AI in Wireless Networks: The Personal Router Agent for Wireless Access

P. Faratin and **G. Lee** and **J. Wroclawski**
Laboratory for Computer Science
M.I.T
Cambridge, 02139, USA
{peyman,gil,jtw}@mit.edu

S. Parsons
Department of Computer and Information Science
Brooklyn College, City University of New York
NY,11210, USA
parsons@sci.brooklyn.cuny.edu

Abstract

Elicitation of user preferences has been recognized to be one the most important goals of user-centered AI systems. Solutions to this problem have been cast as a utility function construction problem to adaptive classi£cation given classes of utility functions, to sequential decision making. In this paper we present the preference elicitation problems involved in dynamic user access to wireless networks. We propose an interactive, adaptive agent-based solution to the problem and show how the full nature of the problem can be represented within a Markov Decision Process (MDP). Adaptive reinforcement learning solutions are then evaluated for two subclasses of tractable MDPs via simulations of some representative user models.

1 Introduction

AI components are increasingly being embedded within complex user-centered systems as autonomous problem solving modules. In this paper we present an application of AI within the domain of future wireless networks. In particular, in contrast to the vertically integrated structure of the current Internet, we seek enabling technologies and policies for future wireless networks where more complex network services can be represented and traded amongst multiple buyers and sellers in a dynamic market.

In this paper we will illustrate the potential of AI solutions for this overall goal by presenting an adaptive *personal* agent system, called the Personal Router (PR), that solves one dimension of nomadic user access problem to wireless networks (see [Faratin *et.al.*, 2002] for a full problem description). The problem dimension of interest is referred to as the Autonomous User Modeling (AUM) problem. Informally, the AUM problem is stated as the problem of autonomously modeling a nomadic user's preference for different wireless services, where services, like goods in a market, are offered by multiple sellers and which may have multiple dimensions. Once derived this information model can subsequently be used as input for dynamic or single shot optimization, depending on the type of market protocol the agent is currently trading in.

We frame the AUM problem as an adaptive (feedback) control problem where a personal agent (a controller) takes adaptive actions (control signals) in an environment (a control system) to achieve its goals. Furthermore, as opposed to the classic approaches where a *single* optimal decision is made *after* eliciting the *complete* preference structure of the user [Keeney and Raiffa, 1976], the goal of the agent is to £nd an optimal *strategy* over sequences of decisions over time, where the sequence is composed of elicitation followed by decision making processes. In such a user-agent coupling, initially *user perceived* sub-optimal agent decisions can be improved over time with additional information acquired by elicitation. A sequential decision making approach is adopted because the dynamic decision environment creates a barrier for obtaining a priori information about the user and the user cannot be engaged in a costly elicitation process similar to traditional solutions. Minimal user interaction during preference elicitation is also sought by Chajewska *et.al.* where the UM problem is viewed as a classi£cation problem [Chajewska *et.al.*, 2000]. A myopically optimal elicitation strategy is constructed to ask the single query with the greatest expected value of information with respect to a distribution of clusters of utility functions. The uncertainty over this distribution is then re£ned as users answer queries. However, the hillclimbing nature of myopic strategy can fail to ask appropriate questions because *future* values are neglected when determining the value of current questions. Boutilier's extension of the above model to a Partially Observable Markov Process (POMDP) [Boutilier, 2002], implements a sequential decision problems with multistage lookahead. However, although also modeling the elicitation process as a sequential decision making problem this model assumes a priori belief model (although any arbitary model) for optimization.

Generally, we are interested in cost and bene£t trade-offs involved between perfect information for a single optimal decision mechanism against imperfect information and suboptimal but iterative and adaptive mechanisms. In this paper we propose a state-based model that approaches the latter mechanism.

The paper is organized as follows. A general description of the PR problem is brie¤y described in the £rst section. We then present a formal model of the service selection problem. Next we show how the *full* problem description can be computationally represented within a Markov Decision Process.

This is followed by two simpler MDP models of the PR problem that are motivated by the intractability of the fully dimensioned model. Next we review how reinforcement learning algorithms can be used to solve the agent's action selection problem. The behaviours of these algorithms are then shown in a set of user model simulations. Finally, we presents our conclusions together with the directions of future research.

2 The Personal Router

Optimal decision making requires access to a well de£ned preference structure of the decision maker that gives a meaningful ordering to the set of possible outcomes. Traditionally, a solution to such user modeling problems are formulated within the classical utility analysis framework, such as conjoint analysis [Keeney and Raiffa, 1976]. However, although useful for most static and tractable domains, decision analysis techniques are inappropriate in wireless networks due to the complexity in and dynamicity of the user's context [Faratin *et.al.*, 2002]. The user context is de£ned by: a) the user's goals (or activities—e.g. arranging a meeting, downloading music), b) the class of application the user is currently running in order to achieve her goals (e.g. reading and sending emails, £le transfer), c) her urgency in using the service and d) her location (e.g nomadic or stationary). This context is highly complex not only because a user may have multiple concurrent goals/activities but also because different elements of the user context (goals, locations, running applications, etc.) may change at different rates. Indeed, we claim that such complexities necessitate a personal agent-based approach because on-line agents can learn user's preferences over time rather than requiring users to participate in an unrealistic and expensive elicitation process. For example, the set of service choices (or outcomes) can change dynamically when current network connection(s) instantly become unreachable as mobile users change locations. Alternatively, preferences themselves can also be dynamic where different services may be needed as users dynamically change and begin different tasks. In addition to the complexity of the user context there exist additional barriers for traditional preference modeling approaches. For example, due to cognitive costs and nomadic nature of wireless access users may be reluctant to engage in costly communications over their preferences, especially if the elicitation space is combinatorially large. In the worst case users may be ignorant of their preferences, a real possibility with network services which the user, unlike common goods such as bread or milk, has little or no experience of in order to form a preference over. Indeed, intangibility of network services may necessitate a trial period before the user can begin to form preferences. Furthermore, there exists an inherent variability in the network itself resulting in uncertainties by both the buyers and the sellers of a service as to the guarantees that can be made over the quality of a service (QoS). To overcome this uncertainty users are therefore given a user interface to manipulate service features as free variables via *better* and *cheaper* buttons on the PR respectively. The assumption we make is that the user will choose better or cheaper services if the current selected service is either of poor quality or high price respectively. This process of interaction with the PR may continue until the PR learns to select a service that satis£es the user's current tasks and goals. Note, that because of relatively low wireless service prices we assume users are tolerant to suboptimal decisions in service selection because the (monetary) cost of decision errors is low.

3 Problem Representation as a MDP

A representation of the PR that *fully* modeled the complexity and dynamic nature of the problem within the Markov Decision Process (MDP) modeling framework was presented in [Faratin *et.al.*, 2002]. Brie¤y, an MDP is a directed acyclic graph composed of a set of nodes and links that represent the system states $\mathbf{S}$ and the probabilistic transitions $\mathbf{L}$ amongst them respectively [Kaelbling *et.al*, 1996]. Each system state $S \in \mathbf{S}$ represents the information available to the decision maker and is speci£ed by a set of variables that describe the states of the problem to some degree of speci£city. In our problem system state $S \in \mathbf{S}$ is constructed by the conjunction of: a) the user context ($c^g = \langle \beta^g, \gamma^g, \delta \rangle$) current running applications, user deadlines and locations for goal g respectively, b) the set of pro£les available in the current location (P^δ) and c) the user interaction with the PR, which we will represent by the variable I. Therefore, a complete description of *a* system state at time t is represented by the conjunction $S^t = (\beta^g, \gamma^g, t, loc^g, P, I)$.

The other element of a MDP is the set of possible actions $\mathbf{A}$. Actions by either the user, the PR or both will then results in a state transition, that change the values of the state variables, to another state in the set of all possible states $\mathbf{S}$. In an MDP these transitions are represented by links $\mathbf{L}$ between states that represent the transition of a system state from one con£guration to another after performing some action. In our problem the set of actions A available to the user u are de£ned by the set $A^u = \{\Delta^{loc}, \Delta^{app}, \Delta^I, \phi\}$, representing changes in the user location, set of running applications, service quality and/or price demand and no action respectively. The consequences of user actions are changes in values of state variables.

In a negotiation mechanism the actions of the agent can be to increase/decrease in each, or product of, decision variable. For example, if the decision variables are price and bandwidth of a service then $A^{PR} = \{LBW, HBW, LP, HP, \phi\}$, where the elements correspond to a request for a service with a lower bandwidth, higher bandwidth, lower price, higher price and no action respectively. As mentioned above, because the state of the network is uncertain the consequences of the agent actions are indeterministic. To model this the transitions between states in the MDP are probabilistic. Therefore there exists a probability distribution $Pr_{a_j}(S_k|S_j)$ over each action a_j reaching a state k from state j.

We can also compute the utility of a service pro£le i in context c for goal g (or $u^{c^g}(P_i)$) as the utility of being in a unique state whose state variables $(\beta^g, \gamma^g, t, loc^g, P, I)$ have values that correspond to service i in context $c = \{\beta^g, \gamma^g, t, loc^g\}$. The utility of this corresponding state, say state m, is then referred to as $U(S_m)$.

Optimal reasoning with an MDP (or user modeling in our

problem) in turn is de£ned as £nding a policy π or a function that maps the from each state to an optimal action. If the MDP model (transition probabilities and state utilities) is given then exact methods such as dynamic programing can be used to solve the MDP by computing the expected values of each action at each state recursively from the £nal state. However, exact methods are inappropriate for our problem because a) the transition probabilities are not given a priori and b) the horizon of the state space can be in£nite because the personal agent is assumed to be on-line and continuously making decisions. Below we propose adaptive mechanisms for updating model estimates over a reduced state space.

In general MDPs suffer from the "curse of dimensionality", where the state-space grows exponentially in the size of the variables. As a result it is impractical to consider learning the highly expressive model given above. The strategy we adopt to overcome this problem is to incrementally search for a computationally tractable MDP model that *increasingly* approaches some acceptable level of expressiveness, derived through ecological experiments. Different expressive-computational trade-off regimes can then be constructed that range from single state MDPs, with coarse state signals, to richer state signal MDP, described above, each with relatively different expressive and predictive power. The natural expectation, veri£able through ecological experiments, is that richer models result in more adequate behaviours because they can form better associations of states to actions (or policies) than simpler non-associative models. In the remainder of the paper we describe the £rst step in this strategy where the complex state-space of the above MDP described is "collapsed", through disjunction of all of the states, to either a single state signal or a slightly more complex state signal consisting of service pro£les. We show how the former reduced problem is equivalent to the k armed bandit problem and review heuristic solution methods for both classes of problems based on reinforcement learning. Finally, we evaluate the adequacy of the model through simulations.

4 Single State PR—a Bandit Problem

The computationally simplest model of the PR problem is to cast the agent action selection problem as a k armed bandit problem. Bandit problems have been extensively studied in statistical and mathematical literatures with application to medical diagnosis and sequential decision making in general. In this class of problems an agent has a choice of pulling one arm of a k-armed bandit machine at each time step. When the arm i of machine is pulled the machine pays off 1 or 0 according to some underlying probability p_i, where payoffs are independent events and unknown to the agent. The game often has a £nite horizon where the agent is permitted to pull h pulls. The goal of the agent is to select a policy that maximize some function of the total expected payoffs R_t, typically given by $R_t = E(\sum_{t=0}^{h} r_t)$, where r_t is the payoff or reward at time t.

Similarly, we model the PR problem as a single state in£nite horizon MDP where at each discrete time the agent takes an action (pulls an arm i), receives a reward from the user for its action and returns to the same state. Generally, the size of k, or the number of the arms of the bandit machine, is determined by the rules of the market mechanism the agent is trading. Thus in a negotiation mechanism the set of agent actions, or agent's strategy space, at each state is given by $A^{PR} = \{LBW, HBW, LP, HP, \phi\}$. That is, under such a mechanism the agent is playing $k = 5$ one-armed bandit machines. Furthermore, since we are interested in *continual* learning of user preferences the goal of the agent is to maximize its total rewards over long-run, or in£nite horizon, of the game. The reward model in turn is constructed from the actions of the user *with the agent* through the interface and not the environment. Thus, we only consider a subset of user actions $A^u = \{\Delta^I, \phi\}$, where Δ^I is changes in price and bandwidth demands (or "cheaper" or "better" button presses respectively) by the user and ϕ is no action. We map these user actions to binary agent reward values using the simple binary rule of $r_t = +1$ if ϕ else $r_t = -1$, representing positive rewards for lack of user intervention. We model the user's preferences for different agent actions by a reward probability distribution with mean $Q^*(a_i)$ for each action a_i. Finally, if this probability distribution is constant over time then we say that the user's preferences is stationary. Conversely, if the distribution of the bandit changes over time then the user's preferences is said to be non-stationary. We represent the latter with two parameters: (θ, η), the frequency and magnitude of change in $Q^*(a_i)$ respectively.

There exist a number of solutions for solving the optimal policy for the bandit problems, including dynamic-programming, Gittins allocation indices and learning automata [Kaelbling *et.al*, 1996; Sutton and Barto, 2002]. However, although optimal and instructive these methods are known not to scale well to complex problems [Kaelbling *et.al*, 1996]. Because our goal is to incrementally increase the complexity of the problems, and also because the "forgiveness" of the user to the suboptimal decisions, we instead concentrate on heuristic action selection techniques that, although are not provably optimal, are nonetheless tractable and approximate optimal solutions. Furthermore, as mentioned above, optimal techniques compute optimal policies *given* a model (model of the process that generate $Q^*(a_i)$). In the absence of this information the agent must form estimates over and update the value of each action. Therefore, the problem of action selection requires both a learning phase followed by action selection phase.

One popular method for updating the values of $Q_{k+1}(a_i)$, estimates for $Q^*(a_i)$ for action i after k rewards is the exponential recency-weighted average:

$$Q_{k+1}(a_i) = Q_k(a_i) + \alpha_k(a_i)[r_{k+1}(a_i) - Q_k(a_i)] \qquad (1)$$

where $0 < \alpha_k(a_i) \leq 1$ is the step-size, or learning, parameter for action a_i after k selection. If $\alpha_k(a_i) = 1/k$ then the learning rate varies at each time step. Under this condition the update rule 1 implements a sample average method [Sutton and Barto, 2002].

The next step in solving the bandit problem is to select an action given an estimate of value of actions. We compare the behaviour of three action selection strategies given $Q_{k+1}(a_i)$: greedy, $\epsilon-$greedy and softmax. The £rst strategy exploits

the current agent knowledge by selecting that action with the highest current value estimate: $a_i^* = arg\ max_a Q_t(a_i)$. Conversely, as the horizon of interaction increases then it may be more bene£cial to explore the action space since higher valued longer term rewards may be biased against by lower valued shorter term rewards (expressed as non-linearity in the optimization objective function). Exploration, or probability of selecting action a at time t, $(P_t(a_i)$ may be at some constant rate, ϵ or given by a Gibbs, or Boltzmann, distribution:

$$P_t(a_i) = \frac{e^{Q_t(a)/T}}{\sum_{a' \in A}^{n} e^{Q_t(a')/T}} \quad (2)$$

where the temperature T cools at rate μ with time t according to the equation $T_t = T_0 (1 - \mu)^t$. Action selection strategies with constant and variable values of ϵ are referred to as ϵ-greedy and softmax strategies respectively.

Learning the best action can also be achieved not by maintaining estimates of action value but rather an overall reward level, called the reference reward, that can be used as a decision criteria. Techniques based on this method are known as Reinforcement Comparison (RC) methods, precursors to actor-critic methods [Sutton and Barto, 2002]. In RC a separate measure of action preference for each action at t play of the bandit, $\rho_t(a)$, is kept that are used to determine the action-selection probabilities according to softmax rule 2. The preferences are updated as follows. After each play of the bandit the preference for the action selected on that play, a_t, is incremented by the error signal between the reward r_t and the reference reward $\bar{r}_t$, by $\rho_{t+1}(a_t) = p_t(a_t) + \alpha[r_t - \bar{r}_t]$, where α is a positive step-size parameter. Unlike action-value updates, the reference reward is an incremental average of all recently recieved rewards independently of the action taken: $\bar{r}_{t+1} = \bar{r}_t + \alpha[r_t - \bar{r}]$, where $0 < \alpha \leq 1$ is again some learning rate.

Finally, in Pursuit methods (PM) both action-value and action-preferences are maintained where the action preferences "pursue" the action that is greedy according to the current action-value estimate [Sutton and Barto, 2002]. If $\pi_t(a)$ represents the probability of selecting action a at time t, determined through softmax, and $a_{t+1}^* = arg\ max_a Q_{t+1}(a)$ represents the greedy action at play $t + 1$ then the probability of selecting $a_{t+1} = a_{t+1}^*$ is incremented by a fraction β toward 1: $\pi_{t+1}(a_{t+1}^*) = \pi_t(a_{t+1}^*) + \alpha[1 - \pi_t(a_{t+1}^*)]$. Then the probabilities of selecting other actions are decremented towards zero: $\pi_{t+1}(a) = \pi_t(a) + \alpha[0 - \pi_t(a)]$ for all $a \neq a_{t+1}^*$.

5 Multi-State PR Problem

As the next step, we consider a multistate model of the PR problem. In the single state bandit problem described above, the expected reward received depends solely upon the current agent action. In our multistate model, the amount of reward is based upon both the agent action and the current service pro£le, allowing us to model the agent's attempts to learn the user's utility function over the space of service pro£les.

Formally, we model the PR as a deterministic MDP with states $s \subset P^\delta$, the set of currently available service pro£les. In order to model the trade-off between quality and cost, we de£ne a service pro£le as a vector of two features (b, h), where b represents bandwidth and h cost. For simplicity, we constrain quality and cost to the set of nonnegative integers.

The set of possible agent actions A^{PR} remains the same as before, but in the multistate model the current service pro£le may change after every agent action. This transition function is deterministic and assumes that the agent gets the service pro£le it requests if it is available. For instance, If the state s at time t is (b_t, h_t), the agent selects action LBW, and $(b_t - 1, h_t) \in P^\delta$, then $s_{t+1} = (b_t - 1, h_t)$. If the desired service pro£le is not in P^δ, then the state remains unchanged. By limiting agent actions to the £xed set A^{PR}, we reduce the complexity of the agent while still enabling the exploration of the full set of services available.

As in the single state model, at each time step the agent receives a reward $r \in \{1, -1\}$ depending on the user action A^u. The user action probability distribution $p(s_i)$ is based on the utility $U(s_i)$ of the current service pro£le s_i. We model user utility with the linear function $U(q, h) = w_q q + w_h h$, where $w_q > 0$ and $w_h < 0$, expressing a desire for high bandwidth and low cost. This utility function is easy to compute while still allowing the description of a wide range of user preferences.

This multistate model is a natural extension of the single state MDP model described earlier. Though the service pro£les and user utility functions in this model have been chosen for ease of computation, the multistate model provides a substantially more detailed view of the interactions between the agent and its environment, capturing the relationship between user utility and user actions as well as the effect of agent actions on the current service pro£le.

Due to the multistate nature of this model, however, the approaches for solving the single state bandit problem cannot accurately learn the optimal agent actions. The bandit solutions can learn which action yields the greatest reward for any given state, but in order to maximize the total return the agent must take into account the value of other states as well.

Possible solutions for this problem include dynamic programming, Monte Carlo methods, and TD learning [Sutton and Barto, 2002]. Dynamic programming is not appropriate because the rewards at each state are not known a priori. Monte Carlo approaches are inadequate because they learn policies off-line; the non-episodic nature of the PR problem requires an on-line solution that can learn a policy as it interacts with the user. In contrast to the other two approaches, TD learning works well for non-episodic tasks with unknown rewards. Of the TD(λ) solutions, we choose to examine the 1-step backup methods as an initial approach.

Many 1-step backup TD control methods exist, including Sarsa, Q-learning, and actor-critic methods. Q-learning is an off-policy method that learns the optimal policy regardless of the policy used. In contrast to Q-learning, Sarsa is an on-policy method that takes the current policy into account in its action-value estimates. It operates according to the following update rule:

$$\begin{aligned} Q(s_t, a_t) \quad \leftarrow \quad & (1 - \alpha)\, Q(s_t, a_t) \\ & + \alpha \left[r_{t+1} + \rho Q(s_{t+1}, a_{t+1}) \right] \end{aligned} \quad (3)$$

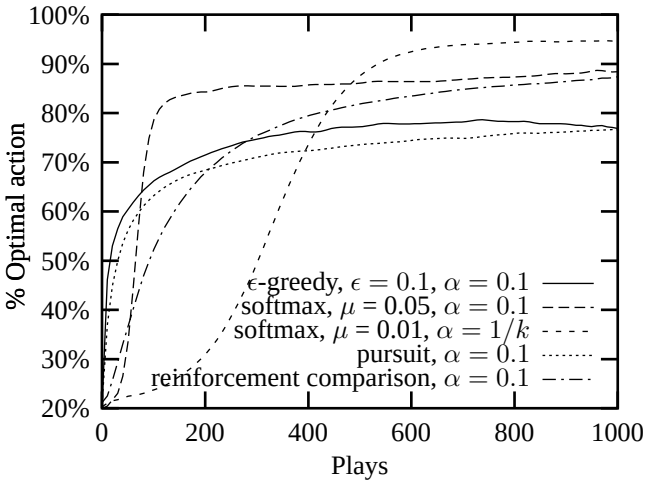

Figure 1: Stationary Bandit User Model

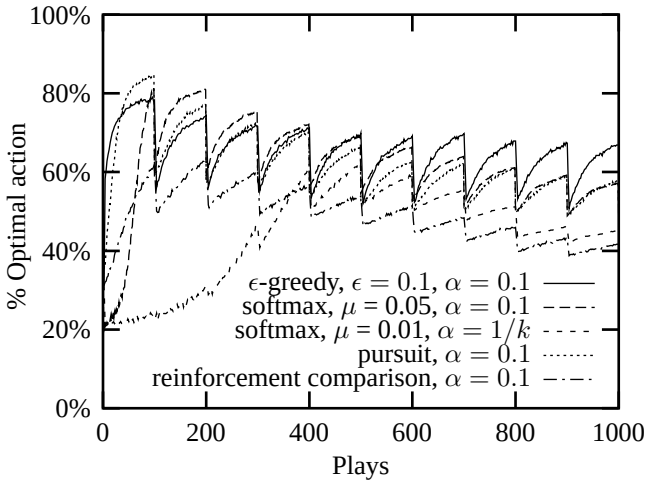

Figure 2: Nonstationary User Model, $\theta = 100$, $\eta = 0.4$

where $s_t \in P^{\delta}$ is the current state, s_{t+1} is the next state, $a_t \in A^{PR}$ is the current agent action, a_{t+1} is the next agent action according to the policy, α is a constant weight, r_{t+1} is the reward received in the next time step, and ρ is the discount factor.

Both learning methods work well with a wide range of policies, including ϵ-greedy methods and Gibbs softmax algorithms. The advantage of Sarsa is that its action-value estimates accurately re¤ect the action values of the policy used, whereas Q-learning always learns estimates for the optimal policy. If the policy converges to the greedy policy, however, then Sarsa will eventually learn the optimal policy.

6 Simulations

6.1 Single State Bandit Model

In order to demonstrate the agent's ability to learn user preferences using reinforcement learning, we simulated the single state bandit solutions described in section 4. To illustrate a wide range of different approaches, we have selected an ϵ-greedy method, two Gibbs softmax methods with initial temperature $T_0 = 10$ and cooling rates $\mu = \{0.05, 0.01\}$, a pursuit method, and a reinforcement comparison approach. All of these implemented exponential averaging with $\alpha = 0.1$ except for the $\mu = 0.01$ softmax which uses $\alpha = 1/k$, giving equal weight to all samples.

Figure 1 shows the observed results for stationary user preferences. The plot shows the average of 5000 tasks, each consisting of 1000 plays, or time steps. At the start of each task, the reward probabilities $Q^*(a_i)$ are initialized with random values between -1 and 1 for each of the £ve agent actions $a_i \in A^{PR}$. The plot shows the percentage of optimal actions for each play, where the optimal action is de£ned as the action providing the greatest expected reward.

The £gure shows how the choice of learning method affects the speed at which the optimal action is learned as well as the long term average reward. The ϵ-greedy and pursuit methods improve very rapidly initially, but soon reach asymptotic levels at approximately 80%. In contrast, the Gibbs softmax method with $\mu = 0.01$ makes poor selections in the £rst few hundred plays while it explores, but eventually selects the optimal action 95% of the time. The other algorithms achieve optimality at various speeds in between these two extremes. in summary, the data shows the trade-off involved between exploration and exploitation; the more time the agent spends exploring, the better the policy it can learn in the long run.

Figure 2 shows the observed behavior for nonstationary preferences that change occasionally. In this simulation, the agent begins with knowledge of the user's preferences. On the £rst play, and every 100 plays afterwards, the reward probabilities $Q^*(a_i)$ for each action is randomly increased or decreased by a constant magnitude of $\eta = 0.4$ (see section 4). In the steady state, the ϵ-greedy method performs the best, selecting the optimal action up to 66% of the time. The nonstationary data shows that when the user's preferences change the agent must use recent observations in its estimates and does not have much time to explore. In this situation, the ϵ-greedy approach quickly £nds a good action and exploits it, but the reinforcement comparison method spends too much time exploring while the $\alpha = 1/k$ softmax algorithm fails to discount old observations.

One would expect that as the rate of user preferences change increases it becomes more dif£cult for the agent to learn their preferences. Figure 3 con£rms this expectation by showing the effect of θ (frequency of change—see section 4) on performance. For chosen values of θ between 1 and 1000, we simulated each agent model for 3000 tasks and 1000 plays using $\eta = 0.1$. For those values of θ, the plot shows the percentage of optimal actions for the play just before the last change in user preferences. At the left when preferences change frequently, the ϵ-greedy method performs the best. As we move to the right we approach the stationary case; as before, softmax with $\mu = 0.01$ performs the best while ϵ-greedy performs more poorly.

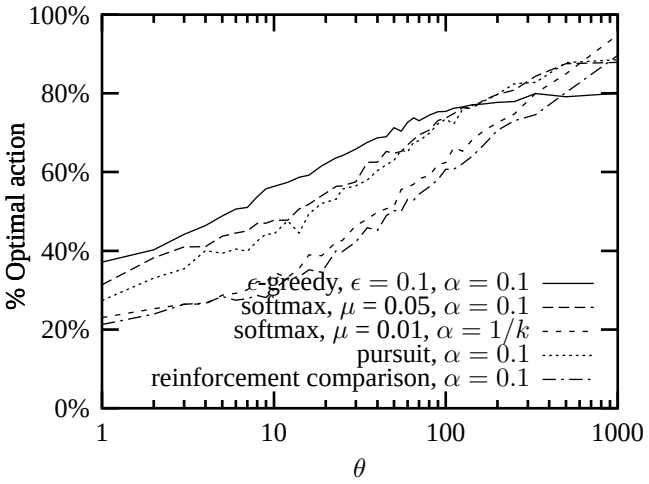

Figure 3: Nonstationary User Model, $\eta = 0.2$

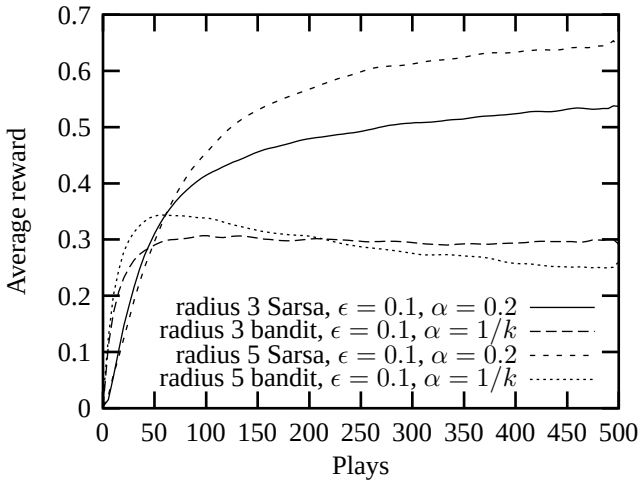

Figure 4: Multistate Stationary User Model

6.2 Multistate MDP Model

We have seen that in the single state case, the agent can learn user preferences for stationary and nonstationary user preference models. The reinforcement learning methods described in 5 allow us to learn these preferences in multistate models as well. Figure 4 contrasts the performance of a Sarsa TD learning approach with a single state bandit method in a simulation over 10,000 tasks. In this simulation, the set of service pro£les P^{δ} consists of all integer bandwidth/cost pairs (b, h) within a 3 unit radius of the initial service pro£le $s_0 = (5, 5)$. At the start of each task, the user utility function $U(q, h) = w_q q + w_h h$ is initialized randomly with $0 < w_q < 1$ and $-1 < w_h < 0$. The expected reward is given by the function $r(s_i) = 1 - \frac{2}{1+e^{-U(s_i)}}$. Though the ϵ-greedy bandit method learns the rewards for each action, it does not accurately compute the long term value as well as the Sarsa method. The advantage of TD learning becomes even more apparent as the state space increases; when the radius is 5, the Sarsa approach obtains 2.5 times the reward of the bandit method, illustrating the effectiveness of a TD learning approach over a bandit method in a multistate PR model.

7 Conclusions and Future Work

In this paper we described a user-modeling problem for the domain of wireless services. An agent, called a Personal Router, was proposed as a solution to this problem. We showed how the nature of the problem bounds the information set of the agent. We then presented a formal model of the service selection problem and showed how it can be captured in an MDP representation. Heuristic solutions from reinforcement learning were then empirically evaluated for two simpler MDP models of the PR problem.

There are a number of future directions. Our £rst goal is to continue to empirically evaluate different learning algorithms for increasingly more complex MDPs. The performance of the resulting agents will then be evaluated for ecological validity in controlled user experiments, the outcomes of which will be used for further (re)design-simulation-user experiment development, followed by £eld testing. Finally, our longer term goal after achieving a satisfactory level of performance of the agent is to extend the single agent decision mechanism to multi-agent systems (MAS) . In particular, we envisage not only negotiation decision mechanisms within a MAS but also that MAS mechanisms (such as distributed reputation, gossiping, and/or collaborative £ltering mechanisms) can be useful information sources for parameters of the agent preference elicitation model.

References

[Boutilier, 2002] Craig Boutilier A POMDP Formulation of Preference Elicitation Problems In *Proceedings of American Association of Arti£cial Intelligence*, pages 239–246, Edmonton, Alberta, Canada. 2002.

[Chajewska *et.al.*, 2000] U. Chajewska and D. Koller and R. Parr Making rational decisions during adaptive utility elicitation. In *Proceedings of the Seventeenth National Conference on Arti£cial Intelligence*, pages 363–369, Austin, TX, 2000.

[Faratin *et.al.*, 2002] P. Faratin and J. Wroclawski and G. Lee and S. Parsons The Personal Router: An Agent for Wireless Access. In *Proceedings of American Association of Arti£cial Intelligence Fall Symposium*, pages 13–21, N. Falmouth, MA, 2002.

[Kaelbling *et.al*, 1996] L.P. Kaelbling and M.L. Littman and A.W. Moore Reinforcement Learning: A Survey. *Journal of Arti£cial Intelligence Research*, 4:237–285, May 1996.

[Keeney and Raiffa, 1976] R. L. Keeney and H. Raiffa *Decisions with Multiple Objectives*. John Wiley and Sons, NY, 1976.

[Sutton and Barto, 2002] R.S. Sutton and A.G. Barto *Reinforcement Learning*. MIT Press, Cambridge, MA, 2002.