ISSN : 2407 - 3911

DESIGN AND ANALYSIS OF UTILIZING POMDP IN ONLINE RESERVATION SYSTEM

Said Al Faraby
Teknik Informatika
Telkom University
Jl. Telekomunikasi Terusan Buah Batu Bandung 40257
Said.al.faraby@gmail.com

Abstract

System personalization is a smart way to handle interaction with various types of customers that come with different preferences. In this project we attempt to optimize the sequential decision processes during the interaction between an online hotel reservation service and its customer. We use POMDP framework to determine the best response for the system. POMDP is suitable for this problem because it takes into account the non-observability of the customer preferences, and the uncertainty of the observations that come from the customer. We present our findings regarding the potential improvement by utilizing belief information, and the sufficient level of observation certainty in order to keep the model considerably useful. We also show the resulting policies trees with the belief information in order to clarify the intuition behind the actions planned.

Keywords:

POMDP, optimization, decision, online reservation.

I. INTRODUCTION

In this project we focused on the interaction between online systems with their customers in an e-commerce setting, for instance, online bookstore and online hotel reservation service. Throughout this project we will use the latter instance as the study case for which we try to build a model.

Different customers with different intentions interact with the reservation website at the same time, even though they search for the same place and the same time, but they might be looking for different type of room, price, and so on. Therefore, giving back the same list of hotels might not be the best solution to help both customers finding the rooms they want.

Online hotel reservation services have already implemented various kind of strategies to help and encourage customers to find their desired hotels, one of the strategies used by a well-known booking website is displaying a badge to some chosen hotels. A badge is an eye catching icon indicating an excellent choice for either '*family*', '*solo*', '*couple*', '*group-of-friends*', or all of them in general as called by '*hotel of the year*'. By seeing these badges, customers are expected to be interested and convinced to book the hotels. However, this strategy can be counterproductive if the chosen badge to be shown are different from the customer-type[1]. Therefore, in order to provide a more relevant information to the customers, we need to have such a knowledge about what type of customers are the system interacting with, whether they are solo, couples, and so on, and what is the optimal action for the system at each step of the interaction.

Unfortunately, those customers-types are similar to intention, they cannot be obtained merely by an observation. Since this is non-observable problem, exact types of customers are very difficult to acquire. Nevertheless, such an estimation or belief over those types are feasible to be established via interaction between the system and the customers. One trivial interaction but could provide a strong indication is by asking directly to the customers, but sometimes it is not a good option. Asking too many questions might be annoying and frustrating for the customers, or they might refuse to answer because the questions are too personal to be recorded, or it can be any other reasons. In this situation, in order to still be able to build the estimation, we need to ask the customers indirectly, particularly by performing sequence of actions and observations, and then doing some reasoning to update the estimation accordingly. Moreover, in this case the observations are noisy, in the sense that the same observation can be received from two or more different types of customer, thus makes the problem possesses uncertainty property.

In some previous works, modelling user intention has been used in spoken dialogue management systems ((Roy, et.al, 2000), (Thomson and Young, 2010)). In those works, POMDP framework were employed to predict user intention in order to handle noisy and ambiguous speech utterances. The other more relevant research in e-commerce setting were attempted by the Advisor POMDP (Regan, et.al, 2005) and the SALE POMDP (Irissapane, et.al, 2014) models, where they try to

[1] We will use the term 'customer-type' to refer the type of room which is intended by customer to book, i.e. solo-type room.

optimally select sellers in e-marketplaces setting by modelling belief over sellers and advisors.

In this project we consider formal models that online systems can use to govern the interaction with their customer. For instance, in the case of booking.com, maintaining some belief over the customers-types will require reasoning over the process of interaction, in addition there is the uncertainty property inherent in the observations which come from the customers' acts and the partial observability property characterized by the hidden states of the true customers-types. By considering those two properties, we choose POMDPs as the framework to investigate and model the problem.

.

II. BACKGROUND

Formally, *A Partially Observable Markov Decision Process* framework (Kaelbling, et.al, 1998) can be described as tuple $\langle S, A, T, R, \Omega, O \rangle$, where S is a finite set of states of the world, A is a finite set of actions, and is a finite set of observations of the world that are possibly experienced by the agent. POMDP is a sequential decision processes framework, at each time step a world has a state $s \in S$ where the agent live in. The agent then takes an action $a \in A$ which triggers a transition from the current state to a new state $s' \in S$. Unfortunately, in this case the agent cannot observe the state, instead it receives an observation $o \in \Omega$ that can help the agent to maintain its beliefs over the hidden states. The probability of arriving at a new state $\Pr(s'|s,a)$ is determined by a *state-transition function* $T: S \times A \to \prod(S)$, and the probability of receiving a particular observation $\Pr(o|a,s')$, is defined by an *observation function* $O: S \times A \to \prod(\Omega)$. Afterward, the agent will receive an immediate reward $R(s,a)$ which is generated by a *reward function* $R: S \times A \to \mathbb{R}$.

Finally, the agent needs to update its current beliefs about the hidden states. The distribution of the new beliefs for every possible end state *s'* is calculated according to the following rule:

$$b'(s') = \frac{\Pr(s', o|b, a)}{\Pr(o|b, a)} = \frac{\Pr(o|a, s')}{\Pr(o|b, a)} \sum_{s} \Pr(s'|s, a)\, b(s),$$

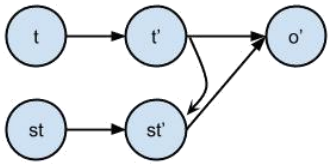
where $\Pr(o|b,a)$ denotes a normalization constant.

For the first update, the agent will use an initial belief b_0 that constitutes a prior over the states distribution. Whenever such prior knowledge does not exist, a uniform distribution is commonly used.

The whole idea of modelling the problem as POMDP framework is to act optimally based on the belief as an estimation of the true state. A *policy* π in POMDP, is a function that maps a belief into an action $\pi(b) \to a$. A quality of policy is usually measured by a value function $V_\pi(b)$, which indicates the expected total reward of following policy starting from b:

$$V_\pi(b) = \mathrm{E}\left[\sum_{t=0}^{h-1} \gamma^t R(s, a, s')|\pi, b\right]$$

(a) DBN



(b) Transition of solo action

st'	t'	Pr(st'|st,t',A_{solo}) un	bk
un	SL	0.6	0.4
un	CP	0.95	0.05
un	GR	0.95	0.05
un	FM	0.95	0.05
bk	*	0.0	1.0

(c) Transition of query action

st'	t'	Pr(st'|st,t',A_{query}) un	bk
un	*	0.8	0.2
bk	*	0.0	1.0

(d) observation of solo action

st'	t'	Pr(o'|st',t',A_{solo}) SL	CP	GR	FM	Bk
un	SL	0.601	0.133	0.133	0.133	0.0
un	CP	0.1	0.3	0.3	0.3	0.0
un	GR	0.1	0.3	0.3	0.3	0.0
un	FM	0.1	0.3	0.3	0.3	0.0
bk	*	0.0	0.0	0.0	0.0	1.0

(e) observation of query action

st'	t'	Pr(o'|st',t',A_{solo})				
		SL	CP	GR	FM	bk
un	SL	0.601	0.133	0.133	0.133	0.0
un	CP	0.1	0.3	0.3	0.3	0.0
un	GR	0.1	0.3	0.3	0.3	0.0
un	FM	0.1	0.3	0.3	0.3	0.0
bk	*	0.0	0.0	0.0	0.0	1.0

Figure 1. DBN and CPT of the POMDP transition and observation functions for solo-specific action and query action. t and st are the customer-type and booking status respectively.

III. THE POMDP MODEL

In this section we will explain our specification of POMDP model by referring to the real situation in an e-commerse setting, particularly an online hotel reservation system.

III.1 States

A state is composed by two variables which are the customer-type and the status of current transaction. Customer-type will never change, but the status of current transaction can switch from *unbooked* to *booked* when the customer decide to book a room. Let $\mathbb{T}$ be the set of customer-type, and *book* is a variable denoting the status of cur-rent transaction, then a state is a tupple ***s*** = < ***t, st*** >, where $t \in \mathbb{T}$ is a certain type of customer which is interacting with booking system, and ***st*** is the transaction status. An episode of interaction always starts with ***st*** = ***unbooked*** and stops after the status variable changes to ***booked***, which is the terminal states. Note that, we model each customer separately, thus each of them has their own states, and there is no connection between those states. Figure 1 illustrates the state in the form of DBN graph.

III.2 Actions

In this online reservation problem, action can be anything from whatever the system can do as ways to interact with its clients. In our case, we categorize an action into two purposes, 1) to help and encourage customers to find and book their rooms faster, 2) to get more information about the customers-types. Presenting a list of hotels, or choosing a certain type of badge to be shown are examples of possible actions that satisfy both of the purposes. Other kind of actions that probably cannot influence the customers directly but can be useful for estimating the customer-type, is called as information-gathered actions. For instance, showing pictures of interesting places or facilities in the hotel that correlate with specific customer-type, could help the system to infer what kind of room-type that the customers are looking for.

However, considering each list of hotels or pictures as a single action will potentially lead to combinatorial explosion of the number of actions in the framework. This makes modelling the problem is very difficult and the intuition behind the resulting policy will be too obscure to be understood. Instead, we can categorize the actions into a simpler yet intuitive form, where we associate actions to the customer-types. For example, *'solo-specific'* action can be represented by a list of hotels dominated by hotels with solo badge and a few numbers for the other types. In addition, we include a query action as an information-gathered action to get more evidences about the customer-type, where in the real situation we could offer a list containing various types of hotels.

III.3 Transitions

States transitions only happen when status of transaction changes from *unbooked* to *booked*. The probabilities of state transitions are characterized by the current state and the current action taken by the system. We assume that taking a specific-action that matches the customer-type will have a higher chance of getting booking than taking other types of specific actions. Moreover, a query action has a lower probability of booking. Figure 1.b shows an example of transitions probabilities triggered by a specific-action solo. Transition probabilities for other specific-actions looks pretty much the same, except different raw of pair of 0.6 and 0.4. On the other hand, transition probabilities of taking query-action seems independent of the customer-type, as described in figure 1.c This is because we consider that query action provides an equal proportion of different type of hotels, and thus we assume it has the same chance of booking for any of customer-types. Finally, *bk* which is booked status, determines the terminal states or absorbing states, which is denoted by probability 1.0 of staying at that states.

III.4 Observations

Selecting the observations that have a good correlation with the true state is a key to maintain a reliable beliefs information. There are abundant alternatives of observations that the system receives from the customers, but starting with a simple yet representative list of observations should be a reasonable way to do. Instead of considering any user act as a single observation, we cluster the observations into a more representative form of state-related observations, which are solo, couple, group, family, and booked observation. These state-related observations are kind of observations that are most-likely to be received from its corresponding customer-types. Specifically for booked-observation, it perfectly reflects the status of the current transaction. Figure 1.d and 1.e are examples of observation probabilities given the system took solo-action and query-action

respectively. Those observation functions has 0.601 of observation certainty, which is the probability of getting the corresponding observation according to the current state. The sufficient value of observation certainty will be investigated in the experiment section.

III.5 Rewards

We use a simple form of reward by specifying the reward value e.g. 10, only when the status of transaction changes from *unbooked* to *booked*. There is no cost or negative reward for taking any action, but the effects are already compensated through transition and observation functions.

IV. Experiments and Results

In this section we present some experimental results regarding our POMDP model. In order to evaluate the model, we utilize APPL toolkit which is based on SARSOP(Kurniawati, et.al, 2008), by computing optimal policy and run simulations on that computed policy. We also built our own parser and simulator to estimate the value of other policies used in the following experiments.

IV.1 Potential Improvement

In order to show potential improvement that could be achieved by making use of belief information, we did an experiment by comparing expected discounted reward of different kind of policies. Random and Fix policies are the most known instances of non-belief policies, which are policies that does not require beliefs state. Random policy will obviously choose an action randomly, and the fix policy in this case always performs query-action.

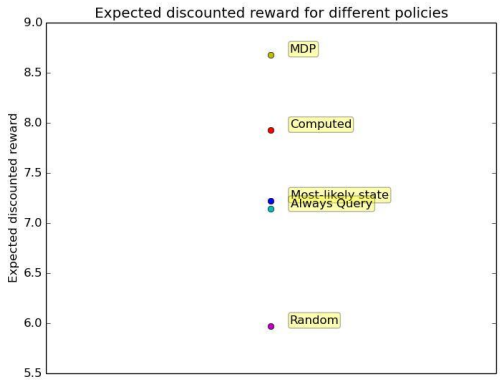


Figure 2. Expected discounted reward of different policies

On the other hand, most-likely-state policy and computed policy are kind of policies that need the belief information in order to choose an action. Most-likely state policy is probably the simplest way of exploiting belief, it chooses an action as it were in the most-likely current state, which is determined by the distribution over the belief state. Finally, we include MDP policy into the comparison in order to show the upper-bound value of our POMDP model. For this experiment, the setting of POMDP model is the same as described in the figure 1, except the observation certainty was set to 0.9, which represent a very ideal type of observations.

The expected discounted reward values for different policies are illustrated by figure 2. It is interesting to see, how the computed policy outperforms the non-belief policies significantly. Moreover, even the simple most-likely-state policy already performs better than the non-belief policies, even though not as significant as the computed policy. The graph suggests that utilizing the beliefs information could improve the expected reward, especially by computing the optimal policy based on the belief, which shows a considerable increase over non-belief policies. Meanwhile, the value of MDP policy as the upper-bound value, is apparently not too far away from the computed POMDP policy, considering the uncertainty that the POMDP policy has to overcome.

IV.2 The impact of observation certainty

As mentioned in the previous section, observations in this particular problem can be derived from many things. Even though we have categorized the observations into the state-specific categories, the certainty level of those state-specific observations can also be vary. Choosing the right observations is important for POMDP policy in or-der to obtain a significant improvement over the baseline policies that do not depend on observation such as random and fix policy. Here we report the impact of the observation certainty to the policies values. Observation certainty in our model is defined by the probability of getting the right observation according to the true customer-type, for example probability of getting solo-observation in state ***<solo, unbooked>.***

The results of the experiment are described in figure 3. It is clearly seen that the observation certainty will only affect the policies that exploit the belief information, where the updating processes of the beliefs depend on the received observations. Furthermore, it is also expected that those policies' values will decrease along with the decreasing of the observation certainty.

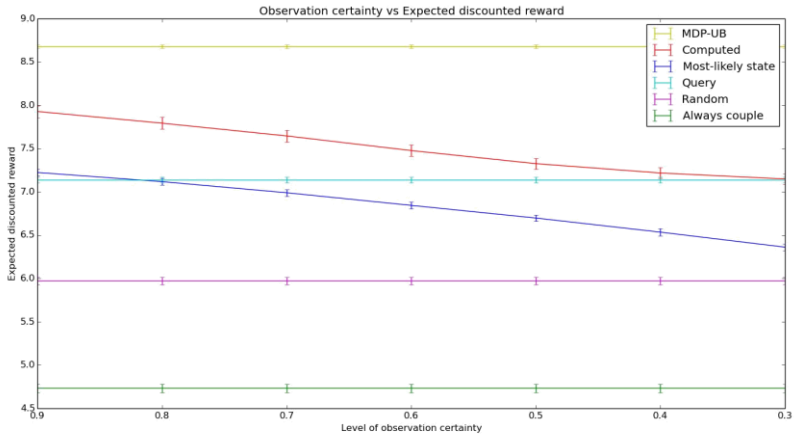
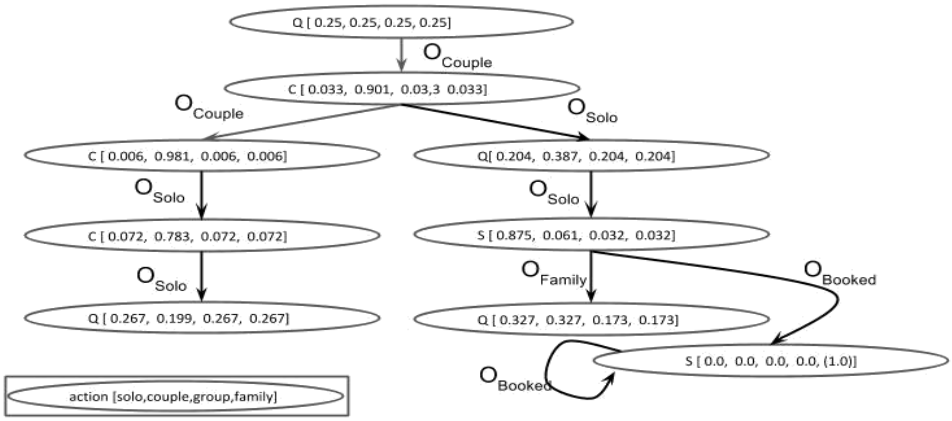
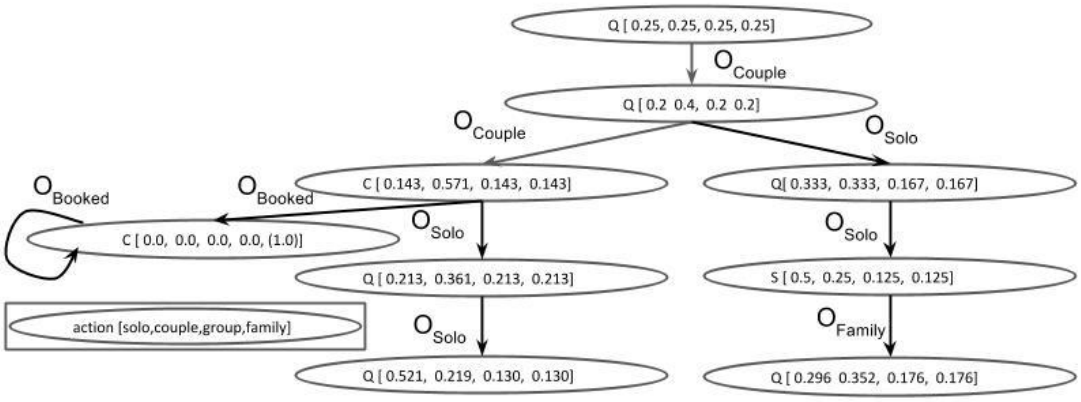


Figure 3. Expected discounted reward of different policies with different observation certainty values



(a) (Partial) High observation certainty policy



(b) (Partial) Low observation certainty policy

Figure 4. Comparison between policy resulting from a high observation certainty setting (0.9) and a low observation certainty setting (0.4). Each nodes represent current belief on each customer-type and action planned for that belief.

.

Since we want the policies utilizing the beliefs to be significantly better than the non-belief policies, we need to estimate the minimum value of the observation certainty such that the expected reward of the belief-policies are sufficiently higher than the non-belief policies. As for the most-likely state policy, it requires a very high observation certainty of 0.9, in order to score higher than the query policy. On the other hand, our computed policy seems much more reliable by being able to perform considerably better than the query policy until the observation certainty is set to 0.5. The vertical bars on the graph denote the 95% confidence interval of the expected reward values resulting from multiple simulations. It is used to determine the statistical difference between the results.

Furthermore, our investigation on the resulting policies shows that the lower the observation certainty is, the closer the gap between the value of computed policy and query policy will be, and the more similar actions that are planned by those two policies.

IV.3 Resulting policies and belief update

Here we compare two policies resulting from different level of observation certainty. Figure 4.a illustrates a policy computed from a high observation certainty of 0.9, and figure 4.b from a lower observation certainty of 0.4. One of the salient differences is how many times query-action with the same resulting observation is needed in order to pick a specific-action

For a high observation certainty policy, once it get a specific observation (i.e. couple-observation) after taking the first query-action, its belief of being in the same state as the type of the observation received is very high (i.e. 0.901), so it plans to take the corresponding specification immediately. On the other hand, the policy resulting from a low observation certainty needs to see the same observation twice before it certain enough to pick a specification, otherwise it remains on the query-action. For instance, after taking the first query-action and get the couple-observation, its belief of being in state **<*couple, unbooked*>** is just 0.4, which is probably not certain enough in order to go for couple-action. After the second time its belief becomes 0.571 and sure enough to pick couple-action.

In addition, both policies start with query-action due to the fact that our initial belief is uniform. Moreover, the transition to the terminal state is depending on the customer's action, and it can happen anytime when the customers decide to book a room. If the customers book a room then booked-observation will received by the agent with probability 1.0, so the states transit to the terminal states also with probability 1.0, and will always remain at that state.

V. CONCLUSION AND FUTURE WORKS

The results suggest that utilizing belief information even in a very simple manner, such as the most-likely-state policy, could be useful to improve the expected reward over the non-belief policies such as random and query policy. Unfortunately it requires a very high observation certainty to do so. On the other hand, the computed policy perform much more better in handling the low observation certainty, however there is still a minimum level of observation certainty where it shows a great improvement. Moreover, the lower the observation certainty is, the more times information-gathered action needed before the agent can choose a specification. Finally, by knowing the minimum level of observation certainty, we could use it to select observations used for our model appropriately.

Regarding future works, increasing the complexity of the model by specifying more kind of observations and actions might bring the model closer towards the real environment, for example dividing the specific-observation into several level of certainty, such as strong solo observation and weak solo observation. Another possible improvement is exploiting the real data to define the observation and the transition functions.

REFERENCE

Irissappane, Athirai A., Oliehoek, Frans A., and Jie Zhang, (2014) "A POMDP based approach to optimally select sellers in electronic market-places". In: Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems. International Foundation for Autonomous Agents and Multiagent Systems. 2014, pp. 1329–1336.

Kaelbling, Leslie Pack, Littman, Michael L, and Cassandra, Anthony R., (1998) "Planning and acting in partially observable stochastic domains". In: Artificial intelligence 101.1 (1998), pp. 99–134.

Kurniawati, Hanna, Hsu, David, and Lee, Wee Sun, (2008), "SARSOP: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable Belief Spaces." In: Robotics: Science and Systems. Vol. 2008. 2008.

Regan, Kevin, Cohen, Robin, and Poupart, Pascal, (2005), "The Advisor-POMDP: A Principled Approach to Trust through Reputation in Electronic Markets." In: PST. Citeseer. 2005.

Roy, Nicholas, Pineau, Joelle, and Thurn, Sebastian, (2000), "Spoken dialogue management using probabilistic reasoning". In: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics. 2000, pp. 93– 100.

Thomson, Blaise, and Young, Steven, (2010),"Bayesian update of dialogue state: A POMDP frame-work for spoken dialogue systems". In: Computer Speech & Language 24.4 (2010), pp. 562–588.