Latent-variable MDP models for adapting the interaction environment of diverse users

Subramanian Ramamoorthy *
M. M. Hassan Mahmud
Benjamin Rosman
School Of Informatics
University of Edinburgh

Pushmeet Kohli
Machine Learning and Perception
Microsoft Research Cambridge

Abstract

Interactive interfaces are a common feature of many systems ranging from field robotics to video games. In most applications, these interfaces must be used by a heterogeneous set of users, with substantial variety in effectiveness with the same interface when configured differently. We address the problem of personalizing such an interface, adapting parameters to present the user with an environment that is optimal with respect to their individual traits - enabling that particular user to achieve their personal optimum. We model the user as a parameterised Markov Decision Process (MDP), wherein the transition dynamics within a task depends on the latent personality traits (e.g., skill or dexterity) of the user. A key innovation is that we adapt at the level of *action sets*, picking a personalized optimal *set* of actions that the user should use. Our solution involves a latent variable formulation wherein we maintain beliefs over the latent *type* of users, which serves as a proxy for the hidden personality traits. This allows us to compute a Bayes optimal action set which when presented to the user allows them to achieve optimal performance. Our experiments, with real and simulated human participants, demonstrate that our personalized adaptive solution outperforms any alternate static solution, and also other adaptive algorithms such as EXP-3. Furthermore, we show that our algorithm is most useful under high diversity in user base, where the benefits of *safe* initialization and *quick* adaptation (properties our algorithm provably enjoys) are most pronounced.

1 Introduction

Interactive interfaces are an important element of many modern systems. Natural user interfaces such as XBox gesture recognition have revolutionised the way in which we interact with video games and related applications. The underlying technology has also gone on to become the standard for mobile robots, especially where cost-effective solutions for human-robot interaction are necessary. In a different sense, many software applications require interactive interfaces. Search engine interfaces such as for Google or Bing, or photo-editing software such as Adobe Photoshop, involve numerous configuration choices that implicitly define the context of the interaction between the human user and a computational process.

Almost all of these applications must be deployed with large user populations, with substantial diversity in the performance that results from any given pair of user and configuration setting. For instance, even with a simple interface like joystick-based navigation within a video game or field robotics application, there is variety in dexterity and skill. A mismatch between a user's skill level and settings such as the sensitivity level at which one interprets a particular motion as a signal can

* **Technical Report version 2:** University of Edinburgh, 2013.

have substantial negative impact on the user's ability to perform a task. Sometimes the mismatches can be in assumptions about perceptual ability, e.g., how precisely aware a user is regarding the current computational context. As studies such as [1] have shown, perceptual limitations can have a disproportionately large effect on user's performance when the task involves interactive decisions in a dynamic environment. Similar effects occur in software applications, such as in the difference in performance between young children and skilled adults when presented with a number of choices by an interface.

The common issue that unifies such diverse applications is personalisation. The need is for automated ways to define the interaction environment so that it is best adapted to the specific user whom the system currently faces. Our particular interest is in adapting online, to personalise on the fly as the user interacts with the system, without long calibration phases and with only partial knowledge of the underlying traits that induce variability.

The objective of this paper is to present such a model of personalisation and an algorithm for optimally adapting the corresponding *action set* that the user must work with. The effectiveness of such a model and algorithm can be characterised by a few key properties - *safety*, i.e., ensuring that individualised adaptation does not lead to worse performance than a pre-configured statically optimal setting, *efficiency*, i.e., learning to obtain settings that enable the user to achieve their maximal performance, and *speed*, i.e., the ability to adapt within a few episodes which roughly corresponds to users' patience. With this in mind, we present a theoretical analysis of our proposed algorithm, proving that these properties are indeed attained. Also, we present empirical evaluation with real and simulated human participants, to further validate our claims.

1.1 Related work

The problem of learning to adapt to user behaviour is garnering increasing attention from researchers in different domains. In one formulation, the learning agent may be assisting a user whose goal is unknown. For instance, [2] present the hidden-goal MDP to model problems such as that faced by a doorman who must anticipate when and where to go in order to help someone. In general, this is a computationally hard problem and fairly strong assumptions are required to achieve tractable results, such as that the only unknown is the goal state of the user. Indeed, this problem has a longer history in the literature on plan recognition [3].

Our focus is different in that our problem is not that of trying to predict the user's future state alone, based on the past (as in the doorman example above); instead, we try to infer latent 'personality traits' or skill/preferences of the user in order to shape their environment - to enable them to do better in their chosen task. In this sense, our work is related to recent work on personalisation [4, 5].

In terms of technique, there is some similarity with models involving partially-observable and mixed-observable Markov Decision Processes [6] [7]. An example of the use of such models in robotics is the work on intention-aware motion planning [8] [9], to accommodate changes in the environment caused by purposeful agents, e.g., pedestrians who wander in erratic ways. While solving a related problem to the above, we adopt a different technical formulation in terms of a Bayesian procedure that allows us to present novel theoretical results and also to better handle large state-action spaces.

Our approach also relates to work in recommender systems and web applications, where it is common to model user behaviour as a Markov Chain, adapting recommendations based on characteristics of these dynamics [10]. Sometimes, this problem is given a sequential decision making flavour [11], [12] such as by using a bandit learning algorithm to pick optimal recommendation strategies over time. In our work, we need more expressive models of the user herself, whom we model as a parameterised-MDP, based on which we devise a novel sequential decision making algorithm.

The notion of designing a customized user interface has been studied in the HCI community, e.g., in [13], where one finds ways to automatically synthesize user interfaces tuned to user abilities. The process involves calibration with a battery of tasks, based on which one obtains information necessary to pose the interface design problem as one of combinatorial search. While we take inspiration from such work, the focus of our own work is different. As we are motivated by quick online adaptation to different users, the harder problem for us is to identify types of users whose relevant traits may be latent and unobserved.

2 A Model of User Interaction

In this section we present the technical setup of our model for shaping an interaction environment. The next section builds on this by presenting the learning algorithm. In our model, the user is an *agent* acting according to a Markov decision process (MDP) [14] and invoking a correspondingly optimal policy [1]. A finite MDP $\mathcal{M}$ is defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, T, R, \gamma)$ where $\mathcal{S}$ is a finite set of states, $\mathcal{A}$ is a finite set of actions and $\mathcal{R} \subset \mathbb{R}$ is the set of rewards. $T(s'|s,a)$ is the state transition distribution for $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$ and $R(s,a)$, the reward function, is a random variable taking values in $\mathcal{R}$. Finally, $\gamma \in [0,1)$ is the discount rate. A (stationary) policy π for $\mathcal{M}$ is a map $\pi : \mathcal{S} \to \mathcal{A}$. The Q function $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ of a policy π is defined by $Q^\pi(s,a) = \mathbb{E}[R(s,a)] + \gamma \sum_{s'} T(s'|s,a) Q^\pi(s', \pi(s'))$. The value function for π is defined as $V^\pi(s) = Q^\pi(s, \pi(s))$. An optimal policy π^* is defined as $\pi^* = \arg\max_\pi V^\pi$ – the Q function is given by $Q^*(s,a) = \mathbb{E}[R(s,a)] + \gamma \sum_{s'} T(s'|s,a) \max_a Q^*(s',a)$. For the optimal policy the value function is denoted by V^*. The agent then chooses action $\arg\max_a Q^*(s,a)$ at state s. We assume, without loss of generality, a fixed starting state s° for a MDP and define $V^\pi \triangleq V^\pi(s^\circ)$.

The user skill level/type determines the transition function of this MDP – this models the fact that, for instance, a less skilled user will have a different way of transitioning between states (e.g., being more variable) than an expert. Unlike in standard applications of optimizing policies for given MDPs, we will be changing the action set as the interaction proceeds in episodes, which is the main point of the interaction shaping process. The goal of the learner [2] will be to choose action sets so as to maximize the future expected discounted reward of the agent. So, the search space for the main learning algorithm in this paper is a type-space, rather than the space of paths the user actually traverses, and the value of an type is the value of its optimal policy given the type.

More precisely, we are concerned with a class of MDPs M parametrized by a parameter τ:

$$\mathsf{M} \triangleq \{((\mathcal{S}, \mathcal{A}, \mathcal{R}, T(\cdot;\tau), R, \gamma) | \tau \in \mathsf{Sk}\} \tag{1}$$

The MDPs differ only in the transition function $T(s'|s,a;\tau)$, which now depends on τ (the skill level of the user). Sk is the set of all types, τ. For instance, the user controlling a joystick corresponds to a MDP. Furthermore, a poorly skilled user will have poor control over the joystick which means that she will only be able to make coarse moves (regardless of the sensitivity of the joystick). Whereas, a highly skilled user will be able to execute sharp curves if given an appropriately sensitive joystick. This implies that the MDP corresponding to the task for each user type has different transition functions (for example, transitions with high and low entropy for low and high skilled users respectively).

We further assume that the action space $\mathcal{A}$ is large, but is partitioned, where AS is the set of all the cells of the partition. That is, if $\alpha, \alpha' \in \mathsf{AS}$, then α and α' are disjoint; and $\bigcup_{\alpha \in \mathsf{AS}} \alpha = \mathcal{A}$. We now only allow policies that take values in a single $\alpha \in \mathsf{AS}$ (we say the policy is restricted to α). Continuing with the joystick example, different action sets correspond to changing the sensitivity of the joystick. With a higher sensitivity joystick, the user now has the option (i.e. actions) to execute finer grained moves. Of course, whether she will be able to execute them depends on her skill level (i.e. the transition function ultimately also depends on the user type). The value function of a policy π now depends on τ and is denoted by V^π_τ. For a given α, we denote $V^\alpha_\tau \triangleq \max_\pi V^\pi_\tau$ and denote by $\pi^*_\alpha \triangleq \arg\max_\pi V^\pi_\tau$, where the $\max$ is over πs restricted to α.

Our learning problem can now be defined as follows. The agent solves a sequence of MDPs from the set M in collaboration with the learner – that is the goal of both the agent and the learner is to maximize the future expected discounted reward of the agent. At the beginning of each phase of the problem, nature chooses the type τ^* according to a distribution $K_1(\tau)$ and in response, the learner is required to choose the action set α. The agent can now only use policies restricted to α. The learner cannot observe the true τ^* but knows the value of T and R for each τ. The agent knows its own type and also knows the value of T and R. Once α is given, the agent learns and acts according to the policy π^*_α.

[1] We make the rationality assumption that the user is an MDP. We recognize that in some applications this is not the case. Handling this requires an extension to our model: the MDP is replaced with an alternate behaviourally motivated model of choice, without fundamentally altering anything else in our framework.

[2] We refer to learning algorithms, which choose the action sets as the *learner*. The human user is often referred to as the *agent*.

The role of the learner is as follows. At any point in time, the learner can step in and change the action set from α to a new action set α'. In response the agent starts acting according to $\pi^*_{\alpha'}$ instead. The goal of the learner is to choose the action set $\arg\max_\alpha V^\alpha_\tau$. In terms of the joystick example, this means that at the beginning some user of unknown skill (τ^*) comes in to use the interface. She operates the interface to the best of her ability given the sensitivity. Given her actions, the learner tries to estimate the skill level and based on that either increases or decreases the sensitivity.

Given the above setup, the optimal action set and a-priori optimal action sets are defined, respectively, to be:

$$\alpha^* \triangleq \arg\max_\alpha V^\alpha_{\tau^*}, \qquad \alpha_* \triangleq \arg\max_\alpha \mathbb{E}_{K_1(\tau)}[V^\alpha_\tau] \tag{2}$$

In the next section, our goal will be to derive an *a-posteriori optimal* action-set selection policy, that is, a policy that adaptively and optimally chooses the action-set that is optimal according to some well-defined and reasonable criteria and also eventually converges to α^*.

3 Determining Agent Type

In the following we present our approach to adaptively determine the agent type through repeated interactions. We first describe our criteria, Bayes optimality, for choosing action sets and then describe the algorithm that uses the Bayes optimal action set. We then present convergence results that show that our algorithm finds the true type τ^* and plays the optimal action set α^* in the limit. We additionally provide convergence rates for our algorithm that depend on the environment parameters.

3.1 Bayes Optimal Action Set

We will assume the knowledge of the set of user types $\mathsf{Sk} = \{\tau_1, \tau_2, \cdots, \tau_n\}$. In the experiments, we acquire these types during an initial training phase by interacting with users. This is discussed further in Section 4. Assume that at time step t of a particular interaction session we have observed a state-action sequence $sa_{0:t} \triangleq s_0 a_0 s_1 a_1 \cdots s_t$ (with $sa_{0:0} \triangleq s_0$). Given this session, the likelihood of a type τ_i can be computed as $L(\tau_i|s_{0:t}) = \prod_{i=0}^{t-1} T(s_{i+1}|s_i, a_i; \tau_i)$, while its posterior probability is defined as $Pr(\tau_i|sa_{0:t}) = L(\tau_i|sa_{0:t})W(\tau_i)$ where $W(.)$ is a prior distribution over Sk.

Having defined the posterior distribution over types, our approach will be, at step t, to choose the Bayes optimal action-set α_{BO} *i.e.* the action set maximizing expectation of V^α_τ with respect to the posterior over Sk. More formally,

$$\alpha_{BO}(sa_{0:t}) \triangleq \arg\max_\alpha \sum_i Pr(\tau_i|sa_{0:t}) V^\alpha_{\tau_i} \tag{3}$$

For the empty sequence $\emptyset$, we define $\alpha_{BO}(\emptyset) \triangleq \arg\max_\alpha \sum_i W(\tau_i) V^\alpha_{\tau_i}$.

3.2 Learning Algorithm

Our learning algorithm, Bayesian Environment Adaptation with Types (BEAT) is listed as Algorithm 1. At each step, BEAT observes the agent action and updates the likelihood (and hence posterior) of each type. Additionally, every k steps, it chooses a new action set. With probability $1 - \epsilon_{t/k}$ it chooses the Bayes optimal action set and with probability $\epsilon_{t/k}$ it chooses an action set at random. The exploration terms satisfy the condition of $\sum_n \epsilon_n = \infty$ and $\lim_{n\to\infty} \epsilon_n = 0$ (for instance $\epsilon_n = 1/n$). To see that it is necessary to explore, consider the following example. There are two types τ_1, τ_2 with τ_1 the true type, and two action-sets α_1, α_2, with α_1 being the optimal for τ_1. Under α_2 both τ_1 and τ_2 have identical transitions and hence $V^{\alpha_2}_{\tau_2} = V^{\alpha_2}_{\tau_1}$. In α_1, τ_1 and τ_2 have different transition distributions, and $V^{\alpha_1}_{\tau_2} \ll V^{\alpha_1}_{\tau_1}$. So under the uniform prior, the Bayes optimal action-set will be α_2, and under α_2, both τ_1, τ_2 will have identical likelihoods and hence identical posteriors. Therefore, because of the condition $V^{\alpha_1}_{\tau_2} \ll V^{\alpha_1}_{\tau_1}$, $\alpha_{BO}(s_{0:t})$ would always be α_2.

3.3 Convergence of the Algorithm

In this section we show that in the limit, BEAT plays only the optimal action set α^* and provide corresponding convergence rates. From the definition of the BEAT algorithm, the probability of

Algorithm 1 Bayesian-Environment-Adaptation-with-Types($\mathsf{M}, term, k, \{\tau_i\}_{(i)}, W, \{T(s'|s,a;\tau_i\}_{(i)}$)

1: **Input:**The set of MDPs M , Task termination condition $term$, action-set selection period k, a set of types τ_i, W a prior over types.
2: **Initialize:** $\forall \tau_i$, likelihood $L_0(\tau_i) = 1$. ϵ_n is a sequence with $\sum_n \epsilon_n = \infty$ and $\lim_{n\to\infty} \epsilon_n = 0$.
3: Compute $V^{\alpha}_{\tau_i}$ for each $\alpha \in \mathsf{AS}$ and type τ_i.
4: Let s_0 be the initial state and $t \leftarrow 0$.
5: **while** $term = false$ **do**
6: Observe agent action a_t, reward r_t and next state s_{t+1}.
7: Update likelihood of each type: $L_{t+1}(\tau_i) = L_t(\tau_i) \times T(s_{t+1}|s_t, a_t; \tau_i)$.
8: **if** $t \mod k = 0$ **then** with probability $\epsilon_{t/k}$ set $\alpha_{t+1} =$ an action-set at random, and with probability $1 - \epsilon_{t/k}$ set $\alpha_{t+1} = \alpha_{BO}(sa_{0:t+1})$ (defined in (3)). **Else** $\alpha_{t+1} \leftarrow \alpha_t$.
9: $t \to t + 1$.
10: **end while**

observing a state action sequence $sa_{0:t}$ is given as

$$Pr(sa_{0:t}; \epsilon, \tau^*) \triangleq \sum_{\alpha_{0:t-1}} \prod_{i=0}^{t-1} T(s_{i+1}|s_i, a_i; \tau^*) Pr(\alpha_i|sa_{0:i}) I_{[a_i = \pi^*_{\alpha_i}(s_i)]} \tag{4}$$

The outer sum is over all possible action-set sequences of length $t-1$ and $Pr(\alpha_i|sa_{0:i})$ is the probability that α_i is chosen in line 8 in BEAT. Given this, the first theorem below shows that BEAT finds the true type $Pr(\cdot|\epsilon, \tau^*)$ with very high probability (proof in supplementary material).

Theorem 1. *Let τ be any type such that there exists at least one action-set $\bar{\alpha}$ for which $T(\cdot|s, \pi^*_{\bar{\alpha}}(s); \tau) \neq T(\cdot|s, \pi^*_{\bar{\alpha}}(s); \tau^*)$ for at least one state for any length k path of the optimal policy $\pi^*_{\bar{\alpha}}$ with probability at least $\beta > 0$. Then,*

$$\lim_{t\to\infty} \frac{L(\tau^*|sa_{0:t})W(\tau^*)}{L(\tau|sa_{0:t})W(\tau)} = \infty \;\; Pr(\cdot; \epsilon, \tau^*) \textit{ almost surely} \tag{5}$$

The theorem says that the posterior of the true type τ^* becomes arbitrarily larger than the posterior of any other type, provided that there exists at least one action-set $\bar{\alpha}$ for which the the types are different and this difference is observed with at least some positive probability β. If this is not true, then clearly τ^* and τ are equivalent and there is no point comparing τ^* and τ (see Appendix A in supplementary material). The proof is different from standard proofs of Bayesian posterior consistency because BEAT is actively choosing action sets and the true type τ^* and non-true type τ may be identical in many action sets. As such it is quite non-trivial. The corollary to the above theorem establishes the rate at which the posterior convergence takes place (proof in supplementary material).

Corollary 1. *Assume that the hypothesis of Theorem 1 is true for type τ. Let $PE_\tau(t, m)$ be a (ϵ_n dependent) lower bound on the probability that after t steps the action set $\bar{\alpha}$ was observed m times. Then with probability at least $PE_\tau(t, m)$, $\frac{L(\tau^*|sa_{0:t})W(\tau^*)}{L(\tau|sa_{0:t})W(\tau)} > PE_\tau(t, m)m\eta\beta$, where η is a lower bound on the KL divergence $D[T(\cdot|s, \pi^*_{\bar{\alpha}}(s); \tau^*)||T(\cdot|s, \pi^*_{\bar{\alpha}}(s); \tau)]$ whenever $T(\cdot|s, \pi^*_{\bar{\alpha}}(s); \tau) \neq T(\cdot|s, \pi^*_{\bar{\alpha}}(s); \tau^*)$.*

So the corollary shows that the rate of convergence depends directly on our exploration frequency (via ϵ_n dependent $PE_\tau(t, m)$), and how easy it is to distinguish between correct and incorrect types (as determined by η and β). Our next theorem almost completes the convergence analysis by showing that in the limit only the optimal action set is chosen (proof in supplementary material).

Theorem 2. *If each $\tau \neq \tau^*$ satisfies the hypothesis of Theorem 1, and if α_t is the action set chosen in BEAT at step t, then $\lim_{t\to\infty} \alpha_t = \alpha^*$ in $Pr(\cdot; \epsilon, \tau^*)$.*

So the theorem shows that the BEAT algorithm chooses the optimal action set in the limit. Our final theorem also gives a convergence rate and completes the analysis (proof in supplementary material).

Theorem 3. *Define $\hat{l} \triangleq |\mathsf{Sk}| - 1$. For a fixed δ, let $t_{\delta,m} \triangleq \min\{t|\forall \tau, PE_\tau(t, m) > 1 - \delta/\hat{l}\}$. Let η_* be the minimum over all $\tau \in \mathsf{Sk}$ of the η parameter defined in Corollary 1. Let $M = \frac{R_{\max}\hat{l}}{(1-\gamma)V^{\alpha^*}_{\tau^*}}$*

$(\eta_*\beta(1-\delta/\hat{l}))^{-1}$. *Then if each* $\tau \neq \tau^*$ *satisfies the hypothesis of Theorem* 1*, then with probability at least* $(1-\delta)$, $\alpha_{BO}(sa_{0:t_{\delta,M}}) = \alpha^*$.

Recall that $R_{\max}$ is the upper bound on per step reward. Then, this theorem shows that for M sufficiently large (a function of $R_{\max}$ and the number of types $|\mathsf{Sk}|$), with high probability the optimal action is chosen by the algorithm.

4 Experiments

In this section we present two sets of experiments in a video game domain to illustrate our method. In the first set, the user is a robot/artificial agent, while in the second set the users are humans. In the following, we describe the comparison algorithms and then discuss the results in detail. In our experiments, we compare BEAT with α^* (optimal action set (2) , α_* (a-priori/population optimal action set (2)), and the EXP-3 algorithm [15, 16] for non-stochastic multi-armed bandits (NSMB). This latter framework is a natural fit for addressing the problem of action-set selection. In the NSMB problem, there are c *arms* where each arm i has a *payoff process* $x_i(t)$ associated with it. The learner runs for T steps and at each step t needs to *pull*/select one of the arms $f(t)$ and his payoff is $x_{f(t)}(t)$. Additionally, the learner only gets to view the payoff of the arm $f(t)$ it has chosen. The goal of the learner is to minimize its *regret* with respect to the best arm, that is minimize the quantity $\max_i \sum_{t=1}^{T} x_i(t) - \sum_{t=1}^{T} x_{f(t)}(t)$. In general it is not possible to minimize this in any meaningful sense. An optimal algorithm in the general case, for minimizing the *expected regret* was developed in [15], called the EXP-3 algorithm. In our experiment, each arm corresponds to an action-set run for k steps and the payoff is the total discounted reward obtained in those k steps.

4.1 Experiment Setup

Design. The video game domain is shown in Figure 1. The goal of the user is to move a ball (red circle) from a fixed start location to the goal location (green rectangle). The ball moves with a constant speed z and at each step the user can choose to change the direction of motion into one of N, S, E, and W. Her loss (negative reward), received when she reaches the goal, is the normalized time taken + normalized number of collisions. During play, the learner chooses the speed to help minimize the loss. We capture two types of limitations in user behaviour. The size of the ball reflects limits to perceptual ability, with large ball sizes indicating reduced acuity. The noise of the motion of the ball reflects limits to motor ability, such as intrinsic jitter in the way they use the device. Combinations of larger ball+noise imply less skill.

Parameters. The state space is the location of the ball at each time step (the game field was 1000×1000). Each action is a move by z-units in a cardinal direction (N, S, E, W), where $z \in \{30, 40, 50, 60, 70\}$. Each action set α_z corresponds to all the motions at speed z. The direction at step t is the most recent direction chosen by the user at $t' \leq t$ (so, the actual play of the user is modelled in the MDP as the user choosing the same action at every step as her last choice of direction). There was a constant amount of noise (on top of noise due to type) with each action, so the ball moves in the given direction followed by (discretized) Gaussian motion (perpendicular to the direction of motion) with mean 0, and $\sigma \in [0.3, 8]$ (chosen randomly for each episode). There were 5 different types: ball size $b \in \{2, 5, 10, 20, 40, 60\}$, noise levels $n_b = b/2\%$. So for type b, with probability $1 - n_b$, the ball moves in the current direction, and with probability n_b, it moves in another random direction. Hence, the type modifies the base transition probabilities (1) by the noise n_b and (2) because the ball sizes determine the locations that result in collisions. We ran two sets of experiments, one with a simulated user and another with human users. In both cases, we ran an initial training phase to collect transition distribution information for each $(b, n_b) \times z$ pair. We used $k = 1$, and $\epsilon_0 = 0.5$, ϵ decreased by 0.1 every episode.

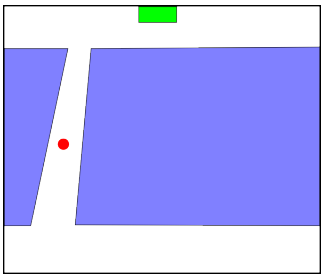

Figure 1: Our video game domain; green rectangle is the goal location, and the red circle is the ball.

Algorithms. The simulated user policy was the A^* algorithm, where the direction chosen at each step, given speed z, was toward the location, at distance z from the current location, that had the lowest cost. After training, we ran a test phase, during which we chose the type at random and recorded the loss over 5 episodes of BEAT, each action-set and EXP-3 averaged over 100 runs. Note that, the user knows her type but *these methods do not*. For the simulated user, the policy used was the same, and for the human experiment, we used three different users, each performing a sequence of many trials involving combinations of ball sizes and speeds.

4.2 Results

Simulated User. We present three types of plots. The first plot, Figure 2 shows the loss of each z for three representative (b, n_b) and illustrates a 'phase-transition' phenomenon on the loss of the speeds for the different sizes: higher speeds tend to work better for smaller sizes and conversely. Figure 3 shows the losses in terms of the *population diversity*. Here, in the x-axis each x corresponds to all possible combinations of x types (so all $\binom{6}{x}$ combinations). This captures different levels of heterogeneity in user populations. In the figure, for the curve α_*, the point $\alpha_*(x)$ gives the loss of best speed averaged across all populations of size x. Hence $\mathrm{EXP}-3(x)$ gives the loss of EXP-3 averaged over all population of size x, while $BEAT(x)$ gives the loss of BEAT averaged over all population of size x. This curve shows, that averaged across population sizes, BEAT outperforms EXP-3 significantly. Furthermore, while for the smaller population size, the α_* beats $BEAT$, for the larger population sizes BEAT beats α_* comprehensively, illustrating the need for adaptation. Additionally, note that the curve for α^*, the optimal per type, averaged over all types, is the line $f(x) = \alpha_*(1)$ because $\alpha_*(1)$ gives the 'population optimal' of size 1, which is just α^*. Finally, Figure 4 shows how well BEAT is able to identify types. It shows that BEAT has trouble identifying the smallest type. We conjecture that this is because the behavior of $b = 5$ and $b = 2$ are nearly indistinguishable. However, other than that, it is able to identify the correct type fairly rapidly, almost always halfway through the first episode.

Human User. The results for the human experiments are presented as before in three plots. Figure 5 shows the phase-transition in the loss at different speeds. Figure 6 shows the performance of BEAT and α_* for the human experiments in terms of population diversity. We do not report the performance of EXP-3 because data collection for that algorithm takes time that exceeded the constraints of our human subjects. In these experiments, we see that BEAT significantly outperforms α_* for all population sizes, hence demonstrating the benefit of adaptation with real human subjects, corroborating our more extensive simulation results above. This is a key experimental result of this paper. Finally, Figure 7 shows the efficacy of our algorithm at identifying different types. As in simulated user experiments, our algorithm has some trouble distinguishing between the very nearby types $b = 5$ and $b = 2$ but the error is not beyond the threshold of small noise.

5 Conclusions

We present a new model and an algorithm for adapting interaction environments. We are motivated by applications wherein interfaces must be quickly tuned to users who may be drawn from a heterogeneous population. Our latent variable model captures behavioural types, where each type defines an instance of a complete decision process. Our algorithm is Bayesian in nature, and enjoys the desirable property that our initial action set choice is already a *safe* one, corresponding to a notion of population-optimal setting. As our theoretical analysis and experiments show, BEAT learns *quickly*, suggesting promise that it address the ways in which untrained users might actually interact with interfaces. A key conclusion of our experiments is that this kind of adaptation is indeed necessary in environments with substantial diversity – the regret of alternate approaches shoots up at high diversity. Our experiments also raise questions for future work. Currently, we work under the assumption that when an action set has changed, the user accordingly changes her behaviour. In our algorithm, selecting small values of k ensures that this is properly handled. Devising more sophisticated ways to accommodate user learning is an avenue for further exploration. Similarly, we work in a setting where a training corpus corresponding to various types seeds our learning algorithm. Incrementally acquiring data corresponding to heterogeneous types and performing (fully unsupervised) learning in a life-long fashion is another important direction to be explored.

SIMULATED USERS

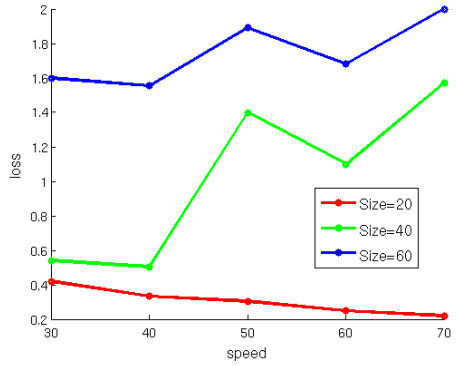

Figure 2: The performance of each speed for each type, illustrating baseline domain performance. The user is simulated.

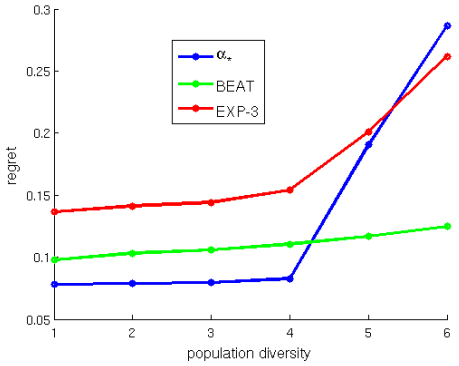

Figure 3: The performance of best static speed α_*, EXP-3 and BEAT for each of 6 possible combinations of type populations when the user is simulated. Please see Section 4.2 for details.

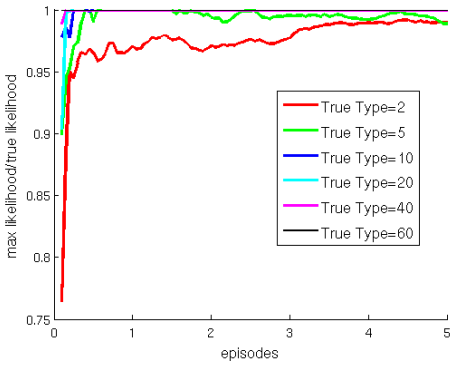

Figure 4: The ratio of the maximum-likelihood type and posterior and the true type in BEAT, averaged over all population-diversity trials in our experiments. The user is simulated.

HUMAN USERS

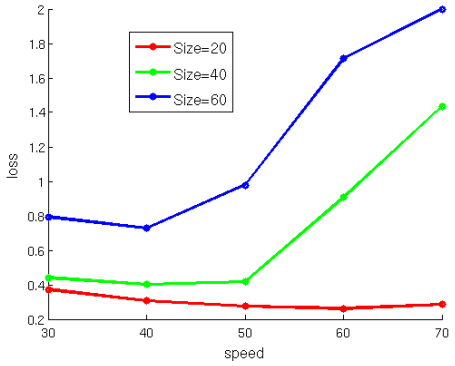

Figure 5: The performance of each speed for each type. This establishes baseline domain performance when the user is human.

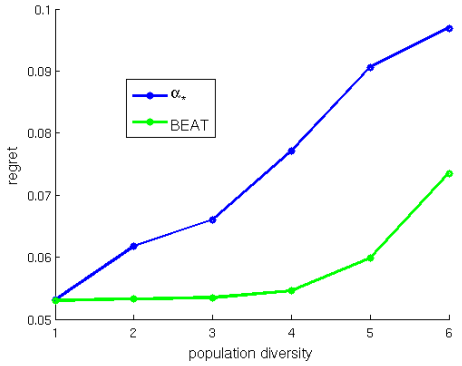

Figure 6: The performance of best static speed α_* and BEAT for each of 6 possible combinations of population types when the agent is human. Please see Section 4.2 for details.

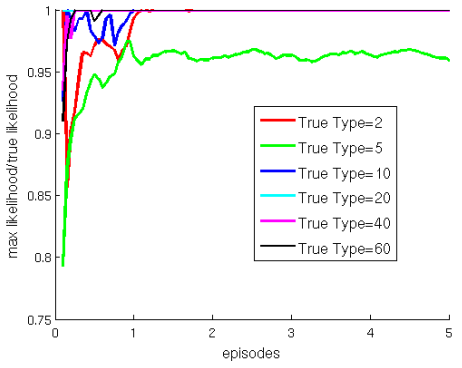

Figure 7: The ratio of the maximum-likelihood type and posterior and the true type in BEAT, averaged over all population-diversity trials in our experiments when the user is human.

A Proofs

For the proofs, we introduce the following distribution over state-action-set-action sequence $s\alpha a_{0:t} \triangleq s_0\alpha_0 a_0 \cdots s_{t-1}\alpha_{t-1}a_{t-1}s_t$:

$$Pr_2(s\alpha a_{0:t};\epsilon,\tau^*) \triangleq \prod_{i=0}^{t-1} T(s_{i+1}|s_i,a_i;\tau^*)Pr(\alpha_i|sa_{0:i})I_{[a_i=\pi^*_{\alpha_i}(s_i)]} \tag{6}$$

So clearly, marginalizing Pr_2 over action-set sequences gives us $Pr(sa_{0:t}|\epsilon,\tau^*)$ defined in (4).

We now prove Theorem 1. The main ideas can be broken down as follows (each appearing as a Claim in the proof). (1) An application of the Borel-Cantelli lemma gives us that BEAT chooses action set $\bar{\alpha}$ infinitely often almost surely. (2) The $\ln$ of the posterior ratio (5) diverges in expectation. This is a non-trivial step that requires a careful decomposition of the expectation into a sum of KL-divergences and then an application of the Gibbs inequality. (3) Finally, using monotonicity of $\ln$ and the fact that convergence in mean *sum* implies almost sure convergence, we show that the ratio (5) diverges almost surely.

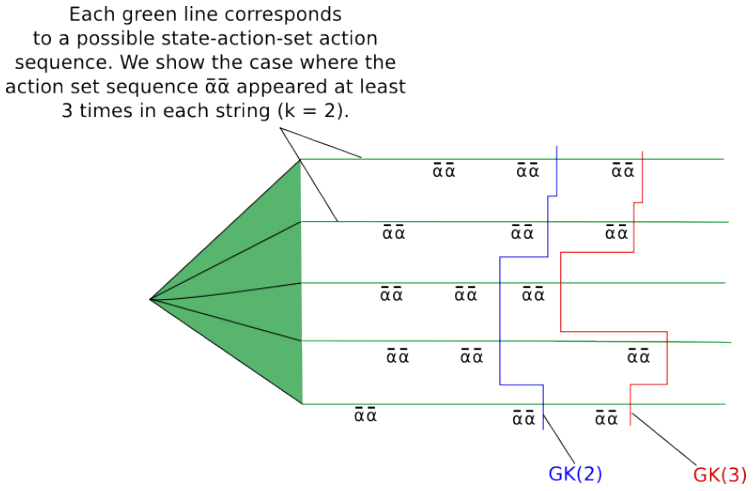

Figure 8: This figure illustrates the notion that the action set sequence $\bar{\alpha}\bar{\alpha}$ appears 3 times for all strings. The sets $GK(m)$ are defined in (7).

Proof of Theorem 1. The proof consists of three claims. The first claim is illustrated in figure 8.

Claim 1. *BEAT selects action set $\bar{\alpha}$ infinitely often almost surely. In other words, for any n, there is a smallest t_n such that in* every *sequence of action sets $\alpha_{0:t_n}$ of positive probability under Pr_2, $\bar{\alpha}$ appears in n, non-overlapping subsequences, each of length k.*

Proof. Recall that, for a sequence of events A_n, $A^* \triangleq \limsup A_n = \cap_{l=0}^{\infty} \cup_{n=l}^{\infty} A_n$. This means that if $x \in A^*$, then there are infinitely many A_n such that $x \in A_n$. Defining $A_n = \{\alpha_{nk} = \bar{\alpha}$ at step nk of BEAT due to exploration via ϵ_n parameter in line 8 $\}$, A^* corresponds to the event that $\bar{\alpha}$ is chosen infinitely often:

$$A^* = \{\alpha_{0:\infty} | \forall n, \exists n' > n, \alpha_{n'} = \bar{\alpha}\}$$

Now A_n are all independent, and by the condition on ϵ_n, $\sum_n Pr(A_n) = \sum_n \epsilon_n |\mathsf{AS}|^{-1} = \infty$. Hence, by the Borel-Cantelli lemma (see for instance Theorem 2.7 [17]), $Pr(A^*) = 1$, and this implies that $\bar{\alpha}$ is chosen infinitely often almost surely. This gives the first part of the claim. The second part now follows from the fact that in BEAT the action-sets are changed only every k steps. □

Claim 2. $\lim_{t\to\infty} \ln \frac{L(\tau^*|sa_{0:t})W(\tau^*)}{L(\tau|sa_{0:t}W(\tau)} = \infty$ *in* $Pr(\cdot|\epsilon,\tau^*)$ *expectation.*

Proof. We can write

$$\ln \frac{L(\tau^*|sa_{0:t})W(\tau^*)}{L(\tau|sa_{0:t})W(\tau)} = \ln \frac{\prod_{i=0}^{t-1} T(s_{i+1}|s_i,a_i;\tau^*)W(\tau^*)}{\prod_{i=0}^{t-1} T(s_{i+1}|s_i,a_i;\tau)W(\tau)} = K + \sum_{i=0}^{t-1} \ln \frac{T(s_{i+1}|s_i,a_i;\tau^*)}{T(s_{i+1}|s_i,a_i;\tau)}$$

The first equality expands using the definition of $L(\cdot)$ and the second equality uses the defining property of $\ln$, where we set $K \triangleq \ln[W(\tau^*)/W(\tau)]$, and K is finite. Taking expectation of the final term above, sans K, with respect to $Pr(sa_{0:t};\epsilon,\tau^*)$ we can rewrite as follows:

$$\begin{aligned}
&\Rightarrow \sum_{sa_{0:t}} Pr(sa_{0:t};\epsilon,\tau^*) \sum_{i=0}^{t-1} \ln \frac{T(s_{i+1}|s_i,a_i;\tau^*)}{T(s_{i+1}|s_i,a_i;\tau)} \\
&\overset{(1)}{=} \sum_{i=0}^{t-1} \sum_{sa_{0:t}} Pr(sa_{0:t};\epsilon,\tau^*) \ln \frac{T(s_{i+1}|s_i,a_i;\tau^*)}{T(s_{i+1}|s_i,a_i;\tau)} \\
&\overset{(2)}{=} \sum_{i=0}^{t-1} \sum_{sa_{0:i+1}} Pr(sa_{0:i+1};\epsilon,\tau^*) \ln \frac{T(s_{i+1}|s_i,a_i;\tau^*)}{T(s_{i+1}|s_i,a_i;\tau)} \\
&\overset{(3)}{=} \sum_{i=0}^{t-1} \sum_{sa_{0:i}} Pr(sa_{0:i};\epsilon,\tau^*) \sum_{\alpha_i} Pr(\alpha_i|sa_{0:i}) \sum_{a_i} I_{[a_i=\pi^*_{\alpha_i}(s_i)]} \\
&\qquad \left[\sum_{s_{i+1}} T(s_{i+1}|s_i,a_i;\tau^*) \ln \frac{T(s_{i+1}|s_i,a_i;\tau^*)}{T(s_{i+1}|s_i,a_i;\tau)} \right] \\
&\overset{(4)}{=} \sum_{i=0}^{t-1} \sum_{sa_{0:i}} Pr(sa_{0:i};\epsilon,\tau^*) \sum_{\alpha_i} Pr(\alpha_i|sa_{0:i}) \sum_{a_i} I_{[a_i=\pi^*_{\alpha_i}(s_i)]} \\
&\qquad D[T(\cdot|s_i,a_i;\tau^*)||T(\cdot|s_i,a_i;\tau)] \\
&\overset{(5)}{=} \sum_{i=0}^{t-1} \sum_{s\alpha a_{0:i}} Pr_2(s\alpha a_{0:i};\epsilon,\tau^*) \sum_{\alpha_i} Pr(\alpha_i|sa_{0:i}) \sum_{a_i} I_{[a_i=\pi^*_{\alpha_i}(s_i)]} \\
&\qquad D[T(\cdot|s_i,a_i;\tau^*)||T(\cdot|s_i,a_i;\tau)]
\end{aligned}$$

In the above, in $^{(1)}$ we moved the probability/multiplier inside and then swapped the sums. In $^{(2)}$ we used the fact that the $\ln$ term is independent of $a_{i+1}sa_{i+2:t}$ (alternatively, the fact that for any sequence of events $\{B_i\}_{(i)}$, $\sum_i Pr(B_i,C)f(C) = \sum_i Pr(B_i|C)Pr(C)f(C) = Pr(C)f(C)$). In $^{(3)}$ we used the definition (4) of $Pr(sa_{0:i};\epsilon,\tau^*)$. In $^{(4)}$, $D[\cdot||\cdot]$ is the KL divergence between the two distributions. Finally in $^{(5)}$ we introduced the action-sets explicitly by using (6). Below we will work with this final form $^{(5)}$ of the expectation, w.r.t. $Pr(sa_{0:t};\epsilon,\tau^*)$, of the log-likelihood ratio. In the following, our goal will be to write this as a sum over state-action-set-action strings that end in subsequences containing $\bar{\alpha}$, has probability at least β and has $D(\cdot||\cdot) = \eta > 0$.

By Claim 1, for any any n, there is a smallest t_n such that for *every* sequence of action sets $\alpha_{0:t_n}$ of positive probability under Pr_2, $\bar{\alpha}$ appears in n, non-overlapping subsequences each of length k. We use this fact to define the set $GK(m)$ with $m \leq n$, which contains all state-action-set-action sequences where non-overlapping k-length $\bar{\alpha}$ sequences have appeared exactly m times, and nothing else has appeared after that (See figure 8 for an illustration).

$$\begin{aligned}
GK(m) \triangleq \{ s\alpha a_{0:t} | t \leq t_n, Pr_2(s\alpha a_{0:t};\epsilon,\tau^*) > 0, \text{ there are } m \text{ non-overlapping subsequences} \\
\alpha_{j:j+k-1} \text{ with } \alpha_i = \bar{\alpha} \text{ for } j \leq i \leq j+k-1, \text{ with } \alpha_{t-k:t-1} \text{ being one such sequence} \}
\end{aligned} \quad (7)$$

First note that $GK(m)$ is a collection of strings that may be of possibly different lengths. Furthermore, because each element of $GK(m)$ has maximal number of occurrences of m sequences, this set is prefix free (i.e. no string is a prefix of another). The maxmimality also implies that $GK(m)$ is disjoint from $GK(m')$ whenever $m' \neq m$. By Claim 1, $\sum_{s\alpha a_{0:t} \in GK(m)} Pr_2(s\alpha a_{0:t};\epsilon,\tau^*)) = 1$ for each m (because for some $t \leq t_n$ *every* string of positive probability has exactly m occurrences of k-length $\bar{\alpha}$ sequences – see figure 8).

The hypothesis of this theorem postulates that there must be a subset of B of $GK(m)$ such that $\sum_{s\alpha a_{0:j}\in B} Pr_2(s\alpha a_{0:j};\epsilon,\tau^*)) \geq \beta$ and for each $s\alpha a_{0:j} \in B$, there must be at least one action a_i and state s_i, $j-k-2 \leq i \leq j-1$, such that $T(\cdot|s_i,a_i;\tau^*) \neq T(\cdot|s_i,a_i;\tau)$. Then, by the Gibb's inequality for KL divergence, $D[T(\cdot|s_i,a_i;\tau^*)||T(\cdot|s_i,a_i;\tau] > \eta > 0$ ($\eta > 0$ because all the spaces are finite, and infinitely decreasing bounds are not possible). We will call such a state-action pair a *witness* pair (i.e. they are a witness that $\tau^* \neq \tau$).

Using these definitions, $^{(5)}$ from above, with $t = t_n$, is

$$
\begin{aligned}
&\overset{(6)}{\geq} \sum_{m=1}^{n} \sum_{s\alpha a_{0:j}\in GK(m)} \sum_{i=j-k}^{j-1} Pr_2(s\alpha a_{0:i};\epsilon,\tau^*) \sum_{\alpha_i} Pr_2(\alpha_i|s\alpha a_{0:i}) \\
&\quad \sum_{a_i} I_{[a_i=\pi^*_{\alpha_i}(s_i)]} D[T(\cdot|s_i,a_i;\tau^*)||T(\cdot|s_i,a_i;\tau)] \\
&\overset{(7)}{\geq} \sum_{m=1}^{n} \sum_{s\alpha a_{0:j}\in GK(m)} \sum_{i=j-k}^{j-1} Pr_2(s\alpha a_{0:i};\epsilon,\tau^*) Pr_2(\bar{\alpha}|s\alpha a_{0:i}) \\
&\quad \sum_{a_i} I_{[a_i=\pi^*_{\bar{\alpha}}(s_i)]} D[T(\cdot|s_i,a_i;\tau^*)||T(\cdot|s_i,a_i;\tau)] \\
&\overset{(8)}{>} \sum_{m=1}^{n} \sum_{s\alpha a_{0:j}\in GK(m)} \sum_{i=j-k}^{j-1} Pr_2(s\alpha a_{0:i};\epsilon,\tau^*) Pr_2(\bar{\alpha}|s\alpha a_{0:i}) \\
&\quad \sum_{a_i} I_{[a_i=\pi^*_{\bar{\alpha}}(s_i)]} \eta I_{[s_i,a_i \text{ is a witness pair}]} \\
&\overset{(9)}{=} \sum_{m=1}^{n} \beta\eta \overset{10}{=} n\beta\eta
\end{aligned}
$$

In the above, in $^{(6)}$ the outer sums in $^{(5)}$ are replaced by 3 sums. To show that this is valid, we need to show that (a) all the strings are of length $\leq t = t_n$ and (b) no string appears more than once in the sums in $^{(6)}$. For (a), by definition $GK(m)$ only contain strings of length $\leq t = t_n$. For (2), all the $GK(m)$ are disjoint. Furthermore, if $s\alpha a_{0:j} \in GK(m)$, then only $s\alpha a_{0:j-l}$ for $l \geq k$ appears in $GK(m-1)$ and so the inner sum for $GK(m-1)$ reaches only up to $s\alpha a_{0:j-l-1}$. Putting all this together, we have that no string appears more than once.

$^{(7)}$ follows because we exclude all action sets $\neq \bar{\alpha}$. $^{(8)}$ follows because we only keep the witness actions and for these the KL divergence is lower bounded by η. $^{(9)}$ follows because $\sum_{sa_{0:t}\in GK(m)} Pr_2(sa_{0:t}) = 1$ for each m and the cumulative probability of the witness strings for each m is $\geq \beta$, and $^{(10)}$ is obvious.

Finally, since by Claim 1 $n \to \infty$ as $t \to \infty$, and since $K = \ln[W(\tau^*)/W(\tau)]$ is finite, we have that $\ln$ posterior ratio in the statement of the claim also diverges in expectation.

□

Claim 3. *Let* $g_t \triangleq \frac{L(\tau^*|sa_{0:t})W(\tau^*)}{L(\tau|sa_{0:t}W(\tau)}$. *Then the* $\lim_{t\to\infty} I_{[g_t>M]} = \infty$ *for each* M *almost surely.*

Proof. In Claim 3 we established that $\mathbb{E}(g_t)$ goes to ∞ in expectation. In particular, for each M, there exists a t, such that $\mathbb{E}(g_t) > M$ – indeed, from the proof of Claim 1, $t = t_{n_M}$, where $n_M = (M+1)(\beta\eta)^{-1}$. For this t, $\mathbb{E}[I_{[g_t>M]}] = 1$. Since $I_{[g_t>M]} \leq 1$ for all t, this implies that $\sum_{t=1}^{\infty} \mathbb{E}[|1-I_{[g_t>M]}|] = \sum_{t=1}^{\infty} \mathbb{E}[1-I_{[g_t>M]}] = \sum_{t=1}^{\infty} 1-\mathbb{E}[I_{[g_t>M]}] \leq t_{n_M}$. Furthermore since $1 - I_{[g_t>M]} \leq 1$, $(1-I_{[g_t>M]})^2 \leq (1-I_{[g_t>M]})$. Putting all these facts together, we have

$$\sum_{t=1}^{\infty} \mathbb{E}[(1-I_{[g_t>M]})^2] < \infty \tag{8}$$

Now recall that a random variable X_n is said to converge *in mean sum* to a random variable X if $\sum_{t=1}^{\infty} \mathbb{E}[(X-X_n)^2] < \infty$. Additionally, of X_n converges to X in mean sum, then it also converges

almost surely. This, togeteher with (8) implies that $I_{[g_t > M]}$ converges to 1 almost surely. Since M was arbitrary, this completes the proof. □

By Claim 3, the posterior ratio is greater than every M almost surely, which means that the posterior ratio diverges almost surely, completing the proof.

□

Proof of Corollary 1. This result follows directly from the derivation of $^{(9)}$ in the proof of Theorem 1. In particular, under the hypothesis of this corollary all the statements leading upto (9) remain true, except that the probability mass of the set of strings in the sum in $^{(5)}$ is $PE_\tau(t, m)$, from which the statement of the Corollary follows. □

Proof of Theorem 2. By Theorem 1 the the relative posterior of all $\tau \neq \tau^*$ vanishes. Hence for each α, $\sum_i Pr(\tau_i|sa_{0:t})V^\alpha_{\tau_i} = V^\alpha_{\tau^*}$ almost surely. From whence, by defintion (3) of α_{BO} and (2) of α^*, $\alpha_{BO}(sa_{0:t}) = \alpha^*$ almost surely. Finally, as $\lim_{t\to\infty} \epsilon_t = 0$, in lines 8-12 of the BEAT algorithm only the optimal action set is chosen in probability. □

Proof of Theorem 3. By Corollary 1 For $t = t_{\delta,M}$, for each τ with probability at least $1 - \delta/\hat{l}$, $\frac{L(\tau^*|sa_{0:t})W(\tau^*)}{L(\tau|sa_{0:t})W(\tau)} > \frac{R_{\max}\hat{l}}{(1-\gamma)V^{\alpha^*}_{\tau^*}}$, rewriting which gives us

$$L(\tau^*|sa_{0:t})W(\tau^*)V^{\alpha^*}_{\tau^*}\frac{1}{(|\mathsf{Sk}|-1)} > L(\tau|sa_{0:t})W(\tau)\frac{R_{\max}}{(1-\gamma)} \tag{9}$$

Now $R_{\max}/(1-\gamma)$ is a bound on V^α_τ for any V^α_τ, and summing (9) over τ gives us that with probability at least $1 - \delta$,

$$L(\tau^*|sa_{0:t})W(\tau^*)V^{\alpha^*}_{\tau^*} > \sum_{\tau\neq\tau^*} L(\tau|sa_{0:t})W(\tau)V^\alpha_\tau$$

for any α, τ. The above implies that $\alpha_{BO}(sa_{0:t_{\delta,M}}) = \alpha^*$, proving the theorem. □

A.1 Discussion of the Hypothesis in Theorem 1

In Theorem 1, we make the hypothesis that the type τ satisfies that there exists at least one action-set $\bar{\alpha}$ for which $T(\cdot|s, \pi^*_{\bar{\alpha}}(s); \tau) \neq T(\cdot|s, \pi^*_{\bar{\alpha}}(s); \tau^*)$ for at least one state for any length k path of the optimal policy $\pi^*_{\bar{\alpha}}$ with probability at least $\beta > 0$. Intuitively it can be seen that if this very weak condition is not satisfied, then the two types are equivalent. We now give a very brief sketch of the formal argument as to why this implies that the two types are equivalent. We make the standard assumption that any policy in the MDP is ergodic [14]. This, coupled with Claim 1 in Theorem 1, implies that on any path generated by a sequential application of optimal policies $\pi^*_{\alpha_t}$ in BEAT, we will observe every sequence of k length paths possible under every action set α infinitely often almost surely. Hence, if the lower bound on the probability β is 0, then it must mean that $T(\cdot|s, \pi(s); \tau) = T(\cdot|s, \pi(s); \tau^*)$.

References

[1] A. Valtazanos and S. Ramamoorthy. Evaluating the effects of limited perception on interactive decisions in mixed robotic environments. In *HRI '13: Proc. ACM/IEEE International Conference on Human-Robot Interaction*, 2013.

[2] Alan Fern and Prasad Tadepalli. A computational decision theory for interactive assistants, advances in neural information processing systems. In *Proceedings of the* 23^{rd} *Conference on Neural Information Processing Systems*, 2010.

[3] Eugene Charniak and Robert Goldman. A probabilistic model of plan recognition. In *Proceedings of the ninth National conference on Artificial intelligence - Volume 1*, AAAI'91, pages 160–165. AAAI Press, 1991.

[4] Sven Seuken, David C. Parkes, Eric Horvitz, Kamal Jain, Mary Czerwinski, and Desney S. Tan. Market user interface design. In *ACM Conference on Electronic Commerce*, pages 898–915, 2012.

[5] Haoqi Zhang, Yiling Chen, and David C. Parkes. A general approach to environment design with one agent. In *IJCAI*, pages 2002–2014, 2009.

[6] Diane Litman Satinder Singh, Michael Kearns, and Marilyn Walker. Optimizing dialogue management with reinforcement learning: Experiments with the NJFun system. *Journal of Artificial Intelligence Research*, 16:105–133, 2002.

[7] Sylvie C. W. Ong, Shao Wei Png, David Hsu, and Wee Sun Lee. Planning under uncertainty for robotic tasks with mixed observability. *I. J. Robotic Res.*, 29(8):1053–1068, 2010.

[8] T. Bandyopadhyay, K. S. Won, D. Hsu E. Frazzoli, W. S. Lee, and D. Rus. Intention-aware motion planning. In *Proceedings Workshop of Algorithmic Foundations, WAFR-2012*, 2012.

[9] E. A. Sisbot, L. F. Marin-Urias, R. Alami, and T. Simeon. A human aware mobile robot motion planner. *IEEE Transactions on Robotics*, 23(5):874–883, 2007.

[10] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. *Recommender Systems An Introduction*. Cambridge University Press, 2011.

[11] David Maxwell Chickering and Tim Paek. Personalizing influence diagrams: applying online learning strategies to dialogue management. *User Modeling and User-Adapted Interaction*, 17(1-2):71–91, 2007.

[12] Guy Shani, David Heckerman, and Ronen I. Brafman. An mdp-based recommender system. *Jouornal of Machine Learning Research*, 6:1265–1295, 2005.

[13] K.Z. Gajos, J.O. Wobbrock, and D.S. Weld. Improving the performance of motor-impaired users with automatically-generated, ability-based interfaces. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1257–1266, 2008.

[14] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons, 1994.

[15] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32:48–77, 2002.

[16] Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction Learning and Games*. Cambridge University Press, 2006.

[17] Achim Klenke. *Probability Theory: A Comprehensive Course*. Springer-Verlag, 2008.