Giving Advice to People in Path Selection Problems

Amos Azaria
Department of Computer Science
Bar Ilan University
Ramat Gan, Israel
azariaa1@cs.biu.ac.il

Zinovi Rabinovich
Department of Computer Science
Bar Ilan University
Ramat Gan, Israel
zr@zinovi.net

Sarit Kraus
Department of Computer Science
Bar Ilan University
Ramat Gan, Israel
sarit@cs.biu.ac.il

Claudia V. Goldman
General Motors Advanced Technical Center Israel
claudia.goldman@gm.com

Omer Tsimhoni
General Motors Advanced Technical Center Israel
omer.tsimhoni@gm.com

ABSTRACT

We present a novel computational method for advice-generation in path selection problems which are difficult for people to solve. The advisor agent's interests may conflict with the interests of the people who receive the advice. Such optimization settings arise in many human-computer applications in which agents and people are self-interested but also share certain goals, such as automatic route-selection systems that also reason about environmental costs. This paper presents an agent that clusters people into one of several types, based on how their path selection behavior adheres to the paths presented to them by the agent who does not necessarily suggest their most preferred paths. It predicts the likelihood that people deviate from these suggested paths and uses a decision theoretic approach to suggest paths to people which will maximize the agent's expected benefit given the people's deviations. This technique was evaluated empirically in an extensive study involving hundreds of human subjects solving the path selection problem in mazes. Results showed that the agent was able to outperform alternative methods that solely considered the benefit to the agent or the person, or did not provide any advice.

1. INTRODUCTION

Research in multi-agent systems primarily encompasses systems composed of automated agents. Cooperative systems are usually described by a single utility function which all agents attempt to maximize. Competitive systems, on the other hand, may be designed and analyzed, for example, as zero sum games where the gain of one agent is the loss of another. In this paper, we focus on systems composed of both automated agents and human users. Although in general these interactive systems are cooperative, users and machines may have different interests. Each party may want to optimize different parameters, not necessarily at the expense of the other. In particular, we study automated agents interested in persuading their users to perform actions that increase the agent's utility.

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), June, 4–8, 2012, Valencia, Spain.
Copyright © 2012, International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

Machines can try to persuade their users to perform certain actions by implementing different methods. For example, machines could provide higher rewards (e.g., score, ranking stars, etc.) when users choose actions desirable by the agents. Automated agents may disclose information not available to their users in order to encourage them to take certain actions. For example, Azaria et al. [3] have shown that agents can provide correct, although partial, information about a state of the world (unknown to the user, but relevant to his decision) and thus persuade them to take certain actions beneficial to the agent. We can also consider agents providing advice (based on the agents' advantageous information or computational power) that may lead their users to choose actions that are beneficial to the agents.

In this paper, we focus on the last method: we study how to automatically generate advice that will encourage users to choose actions preferred by the automated system. We chose the domain of path selection to exemplify the algorithms and framework developed. In such a domain, human users and computers are self-interested, but also have shared goals. For example, consider an automatic system for suggesting commuting routes to a human driver. Both participants in this setting share the goal of getting the driver from home to work and back. However, each participant also has its own incentives. The driver wishes to choose the route that minimizes the commuting time, while the computer may prefer taking a longer route that emits fewer pollutants, or does not pass near schools and playgrounds.

The route selection domain is an example of a computationally demanding domain where even having complete knowledge is not enough for a user to solve such a problem optimally. As we will show in our experiments, finding the shortest path in large maps with many intersections may not be a trivial problem to solve. In such cases, the computer's advice might be perceived as helpful and trustful as it is coming from powerful computational software.

However, the development of methods to identify agent strategies for deciding which advice to give to people is challenging. First, it is known that people are not maximizers of their monetary value. When facing noisy data, people often follow suboptimal decision strategies. This bounded rational behavior [7] is attributed to: 1) sensitivity to the context of the decision-making; 2) lack of knowledge of the user's own preferences; 3) the effects of complexity; 4) the interplay between emotion and cognition and 5) the problem of self-control. Furthermore, people discount the advice they receive from experts [32, 5] and it was shown that if the adviser has a monetary stake in the advice being followed, people will follow its advice even less [21]. Finally, the learned model should be generalized to new environments as well as different people. To face these challenges we will integrate machine learning and psychological models for predicting human response to advice.

Our study includes a 2-participant task setting for choosing a path on a large colored grid that is analogous to the route-selection problem. The person's sole incentive is to choose the shortest path, while the agent's incentives also include the number of color changes in the path. Choosing a path on the grid corresponds, for example, to selecting a commuting route between home and work. The colors on the grid represent constraints, such as environmental and social considerations. Switching between colors on the path represents the violation of one of these constraints. The person's preferences consider the length of the route only, while the agent's preferences take into account both the length of the route as well as the number of constraint violations.

We developed the User Modeling for Path Advice (UMPA) approach for the generation of advice, comprised of a training stage and three additional steps required to learn from this data and to generate the agent's advice. We run, first, experiments with human subjects to collect data on how users react when provided with advice. The system proposes three types of advice in different testing scenarios: advice that is optimal to the user, advice that is optimal to the system and advice that considers both the user's and system's preferences. We found three types of user behaviors: those that follow the system's advice, no matter how bad this advice is subjectively perceived to be; those that ignore the advice and follow their chosen path; and those that modify the advised path. This last phenomenon is very interesting since just the fact that advice is provided affects the users' choices. The users' modifications may completely change the advice or their own choice, but this change occurs only as a result of having seen such a system proposal. In particular, we noticed that users of the third type took *cuts* when solving the route selection problem. Cuts are deviations from a suggested path and are alternative segments for connecting two local points in the original path. A cut may improve the path from the user's point of view by shortening it, but may decrease the benefit to the agent.

Once we collected this data, the UMPA approach proceeds to 1) learn the percentage of types of users who will follow, ignore or modify the given advice, 2) learn with what probability each cut will be chosen for a given advised path and 3) compute the best advice for the agent given the users' predicted types and behaviors.

We evaluated the UMPA approach in an extensive empirical study comprising more than 700 human subjects solving the path selection problem in four different mazes. The results showed that our UMPA agent outperformed alternative approaches for suggesting paths, based on either the users' or the system's preferences. In addition, people were satisfied with the advice provided by the UMPA agent. Given these encouraging results, we expect that the proposed technology can be applied to other applications where the agent's goal is to provide people with advice that will lead them to take beneficial actions. Recent applications, such as coaching humans in weight-loss programs, programs to help quit smoking or online service providers such as automated travel agents are domains that are promising.

2. RELATED WORK

Game theory researchers studied related research questions in the context of persuasion games. In these games, a speaker attempts to persuade a listener to accept a certain request [14, 15, 29, 30]. Most of these works make the strong assumption that people follow equilibrium strategies. However, agents that follow equilibrium strategies when interacting with people are often not beneficial [19, 24, 3]. This can be explained by the significant experimental and other empirical evidence which indicates that people may be nonstrategic when interacting in persuasion games [13, 11, 4, 10, 6, 28, 8].

Route or path selection has become one of the most prominent applications of computer assisted guidance (see a survey in [17]). In fact, route guidance systems using GPS have become pervasive over the years, thanks to the significant research effort in addressing both the cognitive limitations and the range of individual preferences of human users (e.g. [12, 23]). Many of the challenges in the development of route guidance systems stem from the high variance across individuals regarding their evaluation and acceptance of route advice. This variance makes it important to tailor route advice and guidance to a specific user. To this end, a wide range of machine learning techniques are used to capture and utilize user routing preferences (e.g. [23]).

Instead of tailoring routes to users, we model user attitudes towards route advice such that the choices made by the users, after being given advice, will be beneficial to the agent. There has been some work on driver acceptance of unreliable route guidance information [16]. Antos and Pfeffer [2] designed a cooperative agent that uses graphical models to generate arguments between human decision-makers and computer agents in incomplete information settings. They use a qualitative approach that does not model the extent to which people deviate from computer-generated advice. Other works have demonstrated a human tendency to accept advice given by an adversary in games [21]. Some theoretical analysis suggests this behavior to be rational [26]. To some extent, these results were used in the framework of large population traffic manipulation (either by explicitly changing the network topology or by providing traffic information, e.g. [20, 9]). However, to the best of our knowledge, we are the first to study the combination of human choice manipulation and the personal route selection problem in a given network.

3. THE MODEL

To allow a formal discussion of the path selection problem, we employ a maze model. We assume that a user has to solve the shortest path problem within a rectangular maze either by constructing a path or by considering a path suggestion. More formally, we define a *maze* M as a grid of size $n \times m$ with one vertex marked as the source S, and another vertex as the target T. Each vertex v is associated with a label $c(v)$, that we will refer to as the *color* of v. We will denote the white color or label number 0 as an *obstacle*. $x(v)$ and $y(v)$ denote the horizontal and the vertical grid coordinates of the vertex v, respectively. We assume that the user can

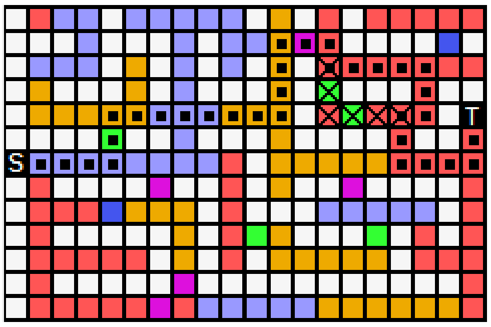


Figure 1: Path selection problem visualized in a small maze

move along the grid edges in the four standard directions: up, down, left or right. A sequence of vertexes that does not include an obstacle and can be traversed by moving in the four standard directions is a *valid path*. In the remainder of the paper, to distinguish between vertexes of different paths, we will denote them by the path's name with a superscript: e.g. vertexes of a path π will be denoted by $\pi^1, ..., \pi^l$. A valid path will be called a *full path* if $\pi^1 = S$ and $\pi^l = T$, i.e. it begins at the source node and ends at the target node, thus solving the maze.

The *path selection problem* is modeled as the user's task to find the shortest full path through the maze. Formally, we assume that the user's cost of a path π is equal to its length, i.e. $Cost_u(\pi) = l(\pi)$. In contrast, the agent's cost depends on the length of the path and also on the number of color switching done along the path. Formally, given a color switching cost W, the agent's cost $Cost_a$ of a full path π is given by: $Cost_a(\pi) = l(\pi) + W \cdot \sum_{1 \le i < l} \mathbf{1}\{c(\pi^i) \neq c(\pi^{i+1})\}$. We use the term *greedy path* to refer to a full path that minimizes $Cost_a$, and the term *shortest path* to refer to a full path that minimizes $Cost_u$. Notice, that there are multiple valid paths through a maze, and it is possible that there are many full paths as well.

Now, in addition to the maze grid, its color labeling and the source and target nodes, we also allow for a secondary labeling of a particular full path through the maze. This labeling represents the path advised by the agent to the user. We assume that the user is aware of this labeling prior to solving the path selection problem. In fact, the advised path is part of the input to *the path selection problem*. When a user is given a maze (with or without an advised path), his goal is to solve the maze by finding the shortest full path from source S to target T. However, due to the complexity of the maze, finding such shortest path may not be trivial or clear from looking at the maze during the limited amount of time given to the user. Therefore, the user may find it beneficial to take some advice provided to him regarding which route to choose.

The *best-advised path problem* is modeled as the agent's task to compute a full path that, once presented to a user, will yield the agent the lowest expected cost.

Figure 1 visualizes the formal setting in a small maze. In the figure, obstacles are represented by the white color, while the start and the target nodes are black. In turn, the dotted nodes represent the advised path, while the crossed nodes represent a valid (partial) path selected by the user.

4. THE UMPA APPROACH

We assume the availability of training data for the prediction stages (see experiments in Section 5). UMPA is given a training set, Ψ, of tuples (M', π, μ, α) collected from experiments where people were provided with advice and where: M' is a maze; π is an advised path through the maze; α is a binary variable indicating whether the user considers π as a good solution or not (α equal 1 or 0 respectively); and μ is the solution selected by a human user, who was presented with M' and π. In addition, we assume that Ψ includes examples (M', μ) collected from games where the agent was silent. Given a maze M (not in the maze set from the training examples), we employ a three-stage process to solve the best-advised path problem: (i) Cluster users into one of three types, depending on the extent to which their path selection behavior adheres to suggested paths that may be more beneficial to the agent than to themselves. Then, we predict the likelihood that a user will belong to one of these three clusters;(ii) predicting the likelihood that people deviate from a suggested path; (iii) generating the advised path using a decision theoretic approach which utilizes the prediction from the first two stages in order to compute the expected cost of the agent from a given path. In the next subsections we provide details of our implementation of each one of these steps.

Predicting human response to an advised path is difficult due to the diversity in people's behavior. We propose to integrate psychological models into the machine learning process. In particular, we have defined a **Seemliness-value** feature that measures the path's direction towards the target node's horizontal and vertical coordinates. This attribute will be used in the learning of UMPA. The feature value is based on the following principles known from behavioral science:

- Loss aversion [31] (Prospect theory) : losses are weighted roughly twice as much as gains. Therefore, while each step in the path toward the target contributes a single unit to the Seemliness-value, each step away from the target reduces two units from the value.

- Future discount [25]: losses or gains in the future are less important than current losses or gains. Therefore, while each step in the path toward the target at the beginning of the path adds one unit (and a step away from the target in the beginning of the path reduces two units), any consecutive steps' contribution is multiplied by a discount factor (which is exponential in the number of steps from the beginning of the path).

The total path Seemliness-value is calculated as a discounted sum of steps contribution along the path and is denoted $s(\phi)$. For an intuitive example, the dotted path shown in Figure 1 has a relatively high Seemliness-value since its earlier steps are in the target direction and steps in the opposite direction appear only later; however, in Figure 2 the dotted path has a relatively low Seemliness-value since the steps at the beginning of the path are in the opposite direction of the target.

4.1 Modeling Diversity in People's Reactions

Based on what was observed in the behavioral data collection experiments (as explained in Section 5), UMPA clusters users into three types: *Advice followers*, *Advice ignorers* and

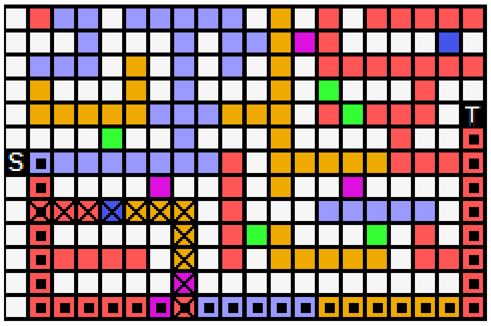


Figure 2: A second example of a path and a cut

Baseline users. Given a new maze, when considering a path to be given as advice, UMPA would like to estimate the probability of a user to belong to one of these clusters. For this task, it first labels the examples of Ψ with one of the three types and put the examples in Ψ^l.

The labels are determined as follows. *Advice followers* are users who follow the advised path blindly without modifying it, even when believing that it is not of good quality. That is, the user of an example $(M', \pi, \mu, \alpha) \in \Psi$ is labeled as *Advice follower* if $\mu = \pi$ and $,\alpha = 0$. Users that took the system's advice as provided and also believed that the advised path really did have good quality were included in the Baseline users type set (these users may have chosen the advice because it did have good quality and not because they were told to choose it).

However, most users would at least attempt to improve upon the advised path, or simply ignore it entirely. In order to characterize these users, we will introduce the concept of a *cut* and a *baseline solution*.

Given two vertices $,\pi^i$ and $\pi^{i'}$, of an advised path π, any path τ between these two vertices (that does not otherwise intersect with π) is termed a *cut*. Although there may be an exponential number of cuts, certain human cognitive tendencies (see e.g. [12, 27]) allow us to bound the maximal cut length. All users who deviated from the advised path solely by taking cuts were considered users who used the advice as a *baseline solution*, and therefore termed *Baseline users*.

More formally, given a valid path π, we define a cut τ of length l to be a valid path such that $\exists i, \tau^1 = \pi^i$ and $\exists i' > i, \tau^l = \pi^{i'}$ and $\forall 1 < i'' < l, \nexists j, \pi^{i''} = \pi^j$. The sequence of $\pi^i, ..., \pi^l$ will be called the original segment of cut τ and will be denoted by $o(\tau)$. Figure 1 and Figure 2 show examples for cuts marked by crossed nodes. We only consider cuts whose lengths are smaller than some threshold and also not much longer that their original segment. Formally, let $L_1 \in \mathbb{N}$ and $L_2 \in \mathbb{R}^+$, $l(\tau) \leq min\{L_1, L_2 \cdot l(o(\tau))$.

Finally, we define the *Advice ignorers* as all users who are neither *Baseline users* nor *Advice followers*. The relevant examples of Ψ were labeled accordingly. It is important to understand that being an advice follower does not depend on the specific maze and advice. However, deciding whether to ignore advice or use it as a baseline depends on the specific maze and advice.

Now, we can turn to compute the likelihood of users being associated with the different types as required in the first step of the UMPA approach. Based on the literature on route selection (see e.g. [18]), we presume that the proportion of *Baseline users* for the given advice π is strongly characterized by the overall Seemliness-value of π, denoted $s(\pi)$. In order to use the Seemliness-value of a path as an indicator for the proportion of Baseline users in that path, we first normalize the Seemliness-value by subtracting the average of all Seemliness-values of all paths that appear in the data-set and divide by their standard deviation. Once we have a standardized (scaleless) value, we assume it predicts a standardized proportion of Baseline users in that path, therefore, this value must be unstandardized using the appropriate units found in the data-set. Formally, given Ψ^l, UMPA generates a set of tuples $\pi', s(\pi'), prop(\pi')$ where $prop(\pi')$ is the proportion of users in Ψ^l that received the advice π' and are labeled as *Baseline users*. Denote by $AvgSV$ ($StdSV$) the average (standard deviation) of the $s(\pi')$s and by $AvgBU$ ($StdBU$) the average (standard deviation) of $prop(\pi')$s. Finally, we estimate the proportion of *Baseline users* to be: $p_b(\pi) = \frac{s(\pi) - AvgSV}{StdSV} \cdot StdBU + AvgBU$.

The *Advice followers* follow the advised path even if they did not evaluate it as a good path, which allows us to assume that the proportion of *Advice followers* is constant across all advised paths. We extracted this proportion from Ψ^l, and in the following we denote it by p_f. The remaining proportion of users $1 - p_f - p_b(\pi)$ is assumed to be the *Advice ignorers*. This latter set of users deviates from the advised path so much that it is possible to assume that they would have selected the same path with or without any advice given.

4.2 Predicting Advice Deviations

Given the possible advice π, UMPA estimates the probability of a user taking a specific cut τ at a given vertex π^i. We denote this probability as $p(M, \pi, \pi^i, \tau)$ and use $p(\tau)$ when the other parameters are clear from the context. UMPA assumes that the function $p(\tau)$ is a linear combination of three cut features: *cut benefit*, *cut orientation* and *cut seemliness* (see e.g. [18]).

The **Cut Benefit** measures the relative reduction in steps between the cut and the original path segment. Formally, $\frac{l(\tau) - l(o(\tau))}{l(\tau)}$. For example, the cut shown in Figure 1 (marked with crossed nodes) has a positive benefit value since the length of the original path segment (between the first and last nodes of the cut) is greater than the length of the cut. The cut shown in Figure 2 has a benefit of 0 since the cut has the same length as the original path segment.

The **Cut Orientation** captures the tendency of human users to continue with a straight line motion. Its value depends on whether the cut or the original segment conformed to this tendency. The reference motion is the edge between the cut divergence node π^i and its predecessor in the advised path π^{i-1}. If the cut deviates from the advice by remaining in the same direction as the edge (π^{i-1}, π^i), we say that the cut has positive $+1$ orientation. If the original path segment (π^i, π^{i+1}) is similarly directed as (π^{i-1}, π^i), we say that the cut has negative -1 orientation. Otherwise, the cut's orientation is 0 (neutral). For example, in Figure 1 the value of the orientation of the cut marked by crossed nodes is 1, since the cut continues straight while the advised path turns left. The cut shown in Figure 2 however, has an orientation of -1 since the original path continues straight and the cut turns left.

The **Cut Seemliness** measures how seemly the cut is in the user's eyes. This value is calculated by subtracting the Seemliness-value of the original segment from the Seemliness-value of the cut. The seemliness of the cut shown in Figure 2 is positive since the first steps of the cut are in the same direction of the target, while the first steps in the original segment are in the opposite direction of the target.

Given that there is a very large number of cuts, it is almost impossible to collect enough examples in Ψ to learn the weights of $p(\tau)$'s features directly. Therefore, this estimation process was divided into two steps. First, UMPA estimates the probability, $r(M, \pi, \pi^i, \tau)$, that a cut τ will be taken by a user at vertex π^i, assuming that τ is the only possible cut at π^i. It was assumed that r is a linear combination of the three cut features described above, similar to $p(\tau)$. To compute the weights of $r(\tau)$'s features, UMPA created a training set of the form $(M', \pi, \pi^i, \tau, prop(\pi^i))$ where τ is a cut of π that starts at π^i and is the cut that was taken at π^i by the highest number of users according to Ψ. $prop(\pi^i)$ is the proportion of users that visited π^i and deviated there by taking any cut. Using these examples, the weights were estimated using linear regression.

Next, $r(\tau)$ is used to compute $p(\tau)$ after normalization. For any π^i, it was assumed (based on the way that $r(\tau)$ was learned) that the probability of the deviation at π^i across all cuts is equal to the highest $r(\tau)$ value of a cut starting at π^i. This probability is distributed across all possible cuts, starting at π^i, proportional to their $r(\tau)$ value.

4.3 Estimating the Cost of an Advised Path

Given a maze M and the possible advice π, UMPA estimates the expected cost that an agent may incur when presenting users with π. We denote this estimation by $ECost(\pi)$. This estimation is based on Ψ^l (the set of examples labeled with user types).

Notice that the contribution of the *Advice followers* is relatively easy to calculate. These are users that, independent of the maze or the particulates of the advised path π, always comply fully with π. Therefore, their contribution to $ECost(\pi)$ will always be $Cost_a(\pi)$ multiplied by the ratio of *Advice followers*.

The contribution of the *Advice ignorers* is calculated based on the data of users who received no advice. Let $\Omega_\emptyset = \{\tau | (M, \phi) \in \Psi\}$, i.e the set of paths in Ψ selected by users who did not receive any advice. We assume that the contribution of *Advice ignorers* to $ECost$ is the average agent cost on the paths in $\Omega_\emptyset$. Denote this value by $ECost_i$.

Calculating the contribution of the *Baseline users* to the agent's expected cost is more complex and is described hereunder. Having the estimated probability for each cut $p(\tau)$, an estimation for the agent's cost associated with *Baseline users* from advice π starting at π^i is denoted as $b(\pi, \pi^i)$. It can be calculated using the following recursive formulas:

$$b(\pi, \pi^{l(\pi)}) = 1$$

$$b(\pi, \pi^i) = \sum_{\tau, \tau^1 = \pi^i} p(\tau) \cdot (Cost_a(\tau) - 1) + b(\pi, \tau^{l(\tau)}) + \\ + (1 - \sum_{\tau, \tau^1 = \pi^i} p(\tau)) \cdot (b(\pi, \pi^{i+1}) + Cost_a(\pi^i \pi^{i+1}) - 1)$$

Note that the expression $Cost_a(\pi^i \pi^{i+1}) - 1$ is the agent's cost of traveling from π^i to π^{i+1}, which can either be 1 if no color switching occurs, or $W + 1$ if a color switching occurs. Now, using b, UMPA can estimate the contribution of the *Baseline users* to the agent's expected cost of an entire path π setting $ECost_b(\pi) = b(\pi, S)$.

An efficient algorithm for computing $ECost_b$ appears in the Appendix.

Given the users' proportions estimated in Section 4.1 and the utility contributions estimated above, we can compose the final *heuristic* estimate of the advised path cost $ECost(\pi)$, which is the expected agent's cost across all human generated path solutions in response to π:

$$\begin{aligned} ECost(\pi) \quad = \quad & p_f \cdot Cost_a(\pi) + (1 - p_f - p_b(\pi)) \cdot ECost_i + \\ & p_b(\pi) \cdot ECost_b(\pi) \end{aligned}$$

4.4 Searching for Good Advice

We view mazes as graphs, and use the A^* search algorithm to find a path π from the start node S to the target node T with the minimal expected cost. We do not use node recognition, therefore during the A^* search the agent generates a search tree in which each node is associated with a vertex of the grid. When given a node n_v in the tree that is associated with the vertex v, there is a unique path in the tree from the root node of the tree to n_v that is associated with a path on the grid from S to v. We denote this path as θ. The cost function for node n_v is then $ECost(\theta)$. The agent uses the minimal agent cost of traveling between v to T as the heuristic function of n_v in the tree. The minimal agent cost to travel from each vertex to T can be efficiently calculated for all vertexes using the Dijkstra's algorithm starting at T.

To limit the manipulation effect of UMPA, the search only considers paths with cuts where the agent does not gain from the user's taking them. That is, the agent prefers the user to take the advised path and does not benefit from his deviation. Formally, UMPA only considers paths such that, for any suffix $\sigma = \pi^i \cdots \pi^{l(\pi)}, i \geq 1$, $ECost(\sigma) \geq Cost_a(\sigma)$ holds. If A^* stops with a path that does not satisfy the condition above it will be rejected, and A^* will be forced to continue the search.

5. EXPERIMENTAL EVALUATION

We have developed an online system that allows people to solve path selection problems in a maze. It can be accessed via `http://cupkey.com/selfmazeplayer.swf`. The maze design was chosen to remove all effects of familiarity with the navigation network from the experiments. We also designed the mazes so that they do not directly resemble a street map to avoid the effects of navigating a city. Furthermore, every human subject was presented with a single instance of the problem in order to exclude effects of learning or trust. We ran two kinds of experiments. First, the experiments were aimed at collecting data on users' behaviors when facing advice that either benefited the users or the system utilities regarding route selection. Second, after the UMPA approach was applied using the collected data, we ran experiments to validate our hypothesis regarding users' behavior change as a result of providing them with advice adapted to the users' behavior learned in the first experiments. Furthermore, the main goal has been to test the hypothesis that UMPA outperformed all other advice generator methods we considered.

Participation in our study consisted of 701 subjects from the USA: 383 females and 298 males. The subjects' ages ranged from 18 to 72, with a mean of 37.

5.1 Methodology

5.1.1 Running Experiments on Amazon Turk

All of our experiments were run using Amazon Mechanical Turk (AT) [1], a crowd sourcing web service that coordinates the supply and demand of tasks that require human intelligence to complete. Amazon Mechanical Turk has become an important tool for running experiments with human subjects and was established as a viable method for data collection [22]. We took several actions to encourage subjects to truly attempt to find the shortest path: we only selected workers with a good reputation; a set of questions, designed to verify understanding of the task, was presented to the subjects prior to the task execution; and as a stimulus, all subjects were guaranteed a monetary bonus inversely proportionate to the length of the path they have selected. Our previous experience in running experiments on Amazon Turk demonstrated that almost all subjects have considered our tasks seriously. We asked a group of university students and Amazon Turk workers to perform the same task and found that the average score of the Amazon Turk workers was higher than that of the students. Thus, our own experience confirms other studies [22] about the viability of this medium for empirical research.

5.1.2 Experimental Setup

Each experiment consisted of a colored-maze panel similar to the one depicted in Figure 1. A single panel was shown to each participant. The user's task was to select the shortest path through the maze that connected the source and target nodes. When subjects were presented with advice from the system, they were informed that this advice was calculated to reduce the path length *and* also the number of color switches. Still, users were not given the specific switching cost W between colors. We used four distinct mazes, all of size 80×40. These mazes were complex enough so that users would find it difficult to compute the shortest path in the limited time allotted for the task. In all of the experiments, we set the weight W for color switching to 15.

We ran four training sessions to learn user behaviors from three mazes. Then we ran our UMPA algorithm on the fourth maze to compute the advice, using information about this maze and the parameters learned from the other three mazes (we did this for each one of the four mazes). That is, UMPA's results are averaged over four different mazes and training and testing data were strictly separated.

Finally, we presented the subjects with post-task questions that were designed to assess the general attitude towards computer advice and the subjective evaluation of the advised path quality.

5.1.3 Basic Algorithms

We compared the performance of our UMPA algorithm to the following three cases:

- *No advice (silent)* – no advice is presented on the maze panel,
- *Shortest path* – the advice presented corresponds to the shortest path from source to target,
- *Greedy* – the advice that the user gets is the path computed to minimize the agent's cost of traversing it, $Cost_a$.

The *Shortest* solution is the one that minimizes the cost of the user and, therefore, we expect that its acceptance by the users will be high. Moreover, the number of *advice ignorers* will be small and the probability of deviation will be small as well. However, since the agent cost of this path is usually high we expect the presenting *Shortest* will yield the agent relatively high expected cost. When providing *Greedy* advice, we run the risk that most of the users will ignore it, while the ones that will accept it will yield the highest benefits to the agent. We first compared the agent's average cost when providing any one of these three types of advice. (This comparison was performed using ANOVA, a method of analysis used to determine the level of statistical significance when dealing with more than two groups). Then we chose the one that was best for the agent and compared the UMPA solution to this *baseline algorithm*. Then, we considered the following questions:

- What is the effectiveness of UMPA estimation methods?
- Is UMPA significantly better than the basic algorithm?
- If UMPA is better than this basic algorithm from the agent's point of view, will it, in return, decrease the users' benefits and satisfaction? Or does UMPA yield mutually beneficial results compared to this basic algorithm?

5.2 Basic Results

We calculated the effects that Silent, Shortest and Greedy types of advice have on the average agent cost across paths selected by users in our experiments. The corresponding three bar charts on the left of Figure 3 summarize the results (the lower the better). The average costs over four mazes of types Silent, Shortest and Greedy were 559.73, 559.55 and 501.68, respectively. That is, the paths chosen by users after receiving *Greedy* advice have resulted in a significantly (with p-value $p < 0.001$) lower cost to the agent than the cost attained when the other two types of advice were present (Shortest and Silent).

We have also studied the statistics of the advice effect on the user's costs (see three left-most bar charts of Figure 4). As expected, the cost of the paths chosen by users was significantly lower (130.85) when *Shortest* advice was provided, than when the other two types of advice were present (Greedy (144.6) and Silent (142.75)). Moreover, we wanted to check whether giving advice that results in lowest costs to the agent can also decrease the costs to the users compared to the case when no advice is provided. The results were mixed and no significant difference was found between Greedy and Silent. That is, while *Greedy* advice significantly decreased the agent's cost, it did not significantly increase the user's costs. We concluded that the UMPA advice generation algorithm should be compared to the case when *Greedy* advice is provided.

5.3 UMPA Advice Algorithm Performance

We set the UMPA parameters as follows: the length of a cut L_1 was bound to 40; a cut's potential increase in length L_2 to 20% of the corresponding original segment and the discount factor δ in the cut-seemliness feature calculation was set to 0.95.

The first step in the evaluation of our UMPA algorithm was to verify the effectiveness in computing $p(M, \pi, \pi^i, \tau)$ (i.e., the *predicted* number of users that will take cut τ when

Figure 3: Average agent's costs

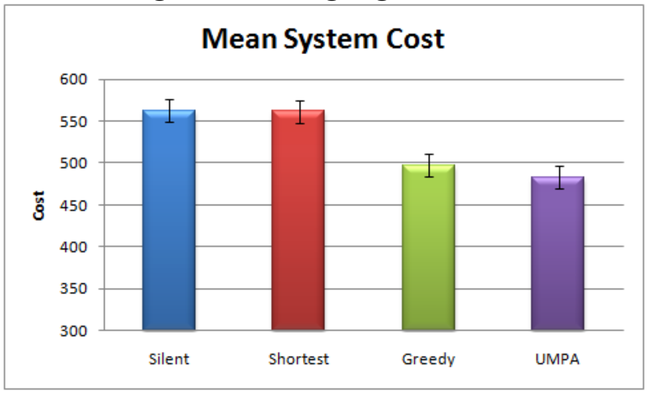
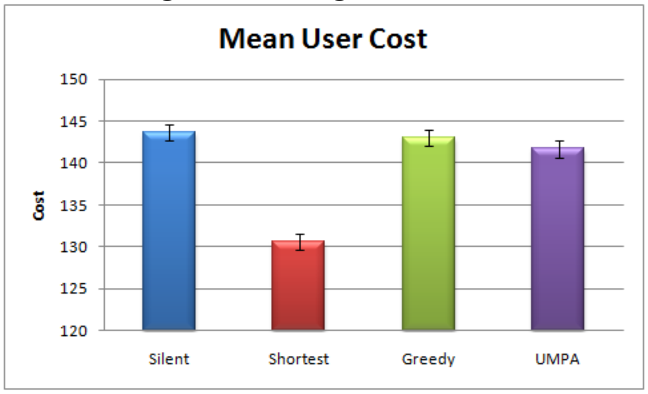


Figure 4: Average users' costs

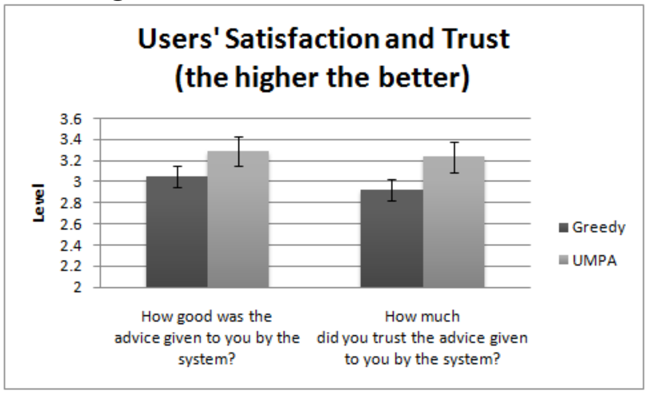


facing divergence node i, when advice π was provided in maze M). We found a high correlation (0.77) between this prediction and the actual fraction of users who took it when reaching the cut's divergence node. High correlation (0.7) was also found between the actual fraction of users that took advice π or manipulated it, *the Baseline users* and our *predicted* number of such users, $p_b(\pi)$. Finally, we obtained a high correlation (0.76) between the estimated value of advice π, $ECost_a(\pi)$ and the empirical average value of the actual paths selected in response to advice π. This is significant since the correlation between the agent's cost of π itself and the empirical average of the selected path was only 0.06.

We then compared the average cost attained by the agents when users chose paths after getting either the UMPA-based advice or *Greedy* advice. Consider the two corresponding bar charts on the right side of Figure 3(the lower the better). UMPA's average costs over the four mazes was 484.95 compared with the Greedy advice that was 501.68. That is, on average, the UMPA approach outperformed *Greedy* advice, resulting in significantly lower costs ($p < 0.05$) for the agent.

We also compared the average cost incurred by the paths chosen by users to the users themselves when receiving the advice provided by the UMPA algorithm and *Greedy* advice (see the two right-most bar charts of Figure 4). To our surprise, the average results attained by the users that were given the UMPA advice (142.33) were significantly better

Figure 5: Users' satisfaction and trust



(lower cost) than those attained by users that were presented with *Greedy* advice (142.33) ($p < 0.05$). In summary, when comparing the results obtained by running two advice generation techniques (one provides UMPA advice and the other advices *Greedy* advice), we conclude that UMPA-based advice outperforms *Greedy* advice. That is, the average cost incurred by the agent when users selected their paths and the average cost incurred by the human users would decrease significantly when the users were provided with UMPA advice. So UMPA manipulative advice is indeed *mutually* beneficial when compared with *Greedy* advice.

Finally, we considered the subjective view of the users on the paths that were advised. Users were presented with the following questions after they finished the route selection task: (i) "How good was the advice given to you by the system?" and (ii) "How much did you trust the advice given to you by the system?" The possible answers were on a scale of 1-5, where 5 indicates highest satisfaction and 1 the lowest satisfaction. The results are presented in Figure 5. Regarding the first question, UMPA advice was considered significantly better than *Greedy* advice, with $p < 0.05$. The average rating for UMPA was 3.29 and the average rating for *Greedy* was only 3.05. Similarly, with respect to trust, the average rating of UMPA was 3.23 whereas the average rating of *Greedy* was only 2.92, i.e., users trusted UMPA advice significantly more than *Greedy* advice ($p < 0.05$).

6. CONCLUSIONS AND FUTURE WORK

This paper presents an innovative computational model for advice generation in human-computer settings where agents are essentially self-interested but share some common goals with the human. To assess the potential effectiveness of our approach, we performed an extensive set of path selection experiments in mazes. Results showed that the agent was able to outperform alternative methods that solely considered the agent's or the person's benefit, or did not provide any advice. In future work, we will extend this approach to settings in which people and computers interact repeatedly, requiring the agent to reason about the effects of its current advice on people's future behavior.

7. REFERENCES

[1] Amazon. Mechanical Turk services. http://www.mturk.com/, 2010.

[2] D. Antos and A. Pfeffer. Using reasoning patterns to help humans solve complex games. In *IJCAI*, pages 33–39, 2009.

[3] A. Azaria, Z. Rabinovich, S. Kraus, and C. Goldman. Strategic information disclosure to people with multiple alternatives. In *Proc. of AAAI*, 2011.

[4] A. Blume, D. V. DeJong, Y.-G. Kim, and G. B. Sprinkle. Evolution of communication with partial common interest. *Games and Economic Behavior*, 37(1):79 – 120, 2001.

[5] S. Bonaccio and R. S. Dalal. Advice taking and decision-making: An integrative literature review and implications for the organizational sciences. *Organizational Behavior and Human Decision Processes*, 101(2):127–151, 2006.

[6] H. Cai and J. T.-Y. Wang. Overcommunication in strategic information transmission games. *Games and Economic Behavior*, Vol. 56, Issue 1:7–36, July 2006.

[7] C. F. Camerer. *Behavioral Game Theory. Experiments in Strategic Interaction*, chapter 2. Princeton University Press, 2003.

[8] Y. Chen. Perturbed communication games with honest senders and naive receivers. *Journal of Economic Theory*, 146(2):401 – 424, 2011.

[9] C. G. Chorus, E. J. Molin, and B. van Wee. Travel information as an instrument to change car drivers travel choices. *EJTIR*, 6(4):335–364, 2006.

[10] V. Crawford. A survey of experiments on communication via cheap talk. *Journal of Economic Theory*, 78:286–298, 1998.

[11] J. W. Dickhaut, K. A. McCabe, and A. Mukherji. An experimental study of strategic information transmission. *Economic Theory*, 6(3):389–403, November 1995.

[12] M. Duckham and L. Kulik. "Simplest" paths: Automated route selection for navigation. In *LNCS*, volume 2825, pages 169–185, 2003.

[13] R. Forsythe, R. Lundholm, and T. Rietz. Cheap talk, fraud, and adverse selection in financial markets: Some experimental evidence. *The Review of Financial Studies*, Vol. 12, No, 3:481–518, Fall 1999.

[14] J. Glazer and A. Rubinstein. On optimal rules of persuasion. *Econometrica*, 72(6):1715–1736, 2004.

[15] J. Glazer and A. Rubinstein. A study in the pragmatics of persuasion: A game theoretical approach. *Theoretical Economics*, 1:395–410, 2006.

[16] R. J. Hanowski, S. C. Kantowitz, and B. H. Kantowitz. Driver acceptance of unreliable route guidance information. In *Proc. of the Human Factors and Ergonomics Society*, pages 1062–1066, 1994.

[17] M. Hipp, F. Schaub, F. Kargl, and M. Weber. Interaction weaknesses of personal navigation devices. In *Proc of AutomotiveUI*, 2010.

[18] H. H. Hochmair and V. Karlsson. Investigation of preference between the least-angle strategy and the initial segment strategy for route selection in unknown environments. In *Spatial Cognition IV*, volume 3343 of *LNCS*, pages 79–97, 2005.

[19] P. Hoz-Weiss, S. Kraus, J. Wilkenfeld, D. R. Andersend, and A. Pate. Resolving crises through automated bilateral negotiations. *Artificial Intelligence journal*, 172(1):1–18, 2008.

[20] S. K. Hui, P. S. Fader, and E. T. Bradlow. Path data in marketing. *Marketing Science*, 28(2):320–335, 2009.

[21] X. J. Kuang, R. A. Weber, and J. Dana. How effective is advice from interested parties? *J. of Economic Behavior and Organization*, 62(4):591–604, 2007.

[22] G. Paolacci, J. Chandler, and P. G. Ipeirotis. Running experiments on Amazon Mechanical Turk. *Judgment and Decision Making*, 5(5), 2010.

[23] K. Park, M. Bell, I. Kaparias, and K. Bogenberger. Learning user preferences of route choice behaviour for adaptive route guidance. *IET Intelligent Transport Systems*, 1(2):159–166, 2007.

[24] N. Peled, Y. Gal, and S. Kraus. A study of computational and human strategies in revelation games. In *Proc. of AAMAS*, 2011.

[25] H. Rachlin and L. Green. Commitment, choice and self-control. *J. Exp. Anal. Behav.*, 17:15–22, 1972.

[26] L. Rayo and I. Segal. Optimal information disclosure. *Journal of Political Economy*, 118(5):949–987, 2010.

[27] K.-F. Richter and M. Duckham. Simplest instructions: Finding easy-to-describe routes for navigation. In *Proc. of the Geographic Information Sience*, volume 5266 of *LNCS*, pages 274–289, 2008.

[28] S. Sanchez-Pages and M. Vorsatz. An experimental study of truth-telling in a sender-receiver game. *Games and Economic Behavior*, 61(1):86–112, 2007.

[29] I. Sher. Credibility and determinism in a game of persuasion. *Games and Economic Behavior*, 71(2):409 – 419, 2011.

[30] J. Sobel. Giving and receiving advice. Working Paper, August 2010.

[31] A. Tversky and D. Kahneman. Loss Aversion in Riskless Choice: A Reference-Dependent Model. *The Quarterly J. of Economics*, 106(4):1039–1061, 1991.

[32] I. Yaniv and E. Kleinberger. Advice taking in decision making: Egocentric discounting and reputation formation. *Organizational Behavior and Human Decision Processes*, 83(2):260–281, 2000.

APPENDIX

```
Input: A maze, with an advised path $\pi$.
Output: $ECost_b(\pi)$ – estimated cost contributed by Baseline users
1: $ECost_b \leftarrow Cost_a(\pi)$.
2: $vec \in \mathbf{R}^{l(\pi)} \leftarrow \vec{0}$. $vec(0) = 1$.
3: for each $i < l(\pi)$ do
4:     for each cut $\tau$ s.t. $\tau^1 = \pi^i$ do
5:         {Predict the fraction of baseline users who take the cut}
           $a(\tau) \leftarrow (1 + \sum_{j<i} vec[j]) \cdot p(\tau)$
6:         $ECost_b \leftarrow ECost_b + (Cost_a(\tau) - Cost_a(o(\tau))) \cdot a(\tau)$.
7:         {Update mass at cut entry point.}
           $vec[i] \leftarrow vec[i] - a(\tau)$
8:         {Update the cut exit point}
           $vec[j|\pi^j = \tau^{l(\tau)}] \leftarrow vec[j] + a(\tau)$
9: return $ECost_b$.
```

Intuitively, the algorithm's basic assumption is that the set of users forms a continuous unit mass. The algorithm then traces the flow of this unit of mass along different cuts that diverge (or converge) at vertices along the advised path.

The complexity of this algorithm that saved $\sum_{j<i} vec[j]$ in each iteration is $O(\#cuts + l(\pi))$.