classified into three categories: content-based filtering, collaborative filtering, and hybrid approaches [3]. Content-based filtering is a recommendation system that learns from the attributes (or the so-called contents) of items for which the user has provided feedback [4]. By doing so, it can make a prediction on the relevancy of items for which the user has not provided feedback. Collaborative filtering looks beyond the activity of the user for which a recommendation needs to be made. It recommends an item based on the ratings of similar users [3]. Hybrid recommender systems make use of a combination of the above-mentioned techniques in order to generate recommendations.

Although recommender systems might seem a good way to address the direct marketing problem, they have some shortcomings. One of the major problems for recommender systems is the so-called cold-start problem. This concerns users or items which are new to the system, thus little information is known about them. A second issue is that traditional recommender systems take into account a set of users and items and do not take into account contextual information. Contextual information might be crucial for the performance of a recommender system [5]. A third issue is overspecialization: "When the system can only recommend items that score highly against a users profile, the user is limited to being recommended items that are similar to those already rated" [3]. Lastly, recommender systems must scale to real data sets, possibly containing millions of items and users. As a consequence, algorithms often sacrifice accuracy for having a low response time [2]. When a data set increases in size, algorithms either slow down or require more computational resources.

The main contribution of this paper is that we address

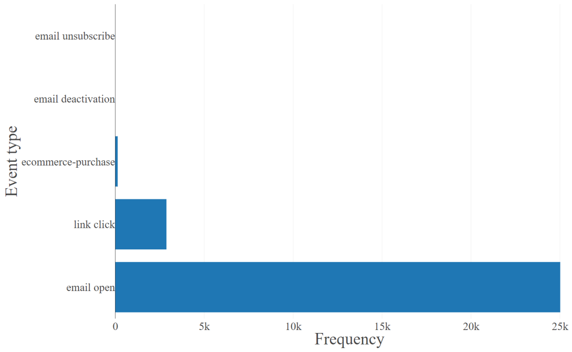

Figure 1. Frequency of event types.

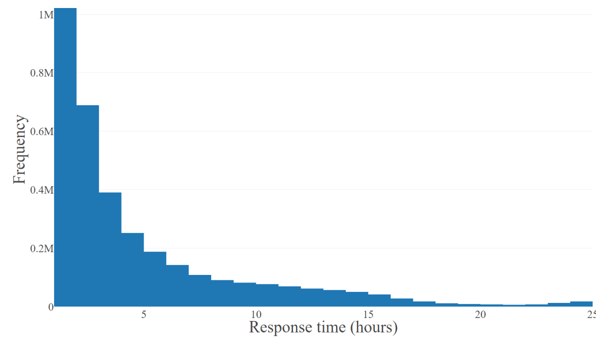

Figure 2. Distribution of time until first interaction with an email.

the mentioned shortcomings of the traditional recommender systems by formulating the direct marketing problem as a Markov Decision Process (MDP). This framework deals with context and uncertainty in a natural manner. The context (such as previous email attempts) can be specified in the state space of the MDP. The uncertainty is addressed by the optimal policy as an exploration-exploitation tradeoff. The scalability of the algorithm is addressed by limiting the history of the process to sufficient information such that the state space does not grow intractably large. Furthermore, we test our model with real data on a greedy and random policy as a benchmark. The results show that our optimal strategy has a significantly higher interaction probability than the benchmark.

The organization of this paper is as follows. In Section II, we describe the data used for our data-driven marketing algorithm. Section III describes the model and introduces the relevant notation. In Section IV, we analyze the performance of the model and state the insights from the model. Finally, in Section V we conclude and address a number of topics for further research.

II. Data

In this section, we describe the data used for this research. The data is gathered from five tables of an international retailer from one complete year and concerns: *sales* data, *email sent* data, *email interaction* data, *customer activity* data, and *customer* data.

The *sales* table contains all orders that have been placed by each customer. This includes information on the product, price, and date. The *email sent* table contains all emails sent to each customer. An email is characterized by attributes such as title, category, type, gender, and date. The *email interaction* table is structured similarly to the *email sent* table, however, it contains an interaction type. An interaction type can be email open, link click, e-commerce purchase, email unsubscribe, or email deactivation. The *customer activity* table contains for each customer its activity on the retailer's platform, such as browsing or clicking on the website. Finally, the *customer* table contains characteristics of a customer, such as date of birth, country, city, and gender.

The retailer has over 1 million unique active customers in its database. In total, a little more than 132 million emails were sent, leading to around 34.5 million interactions. The main interaction category is 'email open', which occurs over five times more frequently than the second interaction category, 'link click'. This is intuitive, as an email needs to be opened in order to click a link. Even fewer emails are related to direct online sales and rarely an email leads to an unsubscribe or deactivation (see Figure 1). The customers that interact with an email, usually do so within a few hours. The majority even within one hour, with the number of interactions declining by the hour afterward. Only after 24 hours, there is a slight increase in the number of interacting customers (see Figure 2).

With the current email strategy, the retailer does not send the same emails to the same customers. The average customer receives an email every other day and interacts with an email every 10 days. Interestingly, some customers interact with more than 1 email per day on average. The email interaction rate varies between the email category and email type. The interaction rate of individual emails shows even larger differences. This rate ranges from 3.4% to 67%.

In this research, we are mainly interested in delivering relevant communication to the customers. Whether an email is relevant to a customer can be expressed by whether the customer interacted with the email. We investigate two correlations related to the email interaction rate. We do this by visualizing the relation with a scatter plot (plotting a random sample of the data) and including a 95% confidence interval for the mean. The confidence interval is created through a bootstrap procedure.

Figure 3 (left) visualizes the correlation between the average number of emails received and the number of interactions. The average daily interactions is positively correlated with the average daily emails. This is intuitive, as it would benefit no strategy to send more emails to a customer that does not interact with emails. Also, it is impossible for a customer to interact with 2 emails if the customer only received 1. However, sending more emails does not necessarily mean more interactions. Figure 3 (right) visualizes the correlation between the interaction probability and total order value of

 ISBN: 978-1-61208-741-2

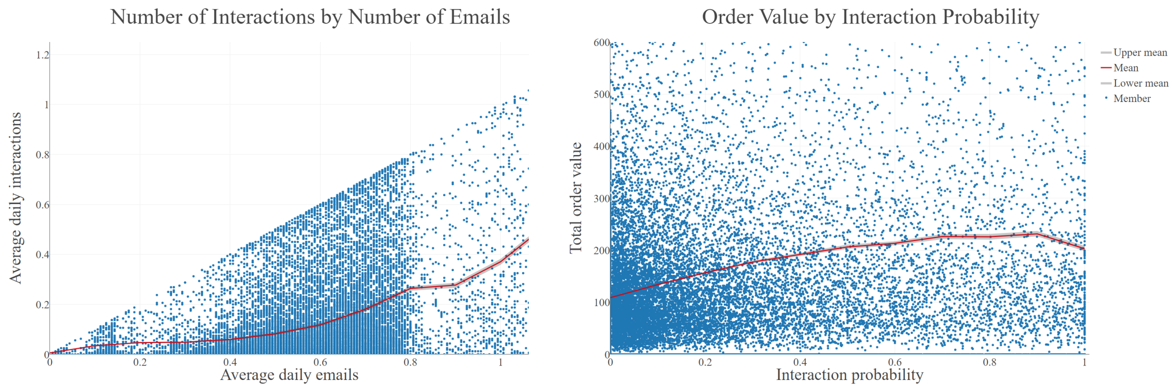

Figure 3. Scatter plots diagrams: # emails vs # interactions (left) and interaction probability vs customer order value (right).

a specific customer. The interaction probability is defined as the number of interactions divided by the number of received emails for a specific customer. The graph indicates that a higher interaction probability is correlated with a higher-order value. When looking at the interaction probabilities of 0.3 and 0.4, the confidence intervals for the mean total order value (averaged over all customers) are non-overlapping. For a probability of 0.3, the confidence interval is $[174.68, 180.71$ and for a probability of 0.4 this yields $[189.02, 195.11]$. Thus, customers that have a higher interaction probability have a higher customer value (for interaction probabilities smaller than 0.8).

III. Model description

We implement a discrete-time MDP for our email marketing process. The MDP is defined by four entities: the state space $\mathcal{S}$, the action space $\mathcal{A}$, the reward function r, and the transition function p.

We define a state $s \in \mathcal{S}$ as a vector of the form $s = (x_0, x_1, x_2, y_0, y_1)$. Here, x_i represents the $(3-i)^{\text{th}}$ previous interaction of the customer for $i \in \{0, 1, 2\}$. Similarly, y_j is defined as:

$$y_j = \begin{cases} 1, & \text{if } (2-j)^{\text{th}} \text{ previous action lead to an interaction,} \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

for $j \in \{0, 1\}$.

This choice for the state is partially inspired by [6], in which the state is defined as the sequence of the past k items bought. We make a clear distinction between actions and interactions, an action meaning sending an email to a customer and an interaction meaning the customer interacting with an email. The x_i's of the state space represent a customer's preference in content, and the y_i's represent the customer's sensitivity to emails. The parameters $i = 3$ and $j = 2$ have been empirically chosen, leading to an approximate model. There is a trade-off between tailoring the model for individuals and more accurately estimating the model parameters. The size of the state space grows exponentially as i and j are increased, since $|\mathcal{S}| = |\mathcal{A}|^i 2^j$.

We define an action $a_i \in \mathcal{A}$ as an integer. This integer represents a combination of email category and email type. An example of a category is 'household products' and an example of type is 'special event'. In our data, 20 categories and 21 types exist. However, not all combinations of category and type appear in the data. Therefore, we focus on the 20 actions that occur most frequently. In this way, we reduce the size of the action set by 95% at the cost of discarding 21% of the data.

The reward function represents the reward (business value) of a customer visiting a state. We aim to maximize the communication relevancy to the customers. This can be measured by customers interacting with emails. Thus, the reward function should measure email interactions. We define the reward function as $r(s) = y_1$ for $s = (x_0, x_1, x_2, y_0, y_1)$. This function expresses whether the previous action leads to an interaction. Conveniently, the last element in the state vector already does so.

The transition probabilities are estimated by simply counting the occurrences of a transition in the data. Specifically,

$$p(s, a, s') = \frac{C(s, a, s')}{\sum_{s' \in \mathcal{S}} C(s, a, s')},$$

in which $C(s, a, s')$ is a function that counts the number of occurrences of transitioning from state s to state s' when applying action a. To create the data to estimate these probabilities, three steps are required. First, we collect on a daily level which action and interaction was registered with which customer. Next, we compute the state of each customer based on this information. Lastly, we aggregate all state changes of all customers into one final table. These steps are visualized in Figure 4.

To summarize the implementation of the MDP, we present an example. This example is visualized in Figure 5. The example highlights that when a customer is in state $s_t = (14, 6, 10, 0, 0)$ and action $a_t = 17$ is applied, we have a 19% chance of transitioning to state $s_{t+1} = (6, 10, 17, 0, 1)$ (since $p(s_t, a_t, s_{t+1}) = p\big((14, 6, 10, 0, 0), 17, (6, 10, 17, 0, 1)\big) = 0.19$) and a 81% chance of transitioning to state $s_{t+1} = (14, 6, 10, 0, 0)$. Note that for any s_t, only two possibilities exist for s_{t+1}.

 ISBN: 978-1-61208-741-2

customer id	date	action	interaction
a	1	18	0
a	3	15	15
a	5	3	3
a	6	14	0
a	7	6	6
a	10	20	0
…	…	…	…

→

customer id	date	state	action	state next
a	1	(1, 1, 1, 1, 0)	18	(1, 1, 1, 0, 0)
a	3	(1, 1, 1, 0, 0)	15	(1, 1, 15, 0, 1)
a	5	(1, 1, 15, 0, 1)	3	(1, 15, 3, 1, 1)
a	6	(1, 15, 3, 1, 1)	14	(1, 15, 3, 1, 0)
a	7	(1, 15, 3, 1, 0)	6	(15, 3, 6, 0, 1)
a	10	(15, 3, 6, 0, 1)	20	(15, 3, 6, 1, 0)
…	…	…	…	…

→

state	action	state next	frequency
(11, 9, 17, 1, 1)	9	(9, 17, 9, 1, 1)	5197
(11, 9, 17, 1, 1)	9	(11, 9, 17, 1, 0)	828
(11, 9, 17, 1, 1)	11	(9, 17, 11, 1, 1)	6561
(11, 9, 17, 1, 1)	11	(11, 9, 17, 1, 0)	1042
(11, 9, 17, 1, 1)	12	(9, 17, 12, 1, 1)	10
(11, 9, 17, 1, 1)	12	(11, 9, 17, 1, 0)	2
…	…	…	…

Figure 4. The three data processing steps required for estimating the transition probabilities.

Modeling considerations

Multiple challenges arise when modeling the problem as an MDP. Most of these have been tackled by defining an appropriate MDP as done in the previous paragraphs. However, some modeling choices remain which are described next.

A. *The unichain condition*

In order for solution techniques to work for our model, the MDP needs to be unichain. The unichain property states that there is at least one state $s \in \mathcal{S}$, such that there is a path from any state to s [7]. A path from z_0 to z_k of length k is defined as a sequence of states $z_0, z_1, \ldots, z_k$ with $z_i \in \mathcal{S}$ with the property that $p(z_0, z_1) \cdots p(z_{k-1}, z_k) > 0$.

The unichain property does not automatically hold when we take all states and state transitions directly from the data. This is because the chain is partially observed, so for some states it is not observed that a specific action causes an interaction. For some states, it might only be observed that the next possible state is the current state. We solve this problem by removing all states for which fewer than 2 next states are observed.

B. *Estimation of transition probabilities*

In our implementation, making the MDP unichain reduces the number of observed states. A problem with the estimation of the transition probabilities is that some probabilities are based upon thousands of observations, whereas others only on a few observations. This introduces noise in the transition probabilities. To tackle this challenge, we recursively remove state transitions that occur fewer than 50 times and, if this leads to states being impossible to transition to, we also remove those and transitions to those states.

The MDP is partially observed, we initially observe 86% of the theoretically possible states. After filtering, we are left with 39% of possible states. This is a large reduction in the number of observed states, however, it does ensure we focus on the most relevant and frequently observed states. Figure 6 shows the distribution of the number of observed transitions per state before filtering.

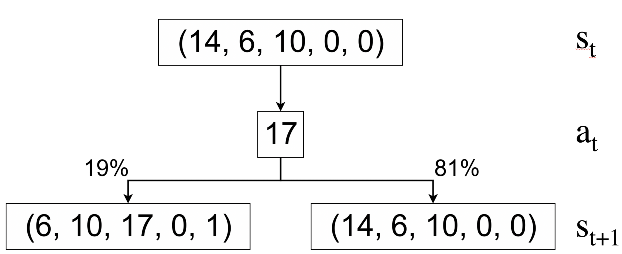

Figure 5. Example transition.

C. *Exponential growth*

Lastly, defining and solving an MDP can be difficult because of the exponential growth of the state space due to the multiple components of the state, as discussed before when setting the values of i and j. If the state space becomes too large, solving the MDP might not be realistic. To ensure the MDP can be solved within a feasible time period, we implement a custom version of the value iteration algorithm, taking into account the following issues.

In our case, the set of possible next states, defined as $E(s, a)$, only consists of 2 states. This significantly reduces the run time of the algorithm. If we would not do this, the algorithm would have to check the transition probabilities to and values of all 32,000 possible states.

We implemented the action set, $\mathcal{A}$, as being dependent on the state, thus redefining it as $\mathcal{A}(s)$. For some states, not all 20 actions are observed. So, it is unknown to the model what the transitions would be. Not taking into account these unknown actions improves the speed of the algorithm.

Finally, we initialize $E(s)$, $\mathcal{A}(s)$, and $p(s, a, s')$ for all s, a, and s' in memory, using Python dictionaries. This allows for $\mathcal{O}(1)$ lookup steps of any probability, action set, or the set of next states within the algorithm.

IV. Results

In this section, we present an analysis of the performance of the models. We analyze the strategy performance by comparing three different strategies, all based on the MDP framework:

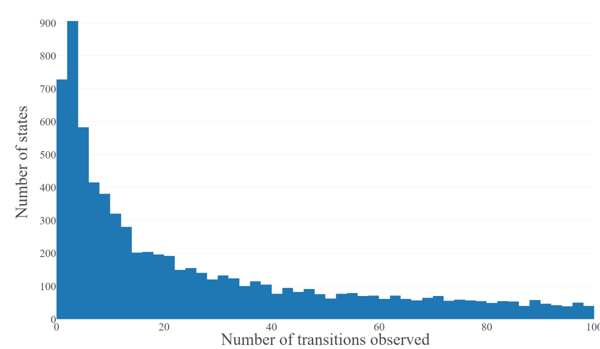

Figure 6. Distribution of the number of observed transitions per state.

Copyright (c) IARIA, 2019. ISBN: 978-1-61208-741-2

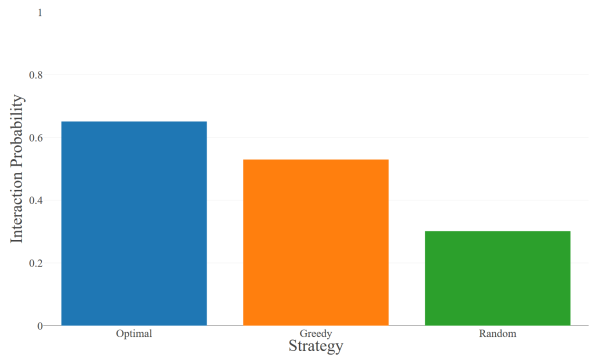

Figure 7. Comparing strategy performance: optimal vs greedy vs random.

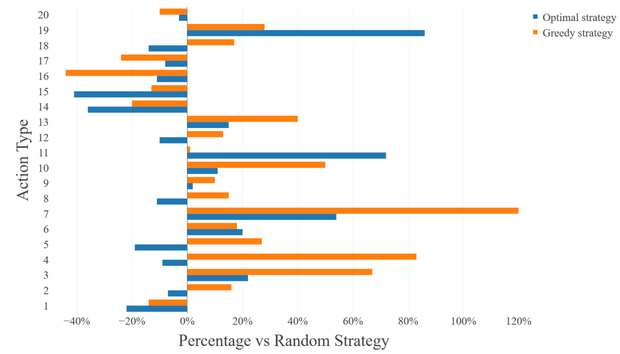

Figure 8. Action performance: frequency of an action within the optimal or greedy strategy divided by the expected frequency of that action.

the optimal strategy, a greedy strategy, and a random strategy (benchmark). The optimal strategy is calculated through value iteration, the greedy strategy through choosing in each state the action with the highest interaction probability, and the random strategy through randomly choosing an action in each state.

Figure 7 shows the resulting performance of the three strategies. The optimal strategy has the highest long-run interaction probability, corresponding to a value of 65%. The greedy strategy is second with a rate of 53%, and the random strategy with 30%. Interestingly, the interaction rate of the optimal strategy is 23% higher than the rate of the greedy strategy, showing that taking into account delayed rewards can highly increase the strategy value. Both the optimal and greedy strategy perform better than the random strategy, showing that using advanced strategies has a large impact on the interaction rate.

Figure 8 highlights the effectiveness of each action type. This effectiveness is measured by dividing the frequency of an action within the optimal or greedy strategy over the expected frequency of that action. It is measured in this way, since an absolute measure would not be accurately representing the action performance, as in some states only one action might be possible. So, the absolute measure would not represent how much the action is preferred over other actions. A comparison between the greedy and optimal strategy is made, to highlight the difference between short- and long-term rewards of the corresponding action.

Large differences are visible in action performance. Actions that perform well on both the short- and long-term are action 7: the type retail clearance, 19: weekly limited product releases in a specific category, and 6: new releases. Interestingly, some actions are highly beneficial for the long-term, but not beneficial for the short-term, see., e.g., action 4. Actions that perform poorly are action 1, 14, 15, 16, or 17, which are all weekly limited product releases. It seems that only the weekly limited product release in a specific category (action 19) performs well.

V. Conclusions and Discussion

This research shows that the retailer can increase its relevance to its customers by applying a different email strategy. Hereby, it possibly increases the revenue it generates. However, the strategy we developed is based on the data generated from the retailer's current email strategy. If the retailer starts experimenting with different strategies, this might uncover patterns unknown to the current model and potentially improve the optimal strategy we presented.

An interesting result of this research is the difference between the optimal and the greedy strategy. The interaction rate of the optimal strategy is 23% higher, relatively. Thus, the balance between short- and long-term reward should be taken into account when dealing with similar problems. If we would have chosen to use traditional methods, such as content-based or hybrid filtering, this result would not have been directly visible. These methods do not explicitly include this balance, so during the modeling process, it will be beneficial to try to include this balance.

Moreover, the results indicate a 'reality gap' between theory and practice. The interaction rate of the random strategy (30%) is higher than the interaction rate of the retailer's current strategy (27%). This is probably because our model has fewer restrictions compared to real life. However, with the interaction rate of the optimal strategy being 65%, the model shows to have potential.

Throughout this research, all data concerns the past. However, to more accurately measure the impact of strategies, it would be better to measure the performance real-time. For example, through an A/B testing procedure. Then, reinforcement learning could be used to learn the value of strategies in real-time. Next to a balance in short- and long-term reward, this algorithm balances exploration and exploitation. Thus, it tries to both learn a better strategy and apply the best-known current strategy.

Furthermore, we can extend the model by redefining actions. In this research, we focused on emails. However, this channel is not tied to the model. In the future, the same model can optimize push notifications of mobile applications, in exactly the same manner as the current model does.

Research opportunities

As with any model, the model we presented in this research is a simplification of reality. The main impact is that, compared to real life, the model can choose between more actions. In reality, not every action can be undertaken in every time period. This can be improved by further restricting the action set, based upon the state. For example, incorporating the previous action in the state and restricting the action set based on this previous action.

Furthermore, the estimate of transition probabilities can be improved. At the moment, this estimation is based upon counting frequencies. However, when transitions are not observed, or observed infrequently, this estimation is unreliable and these transitions are filtered. This leads to a further restricted state space. Instead of removing these transitions, we could initialize a default probability from transitioning from a state to any other state. Or we could use machine learning techniques to estimate these probabilities, as a transition probability might say something about the transition probability of a similar action.

REFERENCES

[1] N. Sharma and P. Patterson, “The impact of communication effectiveness and service quality on relationship commitment in consumer, professional services,” *Journal of Services Marketing*, vol. 13, 1999.

[2] F. Ricci, L. Rokach, B. Shapira, and P. Kantor, *Recommender Systems Handbook*. Springer, 2011.

[3] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, 2005.

[4] M. Pazzani and D. Billsus, “Content-based recommendation systems,” in *The adaptive web*. Springer-Verlag, 2007, pp. 325–341.

[5] G. Adomavicius, B. Mobasher, F. Ricci, and A. Tuzhilin, “Context-aware recommender systems,” *AI Magazine*, 2011.

[6] G. Shani, R. Brafman, and D. Heckerman, “An MDP-based recommender system,” *Journal of Machine Learning Research*, no. 6, 2005.

[7] S. Bhulai and G. Koole, “Stochastic optimization,” September 2014.

 ISBN: 978-1-61208-741-2