Social User Agents for Dynamic Access to Wireless Networks

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

We present a personal wireless access agent for mobile users who require wireless access for multiple, concurrent and varied tasks in different locations. Furthermore, the users are assumed to have cognitive and motivational barriers to providing subjective preference information to the agent. The task of the personal agent is then to dynamically model the user, update its knowledge of a market of wireless service providers and select service providers that satisfies the user's expected preferences based on minimal or missing information. In this paper we then show how the user modeling problem can be represented as a Markov Decision Process and suggest a number of single and multi-agent mechanisms as possible candidate solutions for the problem.

Introduction

Personal agents are autonomous systems whose decision making mechanisms must be highly responsive to their user's complex and evolving problem contexts. Furthermore, the user and the agent form a highly coupled feedback system because the user's needs are often difficult to elicit and consequently the agent's decision making mechanism is required to solve its problems under conditions of high uncertainty which in turn effect the user. We are interested in the design of both agent decision mechanism and supporting single and multi-agent information mechanisms for a personal agent, called the Personal Router (PR), whose goal is to dynamically provision wireless access for mobile users (Clark & Wroclawski 2000; Internet & Telecoms Convergence 2002; Faratin *et al.* 2002). However, the design problem is complicated due to the complexity in the user's context (Faratin *et al.* 2002). For example, current connection(s) may instantly become unreachable as mobile users change locations. Alternatively, different services may be needed as users dynamically change their preferences or begin different tasks. In addition to modeling the complexity of the user context the agent's decision making mechanism must operate with uncertain and incomplete information. For example, users may be reluctant to engage in costly communications with the agent over their preferences, specially if the elicitation space is combinatorially large. In the worst case users may be ignorant of their 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). A net result of this combined complexity and sparseness of information is our inability to use classical utility analysis techniques, such as conjoint analysis, to elicit user preferences (Keeney & Raiffa 1976; 1980).

Instead we propose single agent and multi-agent learning mechanisms for the user modeling problem. In particular, we model the agent problem with a Markov Decision Process (MDP) framework and then present some initial contributions on how to integrate learning and social mechanisms within the MDP framework.

The paper is organized as follows. A general description of the PR problem is briefly described in the first section. We then present a formal model of the service selection problem. We then show how this problem description can be computationally represented within a Markov Decision Process followed by how an agent might use the combination of decision mechanism of an MDP and other information mechanisms to develop a model of the user. Finally, we presents our conclusions together with the directions of future research.

The Personal Router

The Personal Router (PR) is a device that functions as an interface between the user devices and the Internet. In general, the goal of the PR is to deliver wireless services to the user that perfectly satisfy her complex requirements and minimize the user interactions with the system. However, in the absence of perfect information about the user the PR is likely to select inappropriate services that causes the user to experiment with the attributes or features of a PR selected service by continually interacting with the system. The features of a service we consider important in our applications are both the perceived quality and the price of a service. Users are given a user interface to manipulate these features as free variables via *better* and *cheaper* buttons on the PR respectively. The assumption we make is that 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 satisfies the user's current tasks and goals.

The following are some of the important features of the PR problem (see (Faratin *et al.* 2002) for a more complete discussion of the problem and it's features). Firstly, there are high uncertainties associated to not only the buyers and sellers decisions and actions, but also the agent's model of the user. For example, a buyer may not be sure whether she likes a service until she tries it. Conversely, due to the complexities of network model a seller may not be able to guarantee the quality of service that they advertise. Secondly, the user context is highly complex and is defined 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, file 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. Therefore the PR task of learning the user's preferences becomes a non-trivial problem. However, 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 are low.

Finally, we assume there exists a social network of other PR agents who are willing to (or can be induced to) share information at various levels of abstraction about their user's preferences and/or the state of the network. Therefore, the PR of each user has access to a distributed group preference and network model that can be used to infer (and learn) user preferences and update it's knowledge of the network services in order to improve the accuracy of service selection.

Representing the Problem as a Markov Decision Process

In this section we present the PR problem within the Markov Decision Process (MDP) modeling framework (Kaelbling, Littman, & Moore 1996; Boutilier, Dean, & Hanks 1999).

Problem Elements

We condition each service selection process instance on the current context of the user. As mentioned above a user context includes the current running application set, the time deadline and the location of a user for current goal. We let C represent the set of all possible contexts and $C^g \subseteq C$ be the set of contexts that are partitioned by the user goal g. An element $c \in C$ is composed of the tuple $c = \langle \beta, \gamma, \delta \rangle$, where β, γ and δ represent the set of running applications, user deadlines and locations respectively. Then, a particular user context $c^g \in C^g$, partitioned by the goal g, is defined by the tuple $c^g = \langle \beta^g, \gamma^g, \delta \rangle$, where β^g, γ^g and δ represent the set of running applications compatible with current goal g, the user deadline for current goal g and the concrete location of the user respectively. The location of a user at any instance of time is represented by both the physical location as well as the temporal location.

Next we let $\mathbf{P}$ represent the set of all possible service profiles, where each element of this set $P \in \mathbf{P}$ is composed of n features f_i, $P = (f_1, \ldots, f_n)$. Because service profiles available at any time change due to both user roaming (given a nomadic user) and changes in service offerings (given service providers' uncertainty in the state of the network) then we assume the (physical and temporal) location δ of a user partitions the set of possible service profiles available. Therefore we let $P^\delta \in \mathbf{P}$ represent the subset of possible service profiles available to the user in location δ.

Next let the set of all user preferences be given by $\mathbf{U}$. We then let each element of this set, $U \in \mathbf{U}$, represent a unique orderings over all the possible pairs of service profiles $\mathbf{P}$, or $U = (P_i \succ P_j, \ldots, P_{l-1} \succ P_l)$[1] for all combination of l profiles. Similarly, the current user context and goal partition the set of all possible preference orderings, or $U^{c^g} \subseteq \mathbf{U}$.

The ordering generated by U can then be captured by a utility function u such that:

$$u(P_i) > u(P_j) \quad \text{iff} \quad P_i \succ P_j \tag{1}$$

One possible utility function is the simple weighted linear additive model:

$$u^{c^g}(P_i) = \sum_{j=1}^{n} w_{ij}^{c^g} v(P_{ij}) \tag{2}$$

where $u^{c^g}(P_i)$ is the utility for profile i in context c given user goal g. $w_{ij}^{c^g}$ in turn is the weight that the user attaches to feature j of profile i in context c and user goal g. Finally, $v(P_{ij})$ is a function that computes the value (or goodness) of a feature j of profile i.

The MDP Model

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. Each system state $S \in \mathbf{S}$ is specified by a set of variables that completely describe the states of the problem. The value of each state variable is either discrete or continuous but with the constraint that each state's variable values be unique. In our problem each system state $S \in \mathbf{S}$ is fully described by the combination of: a) the user context ($c^g = \langle \beta^g, \gamma^g, \delta \rangle$) for goal g, b) the set of profiles 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 $S^t = (\beta^g, \gamma^g, t, loc^g, P, I)$, where $\beta^g, \gamma^g, t, loc^g$ represent the context of the user for goal g. Note that for reasons to be given below we disaggregate δ, the user location and time, to two state variables t and loc^g, the location of the user in temporal and physical space respectively. We can also specify user goals g in a similar manner by a subset of system states $S^g \subseteq \mathbf{S}$.

[1]The operator $\succ$ is a binary preference relation that gives an ordering. For example, $A \succ B$ iff A is preferred to B.

The other element of a MDP is the set of possible actions **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 **S**. In an MDP these transitions are represented by links **L** between nodes that represent the transition of a system state from one configuration to another after performing some action. Additionally, each link has an associated value that represents the cost of the action. In our problem the set of actions A available to the user u are defined 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.[2] The consequences of user actions are changes in values of state variables $\beta^g, \gamma^g, t, loc^g, P, I$; that is, changes in either: a) the user context (changes in running applications, the time deadlines for connections, the current time, the user location and/or price/quality demands, observed by interaction with the PR via better and cheaper responses) or b) the set of currently available profiles or the combination of the state variables.

The set of actions A available to the PR are defined by the set $A^{PR} = \{\Delta^{P_i \rightarrow P_j}, \phi\}$ representing PR dropping service profile i and selecting j and no action respectively. The consequence of a PR action is a change in the likelihood of future user interaction I, where decreasing likelihoods of user interactions is more preferred.

Additionally, in an MDP the transitions between states 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.

Finally, we can compute the utility of a service profile i in context c for goal g (or $u^{c^g}(P_i)$—see equation 2) 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 $U(S_m)$. However, since in the formal model above a goal partitioned the set of all possible contexts, that in turn partitioned the ordering of profiles, so likewise the utility of a state m is computed by the function $U(P_i, c^g)$, the conjunction of both the the utility of a context given a user goal, $U(c^g)$ and the current profile given the context ($U(P_i|c^g)$). That is:

$$U(S_m) = U(P_i|c^g) \wedge U(c^g) \tag{3}$$

where $\wedge$ is the combining operator (Shoham 1997).

The problem of *estimating* and updating the (link) probabilities and (state) utilities of an MDP is described in the sections below.

[2]Note, that since time is an element of the state description then the system state always changes in-spite of no action by either the user or the PR or both. Furthermore, the granularity of time is likely to be some non-linear function of user satisfaction, where for example time is finely grained when users are not satisfied with the service and crudely grained when they are satisfied. However, the granularity of time is left unspecified in our model.

Reasoning with MDPs

The MDP formulation of the service selection problem gives us a representational framework to model the user behaviour and preferences as the combination of state transition probabilities and utilities. Reasoning with an MDP (or user modeling in our problem) in turn is taken to mean *both*:[3]

- solving an MDP and
- updating the transition and utility estimates over the state space.

Solving an MDP

On each time step solving an MDP is simply defined by finding a policy π that selects the optimal action in any given state. There are a number of different criteria of optimality that can be used that vary on how the agent takes the future into account in the decisions it makes about how to behave now (Kaelbling, Littman, & Moore 1996). Here we consider the finite horizon model, where at any point in time t the PR optimizes its expected reward $E(\sum_{t=0}^{h} r_t)$ for the next h steps, where r is the reward the PR receives which in our problem domain is the utility of the user, observed as interactions with the cheaper/better button. Therefore, the model allows the contribution derived from future h steps to contribute to the decisions at the current state. Furthermore, by varying h we can build agents with different complexities, ranging from myopic agents $h = 1$ to more complex agents $h > 1$.

Given a measure of optimality over a finite horizon of the state-space solving an MDP (or selecting the best policy) is then simply selecting those actions that maximize the expected utility of the user (see example in section above):

$$\pi = \arg\max E(\sum_{t=0}^{h} U_t) \tag{4}$$

Such a function is implemented as a greedy algorithm.

Estimating and Learning Probabilities and Utilities

The other component of reasoning with the MDP is how to form an initial estimate and subsequently update model parameters values (transition probabilities and utilities) that can be used algorithmically given the MDP representation.

Mechanisms for estimating initial model parameters are detailed below. We further note that these initial beliefs over transitions and utilities can then be subsequently updated using reinforcement learning. In classic reinforcement learning this is achieved by using the reward signal r to incrementally update the true estimate of the costs from each state to the goal state. Then the PR maximizes the expected reward given the beliefs. However, under the reinforcement mechanism the agent needs to not only know the goal of the user, but the mechanism also requires the goal context to be repeated in time so that the PR can learn the true costs of

[3]See (Faratin *et al.* 2002) for solutions to the problem of intractability in the size of the state-space.

paths to the goal state in an incremental fashion. Unfortunately, these two assumptions cannot be supported by the service selection problem because of complexity in reasoning about the user goals (since user may not be able to formulate and/or communicate goals) and the low likelihood of user having same repeated goals for the PR to learn from. However, the PR does have access to the utility information at each state. Therefore, rather than using the value of the goal state as the reference point in the optimization problem we instead propose to use the value of each state explicitly.

Model Estimation Strategies

Estimation strategies we consider can be usefully categorized along two dimensions—single v.s multi agent and model v.s observation based approaches.

A single agent model based solution to the problem of deriving the agent's initial beliefs over the state space is to simply design agents with (domain) knowledge that represent the transition probabilities along each dimension of the MDP, modeled by some distribution with a given mean and standard deviation. For example, as a first case approximation we can assume that the probability of a user changing location increases with time. Likewise, an agent can form some initial belief over the utility of each state according to some permissible heuristic such as equal utility to all states. An alternative single agent approach may be to *derive* a user model indirectly through observation of user behaviour rather than a model constructed at design time. Below we consider an example of the latter single agent mechanism.

An alternative solution to the belief and utility estimation problem is to use multi-agent mechanisms to specify missing or uncertain user information needed for the agent decision making. For example, collaborative filtering mechanisms have been used extensively to make individual recommendations based on group preferences (Resnick *et al.* 1994; Breese, Heckerman, & Kadie 1998). Similarly, we can use the user preference information from a large number of users to predict state values (for example predicting the perceived quality for a service profile based on the preferences of users with similar quality functions) or transition probabilities (for example likelihood of changing locations). Furthermore, such a mechanism can either be centralized or decentralized. In the former mechanism each PR send its user preference information to a centralized recommendation server (RS). Individual PRs can then query the RS for state information (such as quality estimates of service profiles) and The RS then attempts to identify users with similar quality functions and generates a quality estimate. Alternatively, in a decentralized mechanism (or gossiping/epidemic mechanisms (Pelc 1996; Hedetniemi, Hedetniemi, & Liestman 1988)) each PR communicates not with a center but with a small subset of individuals in a Peer-to-Peer manner. The choice of which mechanism is often dependent on the trade-offs involved in the system properties (such as flexibility, robustness, etc.) and the quality of the information content of the mechanism.

Updating Transition Probabilities

Assume for now that the agent has some initial probabilities for each transition state (a mechanism for deriving these initial probabilities using data from the group model as described in section below). Then a single agent mechanism can be constructed that updates the transition probabilities based on the observation of actual state transitions made by the user. In particular, we propose the simple exponential weighted moving average as the update rule. Let $S_j, \ldots, S_n$ be the possible transitions from state S_i. Let $Pr_a(S_j|S_i)$ be the probability of going to state S_j from S_i after performing action a. Then on a state change from S_i to S_j on action a, the probabilities are updated as follows:

$$\begin{aligned} Pr_a(S_k|S_i) &\leftarrow \alpha(1 - Pr_a(S_k|S_i)) \text{ for all } k \neq j \\ Pr_a(S_j|S_i) &\leftarrow \alpha + (1-\alpha)Pr_a(S_j|S_i) \end{aligned}$$

where the constant α controls the weight given to most recent observation of the value of the transition link.

Estimating and Updating Utility of States

We next consider mechanisms for reasoning over the utility of a state. Earlier we argued that *part* of the utility of a state is derived from the utility of a service profile (equation 3). We then proposed a candidate linear multi-criteria utility model (equation 2) that aggregates the values of each service profile feature j into a scalar value, representing the overall value of the service profile P_i. From the perspective of the user the two features of a service considered important are the service price and quality. However, the representation that is used at the network level for representing service quality between the service providers and the PR can be ontologically distinct from the representations at the user level. For example, quality may be described in terms of service latency or bandwidth, or alternatively, as discrete user level objects such as "Gold Service" that encapsulates the network details from the user. In either case, in order to compute $v(P_{i,quality})$ the PR must learn the mapping between the service description and the quality of a service the user experiences.

Mapping Service Features to Service Quality

Firstly, the quality of a service profile $v(P_{i,quality})$ depends not only on the user's preferences but also the current context c^g. For example, a high bandwidth, high latency service may have very high quality for bulk transfer, but has low quality for IP telephony. In this section we propose a single agent mechanism for modeling the service quality perceived by the user for a particular activity. We then show how the resulting model can be refined with user feedback.

We make the assumption that if a user uses a service profile P_i in a context c^g, then the PR can implicitly learn from user "better" feedback data which we call the user's quality rating $q_{P_i}^{c^g}$. The mechanism then uses this inferred quality ratings as data points around which to build the quality function $v^{c^g}(P_{i,quality})$. The quality function is constructed

by interpolating between these data points in a piecewise-linear manner. Consequently, if the PR has learned the quality rating $q_{P_i}^{c^g}$ of service profile P_i for context c^g, then $v^{c^g}(P_{i,quality}) = q_{P_i}^{c^g}$. Otherwise, for a new profile P_j, the value of $v^{c^g}(P_{j,quality})$ is linearly interpolated from the nearest known values of $q_{P_i}^{c^g}$. We can plot $v^{c^g}(P_{i,quality})$ in a $n+1$ dimensional space, where n is the number of service profile features. Then for any service profile P_i, we can interpolate the value of $v^{c^g}(P_{i,quality})$ from the $n+1$ nearest neighbors that enclose P_i using repeated piecewise-linear interpolation.

Using a piecewise-linear function has several advantages. This approach is scalable because the PR only needs to consider the quality of similar service profiles to determine the quality of a new profile. Furthermore it does not require any prior knowledge to generate. Piecewise-linear functions can closely approximate any monotonic function, and we expect that perceived quality is likely to be monotonic with many features including bandwidth and latency. In addition, the function will tend towards better estimate of the true quality function as the PR collects more data. It is also robust because changing the quality of one service profile only affects the rating for a few similar profiles.

User's quality rating $q_{P_i}^{c^g}$ is inferred in the following manner from user feedback. The mechanism decreases $q_{P_i}^{c^g}$ when the user presses the "better" button (the user is dissatisfied with the quality of the current service P_i). Conversely, $q_{P_i}^{c^g}$ increases, according to an increasing convex function of service profile usage time, if the user stays with the same service profile for a long time (the longer a user stays with the same profile, the more likely they are satisfied). However, the following constraints must be met by the mechanism. Firstly, the user requires some time to evaluate the service before the PR updates $q_P^{c^g}$. Also, the user can only evaluate a service if it is actively being used to transfer data for some amount of time. Furthermore, if the current service profile is not limiting network performance, then pressing the "better" button does not decrease the quality rating. Finally, confidence in quality estimates should grow as the PR gathers more data about a service profile. Consequently, the quality estimate should change more slowly with user feedback. The updating the quality function is described procedurally in figure 1.

This update mechanism has several important features. Service quality estimates get more accurate with time and user feedback. If a service P_i is overrated, then the user will soon request a better profile and cause $q_{P_i}^{c^g}$ to decrease. Conversely, if a service is underrated, then its rating will increase as it is used more by the user. Furthermore, the mechanism does not depend on the time-scale of button presses. The relative quality of different service profiles remains unchanged if the time between button presses is scaled by a constant factor, as long as τ is small.

Multi-Agent Mechanisms

Initial service selection is a problem in the above single agent mechanism because the mechanism requires user feedback to infer $q_P^{c_g} i$. In cases of missing information

- Let $r_{P_i}^{c^g}$ be the quality rating of profile P_i determined using the group model.
- Let τ be a constant representing the minimum time of network usage required for a user to evaluate a service profile.
- Let ϕ be a variable representing the degree of confidence the PR has in the rating.
- Let $C_d(\phi)$ and $C_i(\phi)$ be factors depending on the level of confidence ϕ controlling the rate of rating decrease and increase, respectively. The function $C_d(\phi)$ increases and gets closer to 1 as ϕ increases, while $C_i(\phi)$ decreases and gets closer to 0 as ϕ increases.
- Let ϵ be a constant controlling how frequently the quality function gets updated.

1. If P_i has not been initialized yet,
   - If P_i is not similar to previously seen profiles, then $q_{P_i}^{c^g} \leftarrow r_{P_i}^{c^g}$.
   - Else $q_{P_i}^{c^g} \leftarrow v^{c^g}(P_i)$.
2. If the user presses the "better" button,
   - If the current service profile is not limiting network performance or if $t < \tau$, ignore the button press.
   - Else $q_{P_i}^{c^g} \leftarrow C_d(\phi) q_{P_i}^{c^g}$ and switch to a higher quality service profile.
3. Else if $t > \tau$ and at least ϵ time has passed since the last update,
   - $q_{P_i}^{c^g} \leftarrow q_{P_i}^{c^g} + C_i(\phi)/q_{P_i}^{c^g}$
   - $\phi \leftarrow \phi + \epsilon$
4. If the user presses the "cheaper" button, switch to a lower price service profile.
5. If the user presses the "undo" button, $q_P^{c_g} i$ reverts to its previous value and the PR switches to the previous profile.

Figure 1: Quality Rating Procedure

multi-agent mechanisms can instead be used to reason about the likely quality preference of the user. Indeed, even in cases when the PR information set is adequate for the task of estimating a quality function, a locally inferred model derived from single agent mechanism can still be correlated and compared to other users' models in a multi-agent mechanism. One such mechanism is collaborative filtering techniques where the PR can use the group's quality preferences as a better indicator of quality function. Large number of data points provided by large number of users can then be used to predict a user's perceived quality for a service profile. We predict service profile quality based on the quality preferences of users with similar quality functions. Each PR collects user quality preference information from other nearby PRs in a distributed peer-to-peer network. The PR then uses a collaborative filtering algorithm to extract infor-

mation from users with similar preferences.

The mechanism is formally defined by the following steps:

1. Given a user x, a service profile P_x, and an activity g:
2. Let $S_x = \{(P_k, h)|$ user x has used service profile P_k for activity $h\}$.
3. For every user y known to the PR and for every service profile and activity pair $(P_i, g) \in S_x$, evaluate $v_q^{c^g}(P_i)$ for user y.
4. Use a collaborative filtering recommendation algorithm to obtain a value for $q_{P_x}^{c^g}$ for user x.

As we noted above this mechanism can also be used to initialize transition probabilities.

1. Given a user x, states S_i and S_j, and an action a:
2. Let $T_x = \{S_k|$ user x has been in state S_k n times $\}$, where n is a confidence threshold.
3. For every user y known to the PR and for every state S_k in T_x, get the value of $Pr_a(S_l|S_k)$ for all states S_l.
4. Use a collaborative filtering recommendation algorithm to obtain a value for $Pr_a(S_j|S_i)$ for user x.

The confidence threshold n is set to improve the quality of the data. If the PR has not seen a state very frequently, then the data may not be very accurate.

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, defined by $\langle \mathbf{S}, \mathbf{A}, \mathbf{T}, \mathbf{C}, \mathbf{U} \rangle$, the set of all possible system states, user and PR actions, transitions between states, costs of action and utility of states. We also hypothesized on how we can solve the information problem using multi-agent and reinforcement learning mechanisms.

There are a number of future directions. Firstly, we are currently evaluating other single agent (e.g Bayesian) mechanisms for capturing and updating initial values for the parameters of the developed MDP. Next we aim to develop solution strategies that can tractably solve the MDP, for making real-time decisions. Strategies currently under consideration include not only state-space pruning strategies but heuristics for solving piecewise MDPs for time, location and applications. Once the resulting MDP has been solved optimally on a tractable decision space we seek to compare the efficiency of heuristic algorithms that can be scaled up to larger search spaces. Another long term goal is to assess the appropriateness of qualitative information within the developed framework. Finally, the overall goal of this project is to develop system prototypes and perform user-centered field trials.

References

Boutilier, G.; Dean, T.; and Hanks, S. 1999. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research* 11:1–94.

Breese, J. S.; Heckerman, D.; and Kadie, C. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence*, 43–52.

Clark, D., and Wroclawski, J. 2000. The personal router whitepaper. *MIT Technical Report*.

Faratin, P.; Wroclawski, J.; Lee, G.; and Parsons, S. 2002. The personal router: An agent for wireless access. In *Proceedings of American Association of Artificial Intelligence Fall Symposium on Personal Agents, N. Falmouth, Massachusetts, US*.

Hedetniemi, S. M.; Hedetniemi, S. T.; and Liestman, A. 1988. A survey of broadcasting and gossiping in communication networks. *Networks* 18:319–351.

Internet & Telecoms Convergence. 2002. `http://itc.mit.edu`.

Kaelbling, L.; Littman, M.; and Moore, A. 1996. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4:237–285.

Keeney, R. L., and Raiffa, H. 1976. *Decisions with Multiple Objectives*. New York: John Wiley and Sons.

Keeney, R. L., and Raiffa, H. 1980. *Design and Marketing of new products*. New York: Prentice-Hall.

Pelc, A. 1996. Fault-tolerant broadcasting and gossiping in communication networks. *Networks* 28:143–156.

Resnick, P.; Iacovou, N.; Suchak, M.; Bergstorm, P.; and Riedl, J. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, 175–186. Chapel Hill, North Carolina: ACM.

Shoham, Y. 1997. A systematic view of utilities and probabilities. In *International Joint Conference on Artificial Intelligence, IJCAI*, 1324–1329.