AAMAS 2013

Saint Paul
Minnesota
USA

6th - 10th May 2013

twelfth international conference on autonomous agents and multiagent systems

W13 - The 2nd International Workshop on Human-Agent Interaction Design and Models (HAIDM 2013)

Sarvapali D. Ramchurn, (University of Southampton)
Joel E. Fischer, (University of Nottingham)
Rajiv Maheswaran, (University of Southern California)
Rong Yang, (University of Southern California)
Long Tran-Thanh, (University of Southampton)

May 7, 2013

Program Committee

Bo An	Chinese Academy of Sciences
Ben Bedwell	University of Nottingham
Enrico Costanza	University of Southampton
Joel Fischer	University of Nottingham
Yakov Gal	MIT
Greg Hines	University of Southampton
Christopher Kiekintveld	University of Texas at El Paso
Rajiv Maheswaran	University of Southern California
James Pita	University of Southern California
Sarvapali Ramchurn	University of Southampton
Alex Rogers	University of Southampton
Avi Rosenfeld	Jerusalem College of Technology (JCT)
Paul Scerri	Carnegie Mellon University
Sebastian Stein	University of Southampton
Long Tran-Thanh	University of Southampton
Matteo Venanzi	University of Southampton
Rong Yang	University of Southern California

Preface

As the boundaries of autonomous agents and multi-agent systems continue to expand, there is an increasing need for agents to interact with humans. In fact, the field of multi-agent systems has matured from conceptual models to applications within the real-world (e.g., Energy and sustainability, disaster management, or health care). One significant challenge that arises when transitioning these conceptual models to applications is addressing the inevitable human interaction. To this end, the workshop on Human-Agent Interaction Design and Models (HAIDM) examines major challenges at the intersection of human-agent systems.

Following the success of workshop at AAMAS 2012 in Valencia, this second edition of the workshop was setup with the aim of strengthening the relationship between the HCI and Agents communities. By so doing, we have attracted a range of submissions from a different communities and, following an intense round of reviews by a strong Programme Committee (who we are ever thankful to) consisting of researchers with backgrounds in human-computer interaction, game theory, and agent-based applications, we are extremely pleased to be able to provide here a collection of the best work and work-in-progress in this exciting research area.

The contributions focus on strategic human-agent interaction settings where agents aim to outwit or work better with humans, other contributions focus on how agent technology can help improve recommendations to humans in health and energy applications. Indeed, it is our hope that more of the research in this community will be driven by key requirements drawn from real-world applications where HCI, Behavioural Economics, and Agents communities can find a common ground.

Finally, we look forward to get you involved in the discussions at the workshop (and online at http://haidm.wordpress.com)!

May 7th 2013, Minnesota

Sarvapali D. Ramchurn, Rajiv Maheswaran,
Long Tran-Thanh, Joel Fischer
& Rong Yang

Table of Contents

On Automated Agents' Rationality

Amos Azaria[1], Ariella Richardson[2], Avshalom Elmalech,
Avi Rosenfeld[2], Sarit Kraus[1,3], and David Sarne[1]

[1] Department of Computer Science, Bar-Ilan University, Ramat Gan 52900, Israel
[2] Dept. of Industrial Engineering Jerusalem College of Technology, Israel
[3] Institute for Advanced Computer Studies University of Maryland, MD 20742

Abstract. Agents that interact with humans are known to benefit from integrating behavioral science and exploiting the fact that humans are irrational. Therefore, when designing agents for interacting with automated agents, it is crucial to know whether the other agents are acting irrationally and if so to what extent. However, little is known about whether irrationality is found in automated agent design. Do automated agents suffer from irrationality? If so, is it similar in nature and extent to human irrationality? How do agents act in domains where human irrationality is motivated by emotion? This is the first time that extensive experimental evaluation was performed in order to resolve these questions. We evaluated agent rationality (for non-expert agents) in several environments and compared agent actions to human actions. We found that automated agents suffer from the same irrationality that humans display, although to a lesser degree.

1 Introduction

Automated Agents are integrated into countless environments, such as Electronic commerce, Web crawlers, Military agents, Space Exploration probes and Automated drivers. Due to the high importance of automated multi-agent environments, many competitions were established where automated agents compete with each other in order to achieve a goal [35,2,1,36].

Modeling agents is beneficial for agent-agent interaction [29]. However, building such a model is a complex issue, and furthermore, if the model built is too far from the actual opponent's behavior, using it may become detrimental [25,22]. How should designers plan their agents when opponent modeling is unavailable? Can any general assumptions be made on automated agents and used for agent design?

Research into peoples' behavior has found that people often do not make strictly rational decisions but instead use sub-optimal, bounded policies. This behavior has been attributed to a variety of reasons including: a lack of knowledge of one's own preferences, the effects of the task complexity, framing effects, the interplay between emotion and cognition, the problem of self-control, the value of anticipation, future discounting, anchoring and many other effects [37,23,5,11]. Since people do not usually use fully rational strategies themselves, agents based on game theory approach, which assume rational behavior in humans often perform poorly [28,20,8]. Many studies have shown that psychological factors and human decision-making theory are needed in order to develop a good model of true human behavior, which in turn is required for optimizing the performance of agents interacting with humans [16,19,26,30,28,7,9].

In domains where opponent modeling is infeasible, can we use rationality models or models of human behavior in order to design agents that interact with other agents (for non-expert-designed agents)? In order to do this we need to determine whether we can assign rationality traits to the agents: Are automated agents strictly rational, and if not how do they compare to humans? The answer to this question will provide useful guidelines for agent strategy planning when opponent agent modeling is infeasible, especially for domains where human rationality has been studied.

Throughout this paper, a rational player will refer to a player who tries to maximize his expected outcome and assumes that all other players are doing so as well. Irrational behavior will be indicated by a player who fails to compute a rational strategy, has his own subjective utility function which differs from the expected outcome or believes (possibly justifiably) that other players use irrational behavior. Note that, in cases in which opponents act irrationally, a player acting irrationally as well, may gain a greater expected outcome than a fully rational player.

We perform an extensive experiment evaluation of automated agents' behavior and compare it to fully rational and human behavior in three different games. Each game will allow the exploration of different aspects of agent rationality. In the first game we examine whether agents exhibit an irrational tendency to keep all options available (as humans are known to do). For the second game we use a costly exploration game. The second game expands our study as it enables the exhibition of behaviors that lie on either side of the equilibrium. In a case where agents are not purely rational, as we hypothesize, it remains to be seen whether they display the same type of irrational behavior as humans or sway in the opposite direction. The third game integrates activities which trigger emotions. We show that as one would expect humans become emotionally involved when playing this game. It remains to be seen how agents react. Do agents use strategies that mimic human emotions? Will they use these strategies if it involves a clear utility loss?

In this study we will determine whether automated agents are rational or not and to what extent, and whether or not they display behavior that is similar to humans. The findings of this research have practical implications for designing agents that interact with non-expert agents.

2 Related Work

Opponent modeling is of great importance when designing automated agents which interact with other agents [29]. Carmel and Markovitch [13] show that if an agent can build a model of its opposing agent, this model could be used to improve its own performance. McCracken and Bowling [25] propose a method for modeling an opponent in the Rock-Paper-Scissors domain. Lazaric et al. [22] propose a method for opponent modeling in Kuhn Poker (a degenerated version of poker). However, in more sophisticated games these methods become intractable.

Angluin [4] proposes an algorithm for modeling an automated agent as a Moore machine. Angluin assumes the existence of a teacher which replies 'Yes' for a correct conjecture or provides a counterexample on which the model and the machine disagree. This algorithm is polynomial in the number of states in the machine. However, the existence of such a teacher isn't likely when modeling an opponent. Carmel and Markovitch [13] relax the teacher assumption and provide a heuristic algorithm for learning Moore machines. The algorithm builds a model consistent with past examples, and when a new counterexample arrives it tries to extend the model in a minimal fashion. However, in practice an automated opponent is not likely to be limited to being a Moore machine but rather a more general Turing machine.

Various studies [10] have compared experimental results between the strategy method (in which a responder makes conditional decisions for each possible information set) to the more standard direct-response method (in which the responder observes the action of the first mover and then chooses a response). There is mixed evidence as to whether the two methods lead to similar results. While in the strategy method the subjects make decisions for all the possible options, in our work the agents must provide a general strategy since the number of possible options is extremely large. Our work should not be confused with research on the psychology of programming [31] that investigates how understanding psychological aspects of programming improves their ability to write error-less and readable code.

Human behavior is well studied and plays an important role in human-agent interaction. However, the evaluation of automated agent rationality is much less explored. Grosz et al. [18] introduced the Colored Trails game in order to investigate decision-making strategies in multi-agent situations. They showed that human players and agents do not play in the same way. Their results indicate that people design automated agents that don't play as well as human players, possibly because they cooperate less than people. It is interesting to note that the agents also do not adhere to the equilibrium strategy. Although the environment used in this game was a mixed human-agent environment, while we are studying environments composed only of agents, this research provides some insight into our problem.

Manistersky et al. [24] explored the evolution of automated agents in a negotiation environment. Designers were given a chance to improve the agents based on previous performance. They found that agents designed by humans seem to perform better than equilibrium agents but not as well as Pareto-optimal agents. The environment explored is a pure automated agent environment, such as the one we use. However, they did not compare the performance of agents to that of humans.

Unlike the above-mentioned prior work, the games we selected are purposely very simple compared to the Colored Trails and negotiation games. Neither do we aim to evaluate the performance of the agents described in this paper. Rather we are interested in studying the degree of rationality displayed by the agents relative to humans.

It seems that automated agents are not purely rational and are different from humans. We would like to know, however, how they relate to humans. Do the agents act in a unique fashion? Or are they displaying the same irrational behavior that humans use, but possibly to a lesser degree?

3 Door Game

The first experiment tests to what extent agents and people exhibit a tendency to keep all options viable, even when the cost of doing so is greater than the potential benefit.

This irrational tendency among humans has been demonstrated by Shin and Ariely [34] via the "Door Game".

In the basic version of the game, the player is faced with three doors (alternatives), each associated with a different distribution of payoffs. The distribution of each door is a priori unknown to the player. The player first chooses with which door to begin, and from that point on, any additional click on that door will yield a reward drawn from that distribution. At any time, the player can switch to any other door by clicking on it. Switching to another door enables the player to receive rewards from the distribution characterizing that door via additional clicks on it. The player can collect rewards only from the door to which she has most recently switched. Rewards are accumulated and the player's goal is to maximize gains given a limited "budget" of clicks. Once the player has used all of her clicks, the game terminates and she is paid the sum of her door-click pay-offs. The click that the player needs to "sacrifice" in order to switch doors is in fact a switching cost. This setting can be mapped to a Mutli-Armed Bandit problem [6], in which a rational strategy focuses most of the exploration in initial rounds and then sticks with the door with the highest average outcome.

Shin and Ariely also considered a variant in which each time a participant clicks on a door, the two other doors are reduced in size by $\frac{1}{15}$ of their original width. A single click on a shrinking door restores it to its original size and the process continues. Once a door shrinks to zero, it is eliminated for the remainder of the game. In the basic version Shin and Ariely show that human subjects followed the rational strategy, however, with the shrinking door variant, players tend to switch from door to door, in an effort to keep their options open. This resulted in a substantial performance degradation (in terms of the rewards accumulated) compared to choosing any single door and sticking with it.

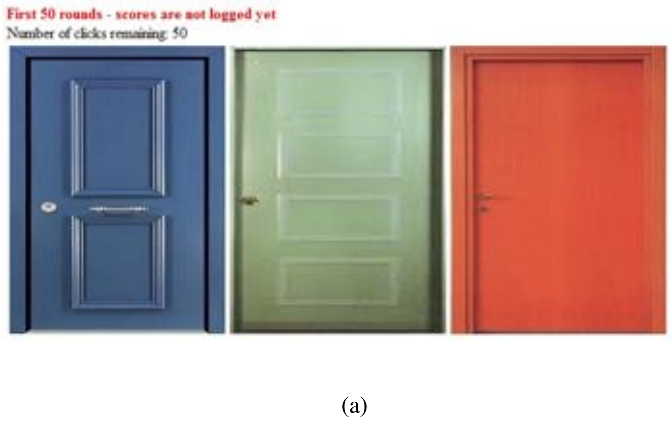
Our experiment with the door game used the game variant with the diminishing doors. We followed a specific experimental design reported in [34] where the game is made up of two phases, each with a "budget" of 50 clicks. In the first phase (the exploration phase), the participants do not receive any payoff and were only notified of the payoff amount. The purpose of this phase is for the participants to identify the best door. This phase is long enough for a rational player to select a single door from which she does not need to divert for the entire second phase (while ignoring the vanishing of the other doors). In the second stage (the exploitation phase), the participants received the payoff obtained from the door on which they clicked.

The experiment included 48 computer science students, each of whom programmed an agent capable of playing this game, as part of the students' regular course assignment. Their grades were in part calculated based on the performance of their specific agent. The students received a thorough explanation which included the game rules, the payoff accumulation method and the division into the two stages.

Human subjects were recruited using Amazon's Mechanical Turk service [3], which is a crowd sourcing web service that coordinates the supply and demand of tasks which require human intelligence in order to be completed. Amazon Mechanical Turk has become an important tool for running experiments with human subjects and has been established as a viable method for data collection [27].

The subjects first received appropriate instructions. In order to assure that they fully understood the game, the subjects had to first play the game when no score was recorded, and only then they could play the actual game. In order to encourage them to play seriously they received a bonus which was proportionate to their performance.

The GUI presented the doors to the human subjects, with each door size changing according to the clicks made. Figure 1(a) demonstrates a screen-shot of a game where the user has not yet clicked on a door. Figure 1(b) shows a screen-shot where the user has picked the middle door and the other two doors have started shrinking.


First 50 rounds - scores are not logged yet
Number of clicks remaining: 50
(a)

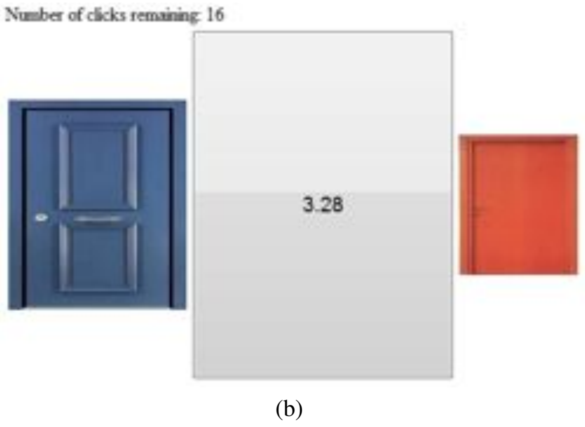


Fig. 1. A screen-shot of the door game user interface: (a) before user clicks on door (b) user clicked on middle door

As discussed above, rational behavior in this game would be to dedicate all clicks in the first phase to exploration and stick to a single door throughout the second phase. Therefore, any switching between doors during the second phase can be classified as irrational behavior. In our experiment, 77% of the human subjects switched doors during the second phase of the game, comparable to only 29% of the agents programmed for this task. While the human subjects switched doors 14.80 times on average, in the last 50 clicks, the automated agents did so only 3.02 times on average. All results reported throughout all experimental sections are statistically significant with $\alpha = 0.05$ (using either student t-test or Fisher's exact test). These results indicate that, just as humans, also automated agents act irrationally to preserver options, even at a great expense. However, the extent of this irrational behavior is much less for automated agents. This provides the first answer to the questions we raised.

In the door game, irrational behavior could only be displayed in a single direction, as rational behavior required not switching doors at all. We next investigate what happens when rational (or optimal) behavior has a value which can be exceeded in both directions. The irrational player may adhere to a value which is either greater or less than the rational one. Assuming that the agents will act irrationally again, will they display the same type of behavior as humans or will they sway in the opposite direction?

4 Search Game

The second irrational behavior we inspect is the property observed in human consumers who tend to perform too short a search prior to purchasing a product [15]. This domain is particularly interesting (assuming the agents are irrational) as we can observe whether they display the human tendency to perform a short search, or if they exhibit a different type of irrational behavior and perform too long a search.

The sequential exploration problem considers an individual facing a number of possible available opportunities (e.g., to buy a product) out of which she can choose only one. The value of each opportunity to the agent is a priori unknown. Instead, only its probability distribution function is known, and revealing the true value of an opportunity is costly. The problem, to which a broad class of real-world situations can be mapped, was formally introduced by Weitzman [38] along with its optimal (cost-minimizing or utility-maximizing) strategy. The optimal strategy derives from the trade-off between the benefit from the potential improvement in the quality of the results which the agent may further obtain with the additional exploration and the costs of carrying out such exploration. The optimal strategy is based on setting a reservation value (a threshold) for each opportunity and choosing to obtain the value of the opportunity associated with the minimum reservation value at any time (terminating the exploration once the minimum value obtained so far is less than the minimum reservation value of any of the remaining opportunities). Intuitively, the reservation value of an opportunity is the value where the agent is precisely indifferent: the expected marginal benefit from obtaining the value of the opportunity exactly equals the cost of obtaining that additional value.

Much evidence has been given in the literature to differences between the behaviors exhibited by people and the theoretic-optimal strategy in costly exploration settings.The major difference reported relates to the tendency of people to terminate their search before the theoretic-optimal strategy would have done so [32,14,15].

In order to evaluate the existence and extent of this phenomena in automated agents, we used 31 agents whose strategies were programmed by computer science students (each agent by a different student) as part of their regular course assignments. (The group of students who designed these agents is different from the group that designed the agents for the door game.) Each agent receives as input a list of opportunities, their distribution of values and the cost of obtaining these values. The agent had to decide at any time the value of which opportunity to obtain next (incurring the appropriate cost) and when to terminate the exploration. Each student's grade in the assignment was correlated with her agent's performance. As part of their assignment, students provided documentation that described the algorithm used for managing the exploration. To compare the results with people, we used a GUI-based experimental infrastructure, simulating a price-search environment. Each opportunity represented a store associated with a different distribution of prices and a cost for obtaining the true price (represented as a "parking cost" for parking next to that store). Figure 2 presents a screen-shot of this GUI. Querying a store is done by clicking the "Check" button below it, in which case the true price of the store becomes known and the parking cost of that store is added to the accumulated cost. The game terminates when clicking the "Buy" button (available only in stores whose prices are known).

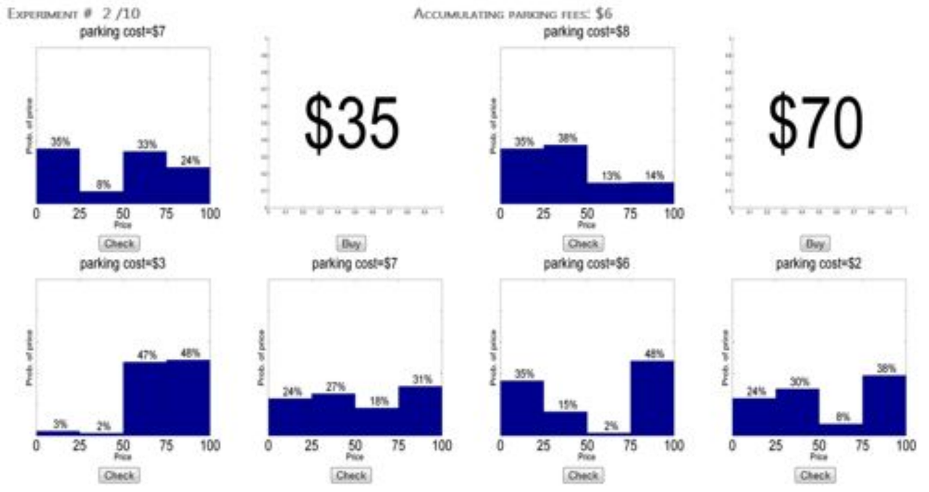


Fig. 2. A screen-shot of the search game user interface

Human subjects were recruited using Amazon's Mechanical Turk service. Overall 150 human subjects participated in this experiment. Subjects received a short textual description of the experiment, emphasizing the exploration aspects of the process and the way costs are accumulated, followed by a short video clip. Then, a series of practice games were played in order to make sure that the subject understood the experiment. Participants had to play at least three practice games; however, they could continue practicing until they felt ready to start the experiment.

We used 300 randomly generated exploration problems. Each problem contained 8 opportunities associated with distributions of values and costs. Distributions were formed as multi-rectangular distribution functions (i.e., based on rectangles of equal width, however each with a different distribution mass) defined over the continuous interval $0 - 100$. Costs were uniformly drawn from the interval $1 - 10$. Each agent was tested using all 300 problems, and each human subject was randomly assigned 10 problems out of this set.

Figure 3 depicts the average search extent in our experiments (measured as the number of opportunities explored within each game) for people, agents and the optimal exploration strategy (EQU). As observed from the figure, the tendency to terminate exploration earlier than is optimal, is reflected both in people's and agents' exploration patterns. This tendency, however, is observed to a greater extent with people, whereas with agents it is somehow less so. Thus we have shown that not only do agents display irrational behavior and on the other hand are more rational than humans, but we also show that the agents veer in the same irrational direction as the humans, as they stop their search before reaching the optimal point, just as humans do (rather than searching for longer than optimal).

In both of the games we have described so far one could maybe claim that the player did not fully understand what the rational behavior was, and therefore did not use it. The third game investigates a setting where we know that the human players understand what the rational behavior is, yet they still choose not to use it. In humans this would be explained as emotional involvement on the part of the player. Will agents still display irrational behavior when the designers are known to understand the rational strategy?

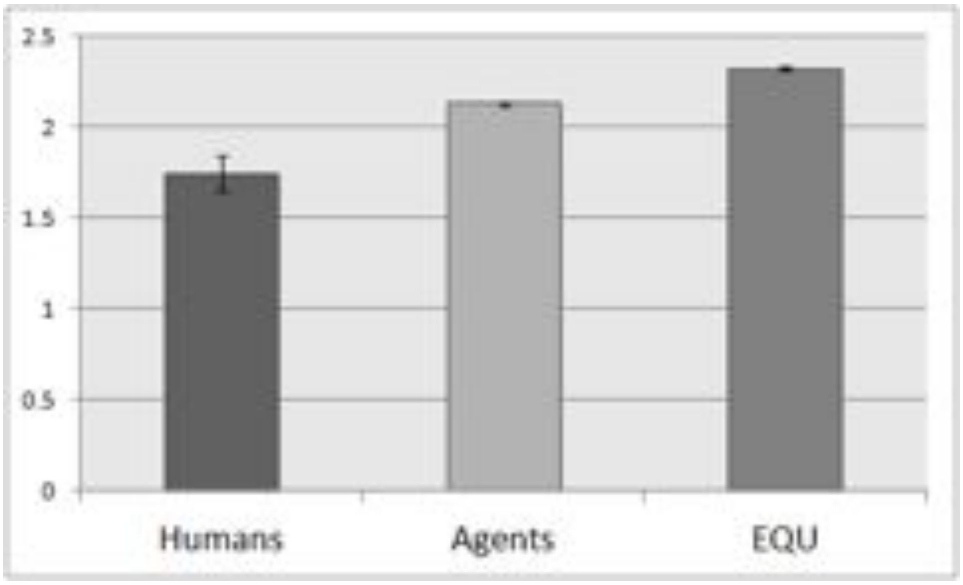


Fig. 3. People, agents and optimal agent search extent.

5 Trust-Revenge Game

In this last game, we studied a more complex two-player domain. The Trust-Revenge Game, which will be described shortly, is designed to arouse three different emotions - trust, reciprocation (or fear) and revenge.

Research with human subjects on trust, reciprocation and revenge (or punishment) has been conducted in the past. The *investment game* was first introduced by Berg et al. [21]. In the *investment game* there are two types of players. Each player is given 10 chips at the beginning of the game. Players of type A are told that they can give some or all of their chips to a player of type B (this is the trust stage). The number of chips that Player A decides to give is multiplied by 3. Then Player B can give back some or all of what he was given (reciprocation stage). The subgame perfect equilibrium for this game is for both types of players to send nothing. Berg et al. conducted the experiment with students (human subjects). As expected, the human subjects did not act according to the subgame perfect equilibrium, and chips were transferred by both types of players.

Gneezy and Ariely [17] describe a variant of the investment game which includes an additional revenge phase. In their experiment, each of the two players starts off with \$10. The first player has to decide whether he wishes to end the game or to pass the full amount to the second player. If he decides to pass his money, the second player receives an additional \$40 for a total of \$50. Now the second player needs to decide whether to keep all of the money or to give half of the money back to the first player. If the second player decides to keep all of the money for herself, the first player may decide to take revenge on her and pay any amount from his own private money (up to \$25); this amount is multiplied by 2 and subtracted from the second player's revenue. Gneezy and Ariely's experiment showed that the first player often took revenge on the second player (when the second player kept all of the money for herself).

We use a variant of the game used by Gneezy and Ariely: the Trust-Revenge Game. This game is composed of three stages: *Trust*, *Reciprocate* and *Revenge*. This game is a "one-shot" game, i.e. after the three stages are completed, the game terminates (there are no repeated interactions). There are two types of players (A and B) in the game. At the beginning of the game Players A and B are both given a certain number of chips. The first stage is the *Trust* stage, where Player A is able to give any portion of his chips to Player B. There is a factor - the Trust Rate (**tr**) - by which the number of chips is multiplied when they are passed from Player A to Player B. The second stage is *Reciprocate*: after the chips have been transferred to Player B, Player B can decide how many chips to transfer back to Player A. Player B can transfer any number of chips (which she acquires) to player A. The third and final stage is *Revenge*: Player A plays another round in which he may pay any number of chips he has to the operator. Note that the chips are not transferred to anyone, merely subtracted from Player A's stack. However, in this round, Player B must pay a factor - Revenge Rate (**rr**) - on the number of chips Player A chose for revenge. Again, the chips are not transferred to anyone, merely subtracted from Player B's stack. Both the Trust Rate and the Revenge Rate are known to both players at the beginning of the game. Consider the following example: Assume that and $\mathbf{rr} = 6$. Assume that both players started with 10 chips. Trust stage: suppose Player A gives 5 chips to Player B. After applying the Trust Rate, Player B will receive 20 chips ($5 \cdot \mathbf{tr}$). Reciprocate stage: suppose Player B decides to give 7 chips to

Player A. At the end of this stage Player A has 12 chips $(5 + 7)$ and Player B has 23 chips $(30 - 7)$. Revenge stage: suppose Player A revenges 3 chips. At the end of this stage (which ends the game) Player A has 9 chips $(12 - 3)$, and Player B has 5 chips $(23 - 3 \cdot \mathbf{rr})$.

In this game there is a clear, unique subgame perfect equilibrium (SPE) strategy. In the revenge stage, there is no rational reason for Player A to revenge, therefore in the SPE there is no revenge. In the reciprocation stage there is no reason for Player B to reciprocate since she assumes that Player A is rational and that he will not revenge, therefore in the SPE there is no reciprocation. Therefore, in the trust stage there is no rational reason for Player A to trust since he knows that Player B will not reciprocate. Therefore the SPE says: ”don’t revenge, don’t reciprocate, don’t trust”. Since the SPE expects a 0 chip action in each of the three stages, we refer to a positive transfer (or payment) in each of the stages as expressing “emotions”. While the trust stage expresses trusting and the revenge stage expresses revenging, the reciprocation stage can either express reciprocation or fear of the other player’s revenge. We do not claim, though, that actual emotions are the only explanation for a positive number of chips transferred. Obviously human subjects do not use the SPE and they show all of the “emotions” mentioned, We will examine what the agents do: Is human emotion embedded in the strategy of the agents?

We had a different set of 37 undergraduate computer science students compose automated agents for the Trust-Revenge Game. Each agent played twice against all of the other agents, once as Player A and once as Player B. The same students also played the game themselves via the web against other human subjects (not against agents). The students’ grades depended on their agents’ performance and its documentation. The agent’s performance was measured according to its final result (relative to the number of chips) and did not depend on the opponent’s performance or on the average performance. The students were explicitly informed of this grading policy. Their performance as humans in the web-based game was added as a bonus to their agents’ performance, assuring that the incentive to play well was identical in both cases. The students knew that the automated agents would play with other automated agents, and, when playing on the web, they knew that they were playing against a human, however they did not know against whom they were playing and we assured them that this information would remain confidential. Table 1 shows the 3 settings that were used.

Settings Index	Player A Initial	Player B Initial	Trust Rate	Revenge Rate
1	10	10	3	3
2	10	10	6	6
3	20	0	6	6

Table 1. Settings Used in the Trust-Revenge Game

In this experiment we chose to have the same group of students who programmed the agents also play the game themselves as human subjects. The students first programmed their agent and then played the game themselves. We did so for the following reasons: 1. The Trust-Revenge game requires some familiarity with the population which each player is facing. Without this familiarity it is very hard to anticipate what the other player would do and therefore hard to know whether to trust her or not. 2. By having the students first build their agents and then play we guaranteed that the human subjects understood the game at least as much as the automated agents' programmers did. 3. Having the exact same group playing both as humans and programmers eliminates any culture or education bias.

We tested both the percentage of players that gave away chips in each stage of the game (Fig.4) and the number of chips transferred (or paid) on average at each stage (Figure 5). The transfer of chips in the first stage expresses trust, in the second it expresses either reciprocation or fear of revenge, and in the last stage it expresses revenge. As can be seen in Figures 4 and 5, all "emotions" are expressed both by the human subjects and by the automated agents. Although it is clear that there is a substantial degree of "emotion" expressed by the agents, they still clearly express less "emotion" than the humans.

Another important insight we would like to mention is that using the SPE when playing versus humans or automated agents doesn't yield the highest outcome. As can be seen in Figure 5, on average Player B reciprocated more than Player A trusted, with both humans and automated agents. No Player B ever reciprocated if Player A did not trust; therefore on average Player A gained from trusting (unlike the SPE which requires that there be no trust at all).

However, the Revenge stage is different. We know from the documentation we collected from the students who designed the agents that they understood that revenge would lower their final profit and that it was not beneficial in this single-shot game. Yet we were surprised to find that they still designed agents that revenge.

6 Discussion and Conclusions

Scientifically designed automated agents or automated agents designed by experts often implement a fully rational strategy. However, most pieces of software are written by amateurs (programmers with no more than a Bachelors' degree in computer science) and therefore are likely to "suffer" from the same irrationalities as humans (but to a reduced extent). Indeed, just as agents designed by experts behave more rationally, humans who acquire vast experience in a game become more rational players [33,12].

In all settings and environments that we tested, the automated agents exhibited irrational behavior. Agents paid chips or spent time and resources even though this is not considered rational behavior. The type of irrational behavior displayed by the agents is the same as the human irrationality, but to a lesser extent. Most surprisingly, even when the rational behavior was clear to the agents and irrational behavior consisted of a clear loss of utility, agents still exhibited irrational behavior (as was the case for the revenge stage of the third game). We would like to note, that while in the first two games the agents were much closer to the rational behavior than to human behavior, in the third game the opposite occurred. We suggest that this is because that the second game involved emotions, and therefore the players' subjective utility was different than the actual expected final stake. E.g. a human may gain subjective utility from the feel-

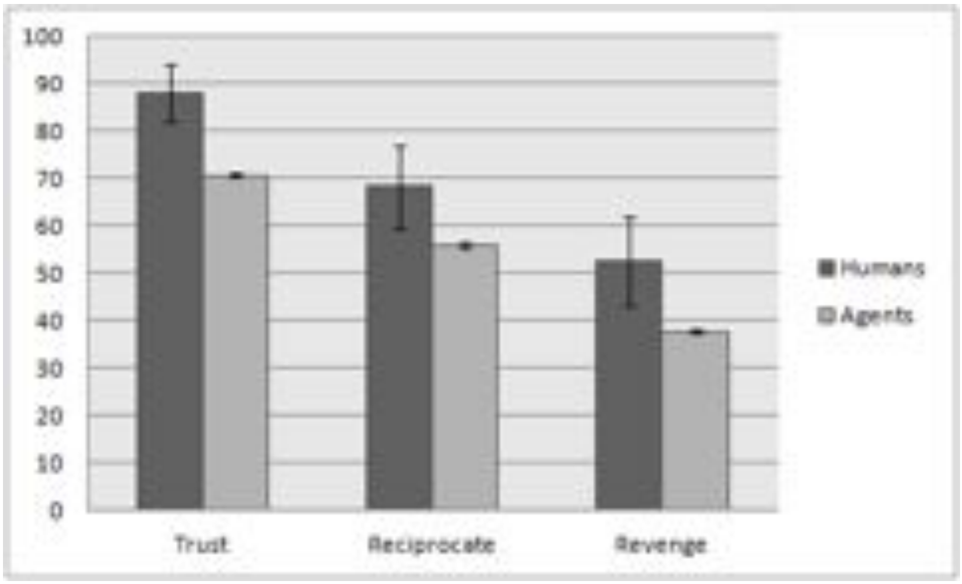


Fig. 4. Percent of Players Using Each Stage

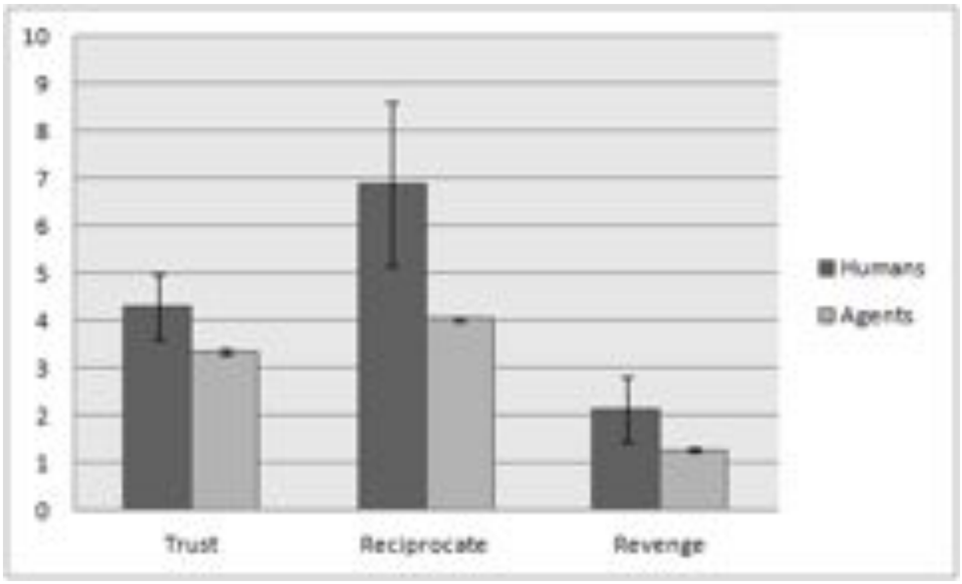


Fig. 5. Average Action in Each Stage (in chips)

ing that justice is being made when he revenges, and therefore embeds this behavior in his agent. Quantifying and predicting the level of automated agents irrationality in comparison to that of humans is a topic for future work.

When building an automated agent that needs to interact with other automated agents, one needs to take into account that the other automated agents are very likely to show irrationalities that can be observed in humans. Assuming that other agents behave rationally when they actually do not, inhibits performance. In poker, for example, it is very likely that automated agents will exhibit attributes that humans use when playing, which many times aren't rational, such as over-bluffing. In electronic commerce, automated agents may be influenced by anchoring or the "sunk cost" effect. However, our results indicate that the automated agents are expected to show a smaller degree of these irrational behaviors. Therefore, considering human behavior when constructing an automated agent for playing poker, for electronic commerce or for other domains seems to be a promising approach.

References

1. AAAI. Annual computer poker competition. http://www.computerpokercompetition.org/, 2012.
2. AAMAS. Automated Negotiating Agents Competition (ANAC). http://mmi.tudelft.nl/anac/, 2012.
3. Amazon. Mechanical Turk services. http://www.mturk.com/, 2012.
4. Dana Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, nov 1987.
5. D. Ariely, G. Loewenstein, and D. Prelec. Coherent arbitrariness: Stable demand curves without stable preferences. *Journal of Business Research*, 59(10-11):1053–1062, 2006.
6. Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. Gambling in a rigged casino: the adversarial multi-armed bandit problem. In *Proc. of FOCS*, pages 322–331, 1995.
7. A. Azaria, Y. Aumann, and S. Kraus. Automated strategies for determining rewards for humanwork. In *AAAI*, 2012.
8. A. Azaria, Z. Rabinovich, S. Kraus, and C. V. Goldman. Strategic information disclosure to people with multiple alternatives. In *AAAI*, 2011.
9. A. Azaria, Z. Rabinovich, S. Kraus, C. V. Goldman, and O. Tsimhoni. Giving advice to people in path selection problems. In *AAMAS*, 2012.
10. Jordi Brandts and Gary Charness. The strategy versus the direct-response method: a first survey of experimental comparisons. *Experimental Economics*, 14(3):375–398, September 2011.
11. C. F. Camerer. *Behavioral Game Theory. Experiments in Strategic Interaction*, chapter 2. Princeton University Press, 2003.
12. Colin Camerer and Keith Weigelt. Experimental tests of a sequential equilibrium reputation model. *Econometrica*, 56(1):1–36, 1988.
13. David Carmel and Shaul Markovitch. Opponent modeling in multi-agent systems. In Gerhard Weiss and Sandip Sen, editors, *Adaption And Learning In Multi-Agent Systems*, volume 1042. 1996.
14. John D Claxton, Joseph N Fry, and Bernard Portis. A taxonomy of prepurchase information gathering patterns. *Journal of Consumer Research*, 1974.
15. David H. Furse, Girish N. Punj, and David W. Stewart. A typology of individual search strategies among purchasers of new automobiles. *Journal of Consumer Research*, 1984.
16. Y. Gal and A. Pfeffer. Modeling reciprocity in human bilateral negotiation. In *AAAI*, 2007.
17. A. Gneezy and D. Ariely. Don't get mad get even: On consumers' revenge. *manuscript*, 2010.
18. Barbara J. Grosz, Sarit Kraus, Shavit Talman, Boaz Stossel, and Moti Havlin. The influence of social dependencies on decision-making: Initial investigations with a new game. In *AAMAS*, pages 782–789, 2004.
19. K. Hindriks and D. Tykhonov. Opponent modelling in automated multi-issue negotiation using bayesian learning. In *AAMAS - '08*, pages 331–338, 2008.
20. P. Hoz-Weiss, S. Kraus, J. Wilkenfeld, D. R. Andersend, and A. Pate. Resolving crises through automated bilateral negotiations. *AIJl*, 172(1):1–18, 2008.
21. Berg Joyce, Dickhaut John, and McCabe Kevin. Trust, Reciprocity, and Social History. *Games and Economic Behavior*, 10(1):122–142, Jul 1995.
22. Alessandro Lazaric, Mario Quaresimale, and Marcello Restelli. On the usefulness of opponent modeling: the kuhn poker case study. In *AAMAS '08*, volume 3, pages 1345–1348, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.

23. George Loewenstein. Anticipation and the valuation of delayed consumption. *Economic Journal*, 97(387):666–84, 1987.
24. Efrat Manistersky, Raz Lin, and Sarit Kraus. *The Development of the Strategic Behavior of Peer Designed Agents*. Lecture Notes in AI, to appear.
25. Peter McCracken and Michael Bowling. Safe strategies for agent modelling in games. In *AAAI Fall Symposium on Artificial Multi-agent Learning*, October 2004.
26. Y. Oshrat, R. Lin, and S. Kraus. Facing the challenge of human-agent negotiations via effective general opponent modeling. In *AAMAS*, 2009.
27. G. Paolacci, J. Chandler, and P. G. Ipeirotis. Running experiments on Amazon Mechanical Turk. *Judgment and Decision Making*, 5(5), 2010.
28. N. Peled, Y. Gal, and S. Kraus. A study of computational and human strategies in revelation games. In *Proc. of AAMAS*, 2011.
29. Patrick Riley and Manuela Veloso. Planning for distributed execution through use of probabilistic opponent models, 2001.
30. A. Rosenfeld and S. Kraus. Using aspiration adaptation theory to improve learning. In *AAMAS*, Taipei, 2011.
31. Jorma Sajaniemi. Guest editor's introduction: Psychology of programming : Looking into programmers heads. *Human Technology: An Interdisciplinary Journal on Humans in ICT Environments*, 2008.
32. Daniel Schunk and Joachim Winter. The relationship between risk attitudes and heuristics in search tasks: A laboratory experiment. *Economic Behavior Organization*, pages 347 – 360, 09.
33. Reinhard Selten and Rolf Stoecker. End behavior in sequences of finite prisoners dilemma supergames, 1986.
34. J. Shin and D. Ariely. Keeping doors open: The effect of unavailability on incentives to keep options viable. *Management Science*, pages 575–586, 2004.
35. TAC Team, Michael P. Wellman, Peter R. Wurman, Kevin O'Malley, Roshan Bangera, Shoude Lin, Daniel Reeves, and William E. Walsh. Designing the market game for a trading agent competition. *IEEE Internet Computing*, 5(2):43–51, mar 2001.
36. The iterated prisoner's dilemma competition. http://www.prisoners-dilemma.com/, 2012.
37. A. Tversky and D. Kahneman. The framing of decisions and the psychology of choice. *Science*, 211(4481):453–458, 1981.
38. Martin Weitzman. Optimal search for the best alternative. *Econometrica*, 47(3):641–54, May 1979.

A modeling framework for inter-cultural social interactions

Taranjeet Singh Bhatia, Saad Ahmad Khan and Ladislau Bölöni

Dept. of Electrical Engineering and Computer Science
University of Central Florida
4000 Central Florida Blvd, Orlando FL 32816
{tsbhatia,skhan,lboloni}@eecs.ucf.edu

Abstract. In this paper, we describe a modeling framework which allows us to reason about the social-cultural behavior of humans in interaction scenarios. The objective is to provide explanatory and predictive power to an agent or robot which either witnesses or participates in the interaction: why have the human agents behaved the way they are and what kind of behavior can I expect as an answer to certain actions? We describe a the scenario modeling framework, the culture-dependent Culture Sanctioned Social Metrics (CSSM) model and the scenario dependent Concrete Belief (CB) model. We illustrate the use of the framework using an inter-cultural scenario.

1 Introduction

For an autonomous robot or software agent to participate in the social life of humans, it must have a way to perform a *calculus of social behavior*, an operational model which would allow it to reason about the implications of it's own and the humans' actions.

To be useful, such a calculus must satisfy several requirements. First, it must have *explanatory power* (it must provide a coherent theory for why the humans act the way they do), and *predictive power* (it must provide some plausible scenarios about the future actions of the humans). The social calculus must also be *culture-specific*, it needs to consider the different social norms and requirements in various societies. As many social interactions take place in public, it also needs to have a *model of public perception*.

The social calculus does not directly determine the behavior of the agent (unless the only goal of the agent is to act in a socially acceptable way). However, the behavioral and planning component can use social calculus to weigh different plans - this is especially critical if the plans require the goodwill or active help of the human actors.

In this paper, we describe a set of social calculus techniques designed to satisfy these requirements. In Section 2 we describe a formal model for the representation of social interactions, with special focus on the separation of the progress state and the full state. The next two sections introduce specific models for the representation of the components of the full state: culture sanctioned social metrics in Section 3, and concrete beliefs of individual actors and perception of the crowd in Section 4. Section 5 describes the use of the model on an intercultural interaction example. We briefly survey related work in Section 6.

2 Scenario modeling

2.1 The scenario model

The modeling of arbitrary, free format interactions between humans is clearly out of reach for theoretical models. Instead, we model specific *scenarios of human interaction* which center around the resolution of a small number of issues, have a limited number of participants and a limited time span.

Definition 1. *We call a* scenario *a tuple* $\{\mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\tau}, \mathbf{S}, \mathcal{F}, \mathcal{P}\}$*, where:*

> $\mathbf{A} = \{A_1, A_2 \ldots\}$ *is a set of* actors*, who are usually humans, although they can also be autonomous robots or software agents.*
> $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2 \ldots\}$ *is a set of distinct* action types*. A concrete action* a *is characterized by the tuple* $\{\alpha, A, (x_1 \ldots x_n)\}$*, that is, by the action type, the performing actor and a list of parameters of arbitrary length. We denote with* $\mathbf{a}$ *the (not necessarily discrete) space of all posible actions.*
> $\boldsymbol{\tau} \subset \boldsymbol{\alpha}$ *is the collection of* terminal action types*. A terminal action, for any actor and parametrization, terminates the scenario.*
> $\mathbf{S}$ *is the (not necessarily discrete) collection of* full states of the scenario S*.*
> $\mathcal{F}$ *is the* action impact function $\mathcal{F} : \mathbf{A} \times \mathbf{S} \times \mathbf{a} \rightarrow \mathbf{S}$*. We interpret* $S' = \mathcal{F}(A, S, a)$ *as the new full state of the system if actor* A *performs action* $a(\alpha, x_1 \ldots x_n)$ *in state* S*.*
> $\mathcal{P} : \mathbf{A} \times \mathbf{S} \rightarrow \boldsymbol{\alpha}^*$ *is the* progress function*. We interpret* $\mathcal{P}(A, S) = \{\alpha_{p1}, \ldots, \alpha_{pn}\}$ *as the set of action types available to actor* A *in state* S*. If the actor can perform a certain action type, it is free to use an arbitrary parametrization of it.*

2.2 The progress model

The progress function $\mathcal{P} : \mathbf{A} \times \mathbf{S} \rightarrow \boldsymbol{\alpha}^*$ had been defined on the full state space of the scenario, which, in many cases is very large and not necessarily discrete. However, many human interaction scenarios are *progress-segmented*, that is, the full states can be grouped into equivalence classes with regards to the output of the progress function.

Definition 2. *We define* $\mathbf{P} = \{P_1, P_2 \ldots P_n\}$ *the collection of a finite number of* progress states*. A progress state* P *is a (not necessarily discrete) collection of full states, such that if* $S \in P \wedge S' \in P \Rightarrow \forall A \;\; \mathcal{P}(A, S) = \mathcal{P}(A, S')$*. The* progress state discretization *function* $PSD : \mathbf{S} \rightarrow \mathbf{P}$ *maps states to progress states.*

Definition 3. *We will call the function $\mathcal{P}_R : \mathbf{P} \times \mathbf{A} \to \boldsymbol{\alpha}^*$ the* reduced progress function *and define it as $\mathcal{P}(A, S) = \mathcal{P}_R(A, PSD(S))$.*

In contrast to $\mathcal{P}$, the reduced progress function $\mathcal{P}_R$ is defined on a discrete and (usually) small space. We will also consider an even more specific class of scenarios where for every progress state only one actor is allowed to take actions.

Definition 4. *We define a* turn taking scenario *a progress-segmented scenario where for any progress state P the reduced progress function $\mathcal{P}_R(A, P)$ is non-empty for only one specific actor A_t. We say that A_t* has the turn *in progress state P.*

2.3 A simple example: Human Bargaining

To illustrate the model, let us consider a simple example. In the Human Bargaining scenario two humans, a seller A and a buyer B are arguing over the price of a good. The action type set contains three action types: $\boldsymbol{\alpha} = \{\alpha_O, \alpha_a, \alpha_w\}$ with the following interpretation:

α_O	making an offer
α_a	accepting the latest offer
α_w	withdrawing from the bargaining

The choice of the parameters is a function of the action type, the social context and the goals and limitations of the model. If we assume that the actors are simple software agents, a single parameter (the value of the offer) is sufficient. If, however, the actors are humans many other parameters can be considered: the verbal phrasing of the offer, the politeness of the addressing form used, the tone of the voice, the body language and facial expressions which accompany the offer and so on.

The scenario can be modeled using only four progress states $\mathbf{P} = \{OA, OB, TN, TP\}$:

OA	turn of A to take an action
OB	turn of B to take an action
TN	the bargaining had been broken with no-deal
TP	deal accepted

Note that this is a turn taking scenario: in progress state OA only actor A can take an action, in progress state OB only B, while TN and TP are terminal states.

In this case the reduced progress function $\mathcal{P}_R$ can be visualized as a *progress graph*, a directed graph where the nodes are progress states and the edges are labeled with the pair of an actor and an action type (see Figure 1).

The progress graph is a helpful modeling tool for the knowledge engineer, but it should not be mistaken for a *full state-action graph* of the scenario.

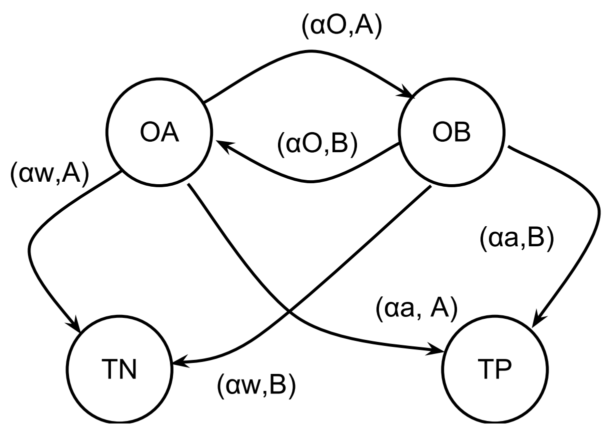


Fig. 1: The progress graph of the Human Bargaining scenario

Such a graph would have full states S as nodes and fully parametrized actions $a(\alpha, A, x_1 \ldots)$ as edges. The full state-action graph is a suitable model for decision theoretic analysis - for instance, it can form the basis of a Markov Decision Process. The progress graph is not sufficient for this.

Normally, the full state includes orders of magnitude more information than the progress state. Even if, A and B are software agents, the full state would have to include the pending offers, the internal valuations of the good by the actors A and B, their negotiation strategies, and possibly other factors such as their models of each other. If A and B are humans, the full state is even more complex: it might include factors such as the level of annoyance of the actors, judgment of personal dignity, the feelings of friendship or animosity towards the negotiating partner, and so on.

We conclude that the full state-action graph is too large for human visual analysis, even for the most simple scenarios such as Human Bargaining. In contrast, the progress graph remains small enough for human intuition for turn taking scenarios, and even for some scenarios which do not verify the turn taking criteria at every progress state.

2.4 Group actors and overlapping scenarios

The scenario framework allows us to segment the social-cultural behavior of human actors into manageable entities. An argument can be made, however, that the flow of events in social life is not normally segmented into such clearly defined scenarios. We will further increase the representational power of the model in two ways:

– We will allow *group actors* to model coordinated groups or unorganized crowds of humans. Naturally, the state and the actions taken by the group actors must be compatible with their internal organization.

– We will allow more than one scenario to be executed simultaneously, with some participants (for instance, the crowd) participating in more than one scenario. As the states of the shared participant are part of the full state of both scenarios, this will allow state information to *leak* from one scenario to another.

2.5 The next steps

The progress model, as defined above, allows us to describe the general flow of the scenarios without having to consider the full richness of the state characteristic for human interactions. The progress states and the reduced progress function describe the various options actors have at certain moments in the scenario - but they do not have explanatory and predictive (E&P) power. The actors can take any action which is permitted by the reduced progress function, without the ability to predict what they will do and how, nor in retrospect why they did it.

To add E&P power to our model, in the following sections we take a closer look at the full state space $\mathbf{S}$ and the action impact functions $\mathcal{F}$.

In particular we will make the claim that the large majority of human interaction scenarios can be modeled with E&P power while restricting the state to only three types of values: *culture sanctioned social metrics* maintained by individual actors, a small set of *beliefs about concrete facts* (maintained by individual actors) and *perceptions* (maintained by group actors). We will show that these values can be given rigorous definitions, and can be reliably measured and estimated.

3 Culture-sanctioned social metrics

3.1 Definitions

Human interaction scenarios have complex states - involving the physical, social, cultural, psychological and even physiological aspects of the actors. Relatively obscure circumstances have been shown to have decisively affect the outcome of certain human interaction scenarios (see for instance the work of Kahneman and Tversky about judgement under uncertainty [22]).

We argue, however that in most human scenarios, the cultural expectations reduce the number of choices the participating humans can take, and such scenarios can be explained and reasonable predictions made based on a set of explicit metrics of the state, which are well specified for a given culture and can be readily estimated by the actors. We will call these *culture sanctioned social metrics* (CSSMs). We say that a culture sanctions a metric if it:

– provides a *name* for it
– provides an (informal) *algorithm* for its evaluation
– expects its members to *continuously evaluate* the metrics for themselves and salient persons in their environment

– provides *rules of conduct* which depend on these parameters

The CSSMs can be either tangible or intangible. Tangible metrics such as financial worth or spent time can be measured by physical means (although many times they are only estimated). Intangible metrics, such as politeness, dignity, “face” or friendliness are socially constructed, not directly measurable and depend on the specific culture.

3.2 Culture specificity and cross-cultural considerations

CSSMs are always defined by a specific culture, and the name of the metric is given in the language of the culture. Knowing the name of a metric is insufficient: it order to be educated in a culture an individual must know the evaluation algorithm and the rules of conduct associated. It is not required, however, for the individual to *follow* the rules of conduct - however, he or she will be *aware* of the rules and their transgression.

The same name might define different metrics in different cultures. For instance, the word “dignity” has different evaluations and rules of conduct in different English speaking cultures. The dictionary translation of the word in other languages, such as *‘azmat’* in Urdu and *‘pratistha’* in Hindi, can yield an even more divergent CSSM.

This being said, there are many CSSMs which appear in several cultures in identical or near identical form. There are groups of cultures with closely related metrics - for instance the cultures aligned with the Western European model, the culture of China and nations influenced by Chinese culture and the cultures of the Near East and North Africa. In addition, certain CSSMs are cross-cutting geographical, language and religious boundaries, such as the striking similarities between “cultures of honour” in places as far away as the Scottish highlands, the Bedouins of the Sahara or the Southern USA [17].

It is beyond the scope of this paper to establish specific measures of similarity between different cultures. A number of established metrics in sociology can be used as a starting point for such classification (for instance, Hofstade’s cultural dimensions [10]).

3.3 The problem of perspective

Many rules of conduct associated with CSSMs consider not only the actor’s own perspective, but the perspective of other actors in the scenario. For instance, in the Japanese definition of politeness, it is not sufficient for the actor to act such that its own evaluation of politeness is correct. Instead, the perception of the interaction partner and of external observers is just as (or more) important.

Taking this into consideration, a CSSM is identified by five parameters: *CSSM(C, M, SA, PA, EA)*, where:

– *C* is the culture which defines the CSSM and specifies its rules.

– *M* is the name of the metric, which is unique in the given culture (but different cultures might mean different metrics under the same name).
– *SA* is the *subject actor*, the actor who owns the metric.
– *PA* is the *perspective actor*, from whose perspective the metric is evaluated.
– *EA* is the *estimator actor*, who estimates the CSSM and who is the owner of the knowledge.

The subject, perspective and estimator actors can be the same. For a CSSM to play a role in a scenario, we need that the EA to be cognizant of culture C. In addition, it is necessary for EA to believe that PA is cognizant of culture C (although this belief might be incorrect). It is not necessary for SA to be cognizant of the culture (although whether he is or not might be a factor in the behavior of other actors).

In the following we will present several examples of CSSMs. Our goal will be to illustrate that all the five parameters of the CSSM model are necessary for building a model with E&P power.

Self perspective the *CSSM(Western, dignity, John, John, John)* represents John's estimate of its own dignity, in the Western cultural model.

Peer perspective the *CSSM(Western, politeness, John, Mary, John)* represents John's estimate about how Mary sees his politeness. If John cares about Mary's opinion, he will adjust its behavior in such a way that Mary's perspective will improve. Note that this value might not be identical to *CSSM(Western, politeness, John, Mary, Mary)*, that is, Mary's own opinion about John's politeness.

Cross-cultural perspective Let us consider the case of János, a Hungarian businessman in China, who publicly admits to a business partner Chen a mistake in formulating a purchase order. This will affect *CSSM(Chinese, Face, János, János, Chen)* that is, Chen's estimate of János's own estimate of loosing face. In this context, Chen does not understand why János would do such a thing. What happens here, is that Chen is evaluating a CSSM which János does not: János is not educated in Chinese culture, and the concept of "face" as a metric does not exist in Hungarian. Nevertheless, this particular CSSM can be an important consideration in action - for instance, Chen might act to prevent János from loosing face, even if János is unaware of this.

3.4 Intra-cultural uniformity

The multiplication of possible perspectives increases the complexity of the CSSM evaluation. However, this increase is mitigated, in part, by the *intra-cultural uniformity conjecture.*

Conjecture 1. Let us consider two human actors A and B, educated in the same culture which defines a CSSM x. Let us now consider a scenario S and a series of actions $a_1, \ldots, a_n$, of which both A and B are aware. The *intra-cultural uniformity conjecture* asserts that the evaluation of A and B of the CSSM x will be similar, irregardless of the position of A and B with regards of the scenario (they can be active participants, real-time passive observers or post-facto judges).

This conjecture is supported by the definition of the CSSM: the two actors have the same information and they use the same algorithm for the evaluation provided by the shared culture.

An example of what the conjecture says is as follows: let us consider two Japanese persons, one of them a participant in a social situation with involves interacting with a Westerner while the other one an outside observer. Let us now consider that the Westerner unknowingly commits an action which is considered impolite in the Japanese culture. The intra-cultural uniformity conjecture states that the two Japanese participants will evaluate the level of impoliteness similarly. This fact will not be changed by the fact that the Japanese might also be familiar with the Western culture.

In conclusion, between two participants educated in the same culture, any difference in the evaluation of the CSSM is reduced to the perspective actor's knowledge of specific events. (Naturally, if the estimator actor itself is unaware of an event, it will automatically mean that he or she cannot assign it to the perspective actor either).

3.5 The problem of cognitive load

The evaluation of CSSMs is a significant *cognitive load*. Although the culture requires every actor to continuously evaluate all CSSMs for *every* salient person in the environment, in complex situations with many actors present, many actors will not be able to evaluate every possible action impact function.

Different CSSMs, actions and actors will be differently affected by the problem of cognitive load. The more complex a CSSM, the more likely that it will not be estimated. Self perspective CSSMs will more likely be evaluated than cross-cultural peer perspectives. The actors involved in the CSSM also affect their priority. CSSMs where the subject and perspective actors are random members of a crowd will have much lower priority than CSSMs where the subject actor and/or perspective actor is the self or close peers.

Finally, the salience of the action also affects the priority. Striking actions, such as large gestures, loud voice, strong verbal expressions will raise the action's evaluation priority.

3.6 Numerical values of CSSMs

For the purpose of our analysis, we will map the CSSMs to a numerical value in the range of $[0.0, 1.0]$. In order to achieve modeling fidelity, this value must be *calibrated*, that is, different numerical values must be associated with the internal perception of the CSSMs (such as degrees of politeness) by the human actors.

The simplest way to perform calibration is by directly requesting numerical values from the human informants. For instance, we can present a human informant with a social situation and ask her to rate the politeness of actor X on a scale from 0 to 100. However, this approach would only work with informants who are comfortable expressing social values on numerical scales.

However, we conjecture that the cultural education with respect to CSSMs is not transmitted through numerical or degree forms, but through narratives containing key words, which can be assigned to various points. It is the task of the knowledge engineer to create a mapping from keywords to numerical values.

4 Concrete beliefs

In the previous section we discussed about the CSSMs which are components of the full state given by the culture. These extend across multiple scenarios, and span the life of the actors even outside the specific scenarios. However, in order to model the scenario accurately, we also need to consider beliefs of the agents which are directly relevant to the specific scenario. We find that for achieving E&P power, we don't need to model large sets of belief structures. Rather, we are only interested in the agent's belief about a small number of *concrete questions* which are important for the ongoing scenario. Examples of concrete questions are "Actor X is holding a gun" or "Actors A and B are engaged in a commercial transaction". We will only consider questions which can be answered unequivocally by an omniscient external observer of the scenario. The participants of the scenario, however, need to work with incomplete knowledge and limited rationality, thus they maintain an uncertain answer.

We will call *concrete beliefs* (CBs) the beliefs maintained by the actors in a scenario with regards to the answers of concrete questions. We say that a scenario implies a CB if:

- there is an algorithm which an omniscient external observer of the scenario could use to unequivocally answer the question underlying the CB
- the scenario expects at least one actor to *continuously evaluate* the CB for themselves and salient persons in their environment
- the scenario provides *rules of conduct* which depend on the CB *or* the CB affects the calculation of CSSMs of the actors involved in the scenario

Concrete beliefs can influence a scenario even when they are evaluated from another player's perspective. Taken this into consideration, we will identify a concrete belief with four parameters: *CB(SC, BD, PA, EA)*, where:

- *SC* is the scenario which specifies the question
- *BD* a description of the belief (normally, through the associated question)
- *PA* is the *perspective actor*, from whose perspective the belief is evaluated.

– *EA* is the *estimator actor*, which performs the estimate and who is the owner of the knowledge.

The definition of the CBs parallel the definition of CSSMs, with several important differences. While CSSMs are defined by the culture, the CBs are tied to a specific scenario. The evaluation of the CBs might be incomplete or wrong due to the lack of information or cognitive overload of the estimator actors. The rules associated with CBs might be broken - but the agents estimating the CB will be aware of the fact that they broke a rule. Another different is that the CB does not provide information for the owner actor. While CSSMs are always an internal value represented by the agent, CBs can represent concrete physical values.

4.1 Representation formats of CBs

CBs can represent any statement which can be made about the scenario, and thus can take various formats. For instance, for situations involving negotiating about goods, the CB represents beliefs about the private values of different actors. In such situations, the statement can have a scalar or, in the case of multi-issue negotiations, a vector value, and the corresponding CBs will take the form of probability distributions of these values.

Another major type of CBs are cases when the question has a boolean answer, where the CB can be a simple probability number. However, for many applications, it is desirable to use a method which keeps track not only the incidents, while at the same time also maintains a representation of the uncertainty. This is especially important if we do not want to continuously maintain all the evidence for the belief.

Our current approach is based on the Dempster-Shaffer model of evidence with the following assumptions:

- the agent's current beliefs are fully encoded in the mass function
- no previous evidences are remembered
- incoming evidence can be weighted by significance
- at every incoming evidence, the belief is updated using the standard Dempster's rule of combination (conjunctive merge).
- the value for the positive belief is used as the indicator of the belief

5 An example of a multi-cultural scenario

Let us now illustrate the way in which the model described in the previous sections can be used to model a simple scenario involving multi-cultural interaction. The Give Way scenario involves two agents A and B approaching simultaneously a door. We assume that the agents are humans, potentially of different cultures, who can have various ages, gender and social status. The scenario also generalizes to situations where one of the agents is a robot.

For each of the agents, there are three different resolutions (1) enter the door first, (2) open and hold the door to the other agent and (3) give way to the other agent to enter first. We assume that the agents do not know each other, they might act under different cultural assumptions, that is they have different CSSMs, with different update rules and associated social requirements. A further complexity can be considered if the scenario happens in the view of the public, in which case the agents also need to consider their estimates of the beliefs of the crowd, in forms of CBs.

Let us now proceed to model the Give Way scenario using the framework developed in the previous sections. The scenario can be modeled with three action types as shown in Table 1. The progress graph, where the nodes are progress states and the edges are labeled with the pair of an actor and an action type is shown in Figure 2). The scenario begins with the start state SS and continues until one of the agents perform the action $\alpha 2$ to reach the terminal state TN.

Table 1: The action types of the Give Way scenario.

Action type	Description
$\alpha 1$	open the door
$\alpha 2$	enter the door
$\alpha 3$	give way to other agent

Let us now consider the CSSMs which determine the behavior of the agents in the scenario. Depending on the cultural background of the agents, different set of CSSMs are evaluated. We will consider agent's of two different cultures, Western and Indian. For the purpose of this paper, we will assume that the two participants are of the same gender and they do not have a significant difference of social rank. By and large, politeness considerations in Western culture require the agents to give way to the peer (although this requirement is frequently ignored). In Indian culture, giving way is considered an ineffectual, wimpish behavior.

Thus our model will consider three CSSMs, one concrete (time), and two intangible (politeness and wimpiness). The time T is the amount of time spent on the current scene measured in seconds. Every time taking action $\alpha 3$ by agent imposes a penalty of 5 seconds. In general, agents avoid wasting time.

The politeness is the conformance to the perceived social norms of speech and gestures. Both Western and Indian cultures have the definition of politeness, but there are different definitions associated with them, which translate into different action impact functions. Giving way in the Western culture is considered polite behavior. In the Indian culture however, giving way to a stranger does not impact the perception of politeness. We will consider the private, peer and public politeness aspects: *CSSM(Western, politeness, A, A, A)*, *CSSM(Western, politeness, A, B, A)* and *CSSM(Western, politeness, A, Crowd, A)*.

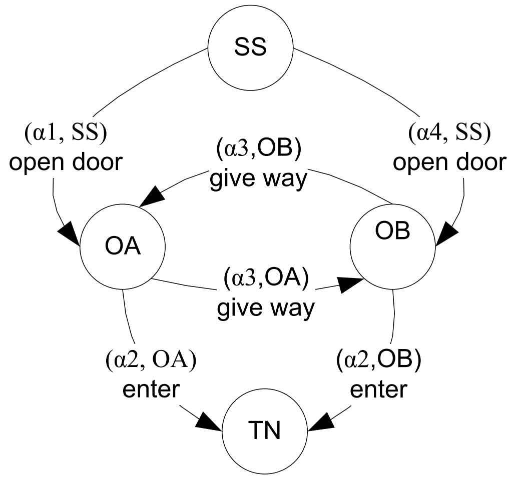


Fig. 2: The progress graph of the Give Way scenario

Wimpiness is the degree of lack of confidence and courage in a person to take initiative. Again, both cultures have definitions for this metric. However, giving way to a person of equal rank does not impact perception of wimpiness in the Western culture, however, it does in the Indian one. We will consider the private, peer and public wimpiness aspects *CSSM(Indian, wimpiness, A, A, A)*, *CSSM(Indian, wimpiness, A, B, A)* and *CSSM(Indian, wimpiness, A, Crowd, A)*.

To achieve E&P power, the analysis of the scenario needs to consider two concrete beliefs, concerning the culture of the two agents, and we need to consider this from the perspective of each other and, potentially, of the crowd. Naturally, *CB(GiveWay, Is-A-an-Westerner, A, A)* is a fixed value, because normally A would know whether he is a Westerner or not. On the other hand, *CB(GiveWay, Is-A-a-Westerner, B, B)*, representing B's belief whether A is an Indian, and *CB(GiveWay, Is-A-a-Westerner, Crowd, Crowd)*, representing the crowd's belief whether A is an Indian are values whose calculation contributes to the E&P power of the model.

Let us now illustrate through several examples the way in which the model traces the evolution of the CSSMs for agents in different cultures. We have modeled the Give Way scenario using our framework, and we traced the evolution of CSSMs for four different sequences of events, each of them representing a different path through the scenario:

$$\text{SS} \xrightarrow{\alpha1} \text{OA} \xrightarrow{\alpha3} \text{OB} \xrightarrow{\alpha2} \text{TN}$$

$$\text{SS} \xrightarrow{\alpha 4} \text{OB} \xrightarrow{\alpha 2} \text{TN}$$

$$\text{SS} \xrightarrow{\alpha 4} \text{OB} \xrightarrow{\alpha 2} \text{TN}$$

$$\text{SS} \xrightarrow{\alpha 1} \text{OA} \xrightarrow{\alpha 3} \text{OB} \xrightarrow{\alpha 2} \text{TN}$$

These scenarios, however, lead to different perceptions and social metrics depending of the culture of the participating humans. We will describe two experiments with different outcomes and cultural backgrounds of the participants.

Experiment 1: we consider that both agents A and B belong to the Indian culture. In this case neither of them consider the politeness as a CSSM. As both parties have the same culture, their peer perceptions are a good approximation of the opponents self perception, *i.e. CSSM(Indian, wimpiness, A, A, A)* ≈ *CSSM(Indian, wimpiness, A, A, B)*. This allows the agents to make a reasonable prediction of the opponent's actions.

Let us assume that agent B arrives at the door before A, and simply moves on without being polite and giving way to A. The *CSSM(Indian, wimpiness, B, B, B)* will be lowered, while *CSSM(Indian, wimpiness, A, A, A)* will not be affected. In colloquial terms, A can feel himself as a efficient and non-wimpy person, and this can explain its behavior, and can be used to predict it.

Experiment 2: let us now consider an experiment in which we don't know the culture of agents A and B, however, we know that the culture of the onlookers (modeled as the Crowd agent) is Indian. If we don't know what culture the agent's belong to, we can simply not trace any CSSMs and CBs for it. We can, however, trace the crowd's belief. Let us consider a scenario where A approaches the door and opens it to agent B. Let us now see how the crowd can reason about this. If A is a westerner, than his politeness level will increase *CSSM(Western, politeness, A, A, Crowd)*. On the other hand, if A is an Indian, his wimpiness will increase *CSSM(Indian, wimpiness, A, A, Crowd)*. As the same action appears rational for a Westerner, but irrational for an Indian, the crowd will treat this occurrence as an evidence which increases *CB(GiveWay, Is-A-a-Westerner, Crowd, Crowd)*.

6 Related work

The study of social behavior of humans had been extensively studied in sociology, psychology and anthropology. The models developed in social sciences frequently rely on the understanding of a human observer. Thus, they are usually more qualitative rather than quantitative in nature, although examples of quantitative models exist (such as Hofstede's cultural factors [10]).

The work described in this paper is aligned with an ongoing effort of the autonomous agents and artificial intelligence communities to develop *operational models* of human social behavior. These research efforts are directed both towards a better understanding of social behavior in general, and towards the practical goal of improving the ability of software agents and autonomous robots to act in the presence of humans.

These efforts can be divided in two large categories. One category involves the study of large societies of humans, either being physical crowds or large organizations Bonabeau[3]. This type of study deals with emergent patterns over large number of interactions, and in recent years, had made significant progress through the data mining of information acquired from social networks, such as Twitter, Facebook, Sina Weibo and others.

The literature being very large, we can only consider several representative examples. Kottonau and Pahl-Wostl [11] studied the evolution of political attitudes in response to political campaigns - while in earlier work they studied the problem of new product diffusion. C. Motani et al. [16] implemented a virtual wireless social network based on the information spread in real social network such as a marketplace. Gruhl et al. [7] and Adar et al. [1] analyzed person-to-person information flow over blog space topic sharing. Recent analysis of Twitter followers by Cha et al. [5] had shown that the influence of user on the topic can be gained by a concerted effort over a long period of time and a large number of followers are not an assurance to fame.

A significant amount of research had been directed towards the epidemic propagation of information in social networks [18, 12, 13]. In these papers, the information spread is modeled as virus infection in computer networks.

The second category of social modeling involves singular interactions, but a higher detail model. An example of argument towards such models is the KIDS (Keep it Descriptive Stupid) approach advocated by Edmonds and Moss [6]. The work we describe here deals with this type of interactions.

An example of similar work involves the work of Miller et al. In [14] propose to operationalize the Brown and Levinson politeness model [4]. The implementation, the Etiquette Engine, is used to assess the politeness of a number of custom crafted social-interaction vignettes involving common culture but different rank (the interaction between a corporal and a mayor). The values were compared against the evaluation by human observers (unfamiliar with the Brown and Levinson model). In a follow-up work [15] they create a more complex model which investigates the relationship between culture (as examplified by Hofstede's cultural factors [10]) as well as politeness levels affect directive compliance. Directive compliance - the way in which people react to instructions, commands or requests, represent a very large proportion of human interactions in work and military settings.

The HAIDM workshop had been the venue of the presentation of a number of contributions in this area. Ramchurn et al. [20] deals with the representation of social and ethical issues in applications such as agile teaming, incentive engineering and flexible autonomy. Haim et al. [8] develop models of human behavior in the setting of a negotiation game with participants from different backgrounds (US, Lebanon and Israel), and develop an agent behavior called PAL (personality adaptive learning) which predicts the human adversaries behavior in a probabilistic manner. Salvit and Sklar [21] model human personality traits using the Myers-Briggs model, and integrate it in a BDI agent architecture.

Harriott et al. [9] describes an empirical study of human-robot teams using performance moderator functions (HPMF) which predict performance under various factors such as fatigue, mental workload or temperature.

Aylett et al. [2] describes a believable agent-based educational application based on emotions, personality and culture, where an agent is provided with a description of the symbols and rituals of a certain society. The application implements virtual reality characters of a fictional alien culture, while the users must adapt to their cultural behavior.

7 Conclusions

In this paper we described a modeling framework for reasoning about the social-cultural behavior of humans. The framework is currently implemented as a Java software library, and is a basis of our work in designing behaviors for autonomous robots which need to act in social settings. Our ongoing work is directed towards the modeling of real-life scenarios of robot deployments in various social settings, such as BigDog [19] class robots assisting a peacekeeping team in a foreign country. We are considering both scenarios where the objective is for the robot must act in socially acceptable ways, and scenarios where the objective is to pursue specified goals while considering the predicted social behavior of human interaction partners and bystanders.

Acknowledgements: The research reported in this document/presentation was performed in connection with Contract Number W911NF-10-2-0016 with the U.S. Army Research Laboratory. The views and conclusions contained in this document/presentation are those of the authors and should not be interpreted as presenting the official policies or position, either expressed or implied, of the U.S. Army Research Laboratory, or the U.S. Government, unless so designated by other authorized documents.

References

1. E. Adar and L. Adamic. Tracking information epidemics in blogspace. In *Proc. of 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 207–214. IEEE, 2005.
2. R. Aylett, N. Vannini, E. Andre, A. Paiva, S. Enz, and L. Hall. But that was in another country: agents and intercultural empathy. In *Proc. of the 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS-2009)*, pages 329–336, 2009.
3. E. Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences of the United States of America*, 99(Suppl 3):7280, 2002.
4. P. Brown and S. Levinson. *Politeness: Some universals in language usage*, volume 4. Cambridge Univ Pr, 1987.
5. M. Cha, H. Haddadi, F. Benevenuto, and K. Gummad. Measuring user influence on twitter: The million follower fallacy. In *Proc. of the Fourth Int'l AAAI Conference on Weblogs and Social Media*, pages 10–17. ACM, 2010.

6. B. Edmonds and S. Moss. From KISS to KIDS–an 'anti-simplistic' modelling approach. *Multi-Agent and Multi-Agent-Based Simulation*, pages 130–144, 2005.
7. D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins. Information diffusion through blogspace. In *Proc. 13th Internat. World-Wide Web Conference*. ACM New York, 2004.
8. G. Haim, Y. Gal, M. Gelfand, and S. Kraus. A cultural sensitive agent for human-computer negotiation. In *Proc. of the Eleventh Int. Conf. on Autonomous Agents and Multiagent Systems(AAMAS-2012)*, pages 451–458, 2012.
9. C. Harriott, R. Zhuang, J. Adams, and S. DeLoach. Towards using human performance moderator functions in human-robot teams. In *Proc. of the Human-Agent Interaction Design and Models Workshop*, 2012.
10. G. Hofstede, G. Hofstede, and M. Minkov. *Cultures and organizations: software for the mind*. McGraw-Hill Professional, 2010.
11. J. Kottonau and C. Pahl-Wostl. Simulating political attitudes and voting behavior. *Journal of Artificial Societies and Social Simulation*, 7(4), 2004.
12. Z. Liu and B. Hu. Epidemic spreading in community networks. In *Europhys. Lett. 72 315*. epl, 2005.
13. R. May and A. L. Lloyd. Infection dynamics on scale-free networks. *PhysRevE*, 64(6):066112, 2001.
14. C. Miller, P. Wu, and H. Funk. A computational approach to etiquette: operationalizing Brown and Levinson's politeness model. *IEEE Intelligent Systems*, pages 28–35, 2008.
15. C. Miller, P. Wu, V. Vakili, T. Ott, and K. Smith. Culture, politeness and directive compliance. In *Proc. of the 5th International Conference on Universal Access in Human-Computer Interaction*, pages 568–577, July 2009.
16. M. Motani, V. Srinivasan, and P. Nuggehalli. PeopleNet: Engineering a wireless virtual social network. In *Proc. of ACM MobiCom 2005*. ACM, 2005.
17. R. Nisbett and D. Cohen. *Culture of honor: The psychology of violence in the South*. Westview Press, 1996.
18. R. Pastor-Satorras and A. Vespignani. Epidemic spreading in scale-free networks. *PhysRevLett*, 86(14):3200–3203, 2001.
19. M. Raibert, K. Blankespoor, G. Nelson, R. Playter, et al. Bigdog, the rough–terrain quadruped robot. In *International Conference of Automatic Control*, pages 10823–10825, 2008.
20. S. Ramchurn, N. Jennings, L. Moreau, S. J. Roberts, T. Rodden, and A. Rogers. ORCHID: Developing the science of human-agent collectives. In *Proc. of the Human-Agent Interaction Design and Models Workshop*, 2012.
21. J. Salvit and E. Sklar. Modulating agent behavior using human personality type. In *Proc. of the Human-Agent Interaction Design and Models Workshop*, 2012.
22. A. Tversky and D. Kahneman. Judgment under uncertainty: Heuristics and biases. *science*, 185(4157):1124–1131, 1974.

Design of a Team Programming Language with Markup for Operator Interaction

Nathan Brooks[1], Ewart de Visser[2], Timur Chabuk[2], Elan Freedy[2], and Paul Scerri[1]

[1] Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA
{nbb, pscerri}@cs.cmu.edu
[2] Perceptronics Solutions, Inc., Falls Church, VA 22042, USA
{edevisser, timc, elanf}@percsolutions.com

Abstract. This paper presents a team plan specification language that combines work in the creation of generic team plans and design of intelligent interfaces. Two motivations the language are (1) to combine inter-agent cooperation and operator interaction of complex behaviors into a single plan, and (2) to separate plan design and UI design such that they are created by application domain experts and human interaction experts, respectively. The result is a generic language for multi-robot plans that defines tasks to be performed, interactions for maintaining situational awareness, and mixed initiative reactions to operator workload. The development and execution of one of these plans for a multi robot team in shown for a simulation environment, highlighting the features of the play specification language which allow it to be used effectively for plan creation across multiple domains.

Keywords: Robotic agent languages and middleware for robot systems; Intelligence for human-robot interaction; Robot teams, multi-robot systems

1 Introduction

Many exciting, emerging applications of robots involve multiple robots working together on a complex task under the control of a single human. In such domains, the robots will need to execute sequences of activities in parallel with contingencies, runtime options, and other deviations from a simple linear progression of tasks. For a variety of reasons, from technical to ethical, an operator will need to be actively involved in supervising the robots executing the activities. The operator will be required for things like choosing options, approving actions, making go/no-go decisions, assisting robots in difficult positions and selecting strategies.

Substantial research effort has looked at the issues of how to have cooperative robots controlled by a single operator. Parasuraman[10], Arkin[4], Tambe[15], and others have shown the potential to have some representation of a team plan be invoked by a human and executed by the team. The infrastructure for executing plans can be created once and the team empowered to do different

Document approved for public release 88ABW-2012-6315 (3 Dec 2012)

things by developing new plans. Ziparo[18] and Xu[17] demonstrate the use of Petri Nets[14] to describe multi-robot plans without human interaction.

From the perspective of the human, researchers such as Cummings[1], Lewis[16], and Goodrich[5] have shown how intelligent graphical user interfaces (UIs) can dramatically increase a human's ability to understand and interact with robot teams. These interfaces utilize domain specific strategies for maintaining operator situational awareness and defining parameters for mixed initiative adjustable autonomy, allowing larger and more sophisticated teams to be used.

While these interfaces and others incorporate mixed initiative adjustable autonomy and adapt based on operator workload, they do not adapt based on what plans are being executed. This is undesirable in a number of situations, such as when different instances have different levels of importance and when different instances should have small variations in presentation, options, and default values. Consider the example interaction of an operator approving an autonomously generated path. In a domain with a highly mobile robot team approving each path would quickly overwhelm the operator, so by default the path should be assumed to be approved. However, the operator might want to approve or edit paths that are used to approach humans to ensure there is not a collision.

To address this, the presented language includes Situational Awareness and Mixed Initiative (SAMI) markup that allows the application domain experts designing the plans to provide details about what information should be presented to the human and how important it is. Interfaces implementing the markup language are plan-centric and can adjust their appearance and behavior based on this information so that the operator has the exact set of tools needed for the current environment. How the interface will interpret the SAMI language markups does not need to be known while plans are being designed. This common vocabulary allows for an interesting and effective division of labor that can take place independently. Domain experts in the deployment environment can create the SAMI plans and specify from experience exactly what information should be exchanged between the operator and the UI to maintain situational awareness and aid the operator. The task of designing the actual UI can be assigned to human factors experts, who do not have to worry about adding special interaction processes in the interface for each current team plan or adding in new ones as new plans are needed. This also allows for the UI components to be reused across multiple domains which use SAMI plans.

As with any flexible team plans, a sophisticated infrastructure is required to execute the plans at runtime. Unlike frameworks such as STEAM[15] which are completely distributed, the infrastructure for these SAMI plans is currently centralized due to initial interest in domains with platforms that have stable communications, such as UAVs. The designed architecture is highly extensible, allowing for services to be easily interchanged for use in different domains. The addition of SAMI markup and intelligent services to team plans makes them substantially more difficult to author correctly. Also presented is a GUI that assists in the specifications of the plans, including intelligent assistants that check certain technical properties of the plans.

Fig. 1. VBS2 simulation of pickup plan

To demonstrate the power of the approach, a complex cooperative mission executed by two UAVs in a high fidelity simulation (Figure 1) is shown in Section 6. The demonstrated plan includes service invocations, SAMI actions, multiple tasks, and parallelism.

2 Related Work

Automation of robotic systems is advancing rapidly, but this development is not quite at the point of fully automated systems. For now, human-robot teams will be mixed-initiative in which the two work as a team to achieve mission success [3][11]. Part of the reason why mixed-initiative is a popular strategy for current systems is that good mission planning is hard to automate. Human experts are better at constructing plans that are context specific. Previous work on developing structures for mixed-initiative planning have focused on either *explicit* human-driven planning languages or *implicit* intelligent interfaces planning approaches. Explicit automation is automation that can be modified by the user whereas implicit automation is automation invoked by the system itself. Many of these systems use proxies for in automated task allocation, agents that represent a specific robot and know its current commitments and capabilities.

Rather than having a fixed planning language, Miller & Parasuraman [10] propose an adaptable flexible delegation interfaces for supervisory control. Delegation interfaces involve the same sense of a supervisor working with a human subordinate. The difference is that the operator uses an interface that allows for high-level communication with the automation in a common language. This approach mimics the flexibility and natural delegation that occurs between humans. This flexible approach has been shown to be more effective in achieving optimal mission compared to a more rigid and fixed strategy [13].

Adaptive automation with intelligent interfaces anticipating the needs of users is the other side of the coin in this debate. Benefits of planning with adaptive automation include increased situation awareness, a more balanced mental workload for the operator [12], and more appropriate trust [6]. With the assistance of intelligent interfaces, operators are able to generate plans more quickly compared to paper planning approaches [2].

While all of these approaches are promising and solve some part of the planning process, none are fully integrated and comprehensive.

3 Example Model

Figure 2 shows a fully designed SAMI plan that makes use of resource allocation and path planning intelligent services, as well as SAMI markups. In this contrived scenario, two UAVs will work together to pick up a payload in low lighting by simultaneously executing two separate tasks, light and lift. To increase the viewability of the entire "pickup" plan in the space of this document, some simplifications and assumptions about the scenario are made. The *light* task requires a UAV capable of hovering with high powered lighting equipment. The plan assumes that the UAV is capable of initially spotting the payload if it is within reasonable distance, and will illuminate the payload until it has been "captured" by some entity. The *lift* task requires a UAV capable of hovering with an aerial crane. The plan assumes that the UAV is capable of capturing the payload if it is within reasonable distance and has adequate lighting. This section walks through the pickup plan at a high level and Section 6 will revisit the plan in greater detail.

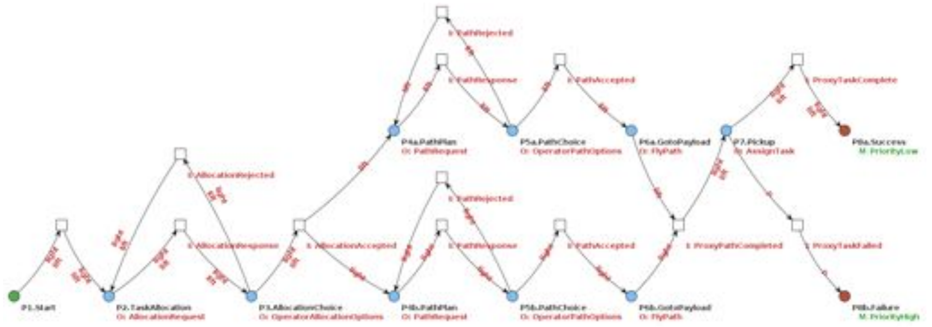


Fig. 2. A team plan with 2 UAV tasks

In the presented SAMI plans places are represented by circles and transitions are represented by squares. Red circles represent starting places for the plan and green circles represent ending places. Black text provides descriptions of elements in the plan, but does not affect functionality. Red text on a place or transition provide the name of output (with a prefix "O:") or input (with a prefix "I:") events attached to the respective vertex. Red text on an arc indicates the name of a token that is either a requirement (for incoming edges) or an output (for outgoing edges). Green text (with a prefix "M:") provides the name of a SAMI markup associated with the event directly above it. These elements will be described in detail in Section 4.

The play begins in *P1.Start* with a copy of each token in the system and transitions to *P2.TaskAllocation. P2* packages up information about the plan's tasks that need to be allocated and hands it off to the intelligent resource allocation service to compute several different solutions. In *P3.AllocationChoice* the operator is presented with the allocation solutions and must chose one or reject them all, which will cause a transition back to *P2* to compute new allocations. When the operator approves an allocation, *T3-4* branches the plan into two sections that will execute in parallel. The upper path is responsible for getting the proxy that was allocated the lifting task to the payload. The lower path does the same for the proxy with the lighting task. In *4a.PathPlan* the location of the lift proxy and the payload are sent to the intelligent path planning service to compute several unique paths to the payload. The operator chooses one of these paths in *5a.Path Choice* or rejects them and triggers a loop back to *4a* to replan. Once the path to the payload has been selected, *6a.GotoPayload* sends commands to the proxy to execute it. The SAMI plan will only enter *7.Pickup* when the proxies for both the lift and light tasks have arrived at the payload location. It then starts both the light and lift tasks. If one or more of the tasks fails, *P8b.Failure* becomes active and signals that abort recovery must take place. Abort recovery can be defined to do any number of actions ranging from prompting the user to abort the plan to autonomously reallocating the task whose proxy failed. This powerful and flexible mechanism is another facet of the SAMI language discussed in detail in Section 4. However, if no failures occur the plan will transition to *P8a.Success* once both proxies signal that they have finished their tasks. *8a* and *8b* are both end states that will mark the plan as being complete.

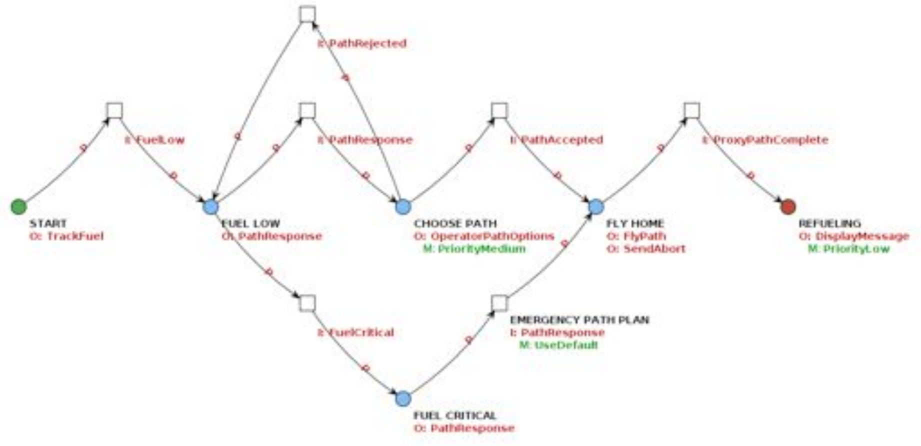


Fig. 3. A fuel contingency plan

4 Implementation

This section describes in detail the plan language representation, SAMI markup, and how plans are executed by the runtime engine.

4.1 Plan Language

Petri Nets consist of *places* connected to *transitions*, similar to states and transitions in state machines. Petri Nets also have a variable number of *tokens* that move between places when transitions fire; the token locations jointly represent the state. The *edges* between places and transitions specify the number of tokens that must be present in the place for the transition to trigger ("incoming" edges from a place to a transition) or will be put in the place when the transition triggers ("outgoing" edges from a transition to a place). Transitions with multiple incoming and outgoing edges and multiple tokens allows for more representational power than state machines and allows parallelism, synchronization and even counters to be implemented compactly. Below are descriptions of additional extensions.

Events: These Petri Net models interact with the environment through input and output events. Input events, received as a result of an action taken by some part of the system, can be placed on transitions as a trigger requirement, in addition to the token requirements on the incoming edges. Output events, which represent commands or requests, are added to places and are executed when new, relevant tokens enter the place.

Specialized Tokens: In traditional Petri Nets, all tokens are equivalent and anonymous. A useful extension is colored Petri Nets[8], which allow individual tokens to be associated with data useful to the plan. The SAMI plan formulation makes use of several specialized tokens which each contain different kinds of data. **Generic** tokens are traditional tokens and do not have unique identities or data attached to them. **Proxy** tokens contain a robot proxy's name and are used to invoke and receive environmental actions on a proxy basis. **Task** tokens contain a task's name and the proxy (if assigned) that will perform it and are used to invoke and receive environmental actions on a task basis. The task tokens in Figure 2, *lift* and *light*, contain knowledge about the tasks' requirements which is used by the resource allocation service invocation in the *Task Allocation* place. When the operator chooses to accept one of the returned allocation options in place *Operator Choice*, resulting in transition *Allocation Selected*, each allocated proxy is written to the matching task token. This is used in turn in the *Pickup* place, when the commands for each task are sent to the appropriate proxies.

Intelligent Service Invocations: An important part of the SAMI plan is the invocation of intelligent services to help the operator manage the robot team. The SAMI plan architecture allows services to have a generic interface with specific algorithm instantiations hidden from the plan, but chosen intelligently by the service provider depending on the circumstances. The presented plan makes use of two services, resource allocation and path planning. Output events that invoke services specify both the technical problem to be addressed, such as the unallocated tasks, as well as some information about what sort of solution the human will require at that particular time in the plan. For example, when initially allocating robots to tasks the event might request three alternative allocations and allow the allocation process to take more time to come up with higher quality solutions, while later in the plan when allocating a robot to a minor task the event might simply ask for a single, quick option. How the service interprets this request is up to the service.

Parameters: When a plan is being developed, there will typically be details for output events that are intended to be provided at runtime. For example, a plan for searching an area will need to know the area's dimensions, but defining those should occur when the operator starts the plan. *Parameters* allow for the design of a generic plan with the details filled in at runtime.

Variables: During the execution of a plan, team members may provide information needed by future events in the plan. For example, a robot might find a location of interest that should be given to a path planner. These are handled with *variables*. A variable allows the mapping of some component of an input event to some component of one or more output events.

Contingencies: Models based on Petri Nets can run multiple plans in parallel, allowing for contingency plans that recognize and react to unexpected behavior or variables beyond the scope of the currently executing plan. Typical problems which would be addressed by contingency plans include low power levels, sensor failures, and reacting to members of the team plan aborting their tasks. In team plans defined visually without parallelism, handling a single mode of failure leads to state explosion.

The fuel contingency plan in Figure 3 can take actions to prevent failures as well as initiating task aborts when a failure is identified. When a UAV's fuel level reaches an operator-defined "low", the operator is warned and prompted to approve one of several refueling plans to run after current tasks finish, preventing the UAV from aborting tasks in the future when the fuel level becomes critical. For this preventative measure, the contingency plan interacts with the operator, but sends no commands to the UAV without operator consent. However, if the operator is busy and does not respond before fuel reaches the "critical" level, the plan will immediately abort any other tasks held by the UAV then generate and execute a refueling task. This abortive action notifies the operator of its actions, but makes the decisions without operator input. If an abortive action is invoked, a "SendAbort" output event is generated, containing the identity of the proxy and the task it aborted. This will be received by any transition that triggers on AbortReceived input events, allowing for customized reactions.

The largest benefit of having explicit abort recovery actions is the customizability, which complements SAMI's goal of providing joint customization of team plans and UI interaction. Defining abort recovery on a case by case basis leverages having exact knowledge about the current state of the plan. Implicit handling has no contextual understanding of how critical a given failure is or how soon it should be resolved, which have a large effect on how the failure would be best addressed. However, explicit handling requires a very thorough examination of the state space to make sure every potential failure is handled. The plan development tool in Section 5 takes this into consideration, designing mechanisms that can aid in specifying abort recovery actions and catching unhandled failure modes. Given the graph structure of the SAMI plans, it is possible to create a single "default" recovery action, such as aborting an assigned task and reallocating it, and linking it to any unhanded failure modes discovered by the tool.

Figures 4 and 5 show examples of two different abort recovery methods. The plan in Figure 4 has an arbitrary nominal execution path *P1* through *P4*, with two tasks that branch into parallel paths and later merge. The plan diverges from nominal actions to the abort recovery section when an AbortReceived input event occurs and *P2b* contains the relevant proxy; that is, when an AbortReceived occurs that was caused by task2's proxy. The recovery plan branches to do two things: task2's token is sent to a GetRAs output event, which will result in the task being reallocated, and the proxy token for the task that was performing task2 is sent to a series of places that will have it fly back to a known base location. In this recovery plan, an interesting mechanism is required: retrieving a proxy token using a task token. This will be necessary in many recovery plans where the task is reallocated and the proxy that failed is assigned a new set of commands. The task2 token cannot be used to command the failed proxy to fly home as the proxy assigned to it may change in a race condition between the fly home and allocation assignment sections of the recovery plan.

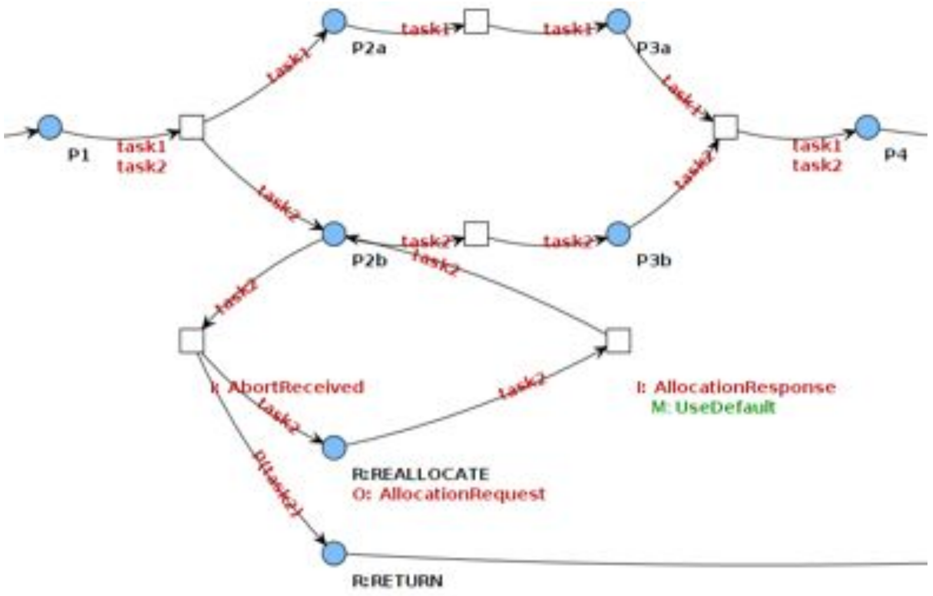


Fig. 4. Abort recovery by partial reallocation

Figure 5 revisits the arbitrary plan, but this time define the recovery plan for task2's proxy failing in *P3b*. In this case, the developer wants to reallocate both tasks and restart the plan from the beginning of the branch. The plan should move the task1 token from the upper branch, but does not know exactly what place it is in. This is solved with another special edge labeling, "ALL". If placed on an outgoing edge, a copy of every token taken from the transition's incoming edges will be put in the connected place. When placed on an incoming edge, all (if any) tokens present in the connected place to be removed when the transition finishes firing. Similar to the previous recovery plan, the failed proxy is sent back to base, but here the plan reallocates both task1 and task2 with input from the operator.

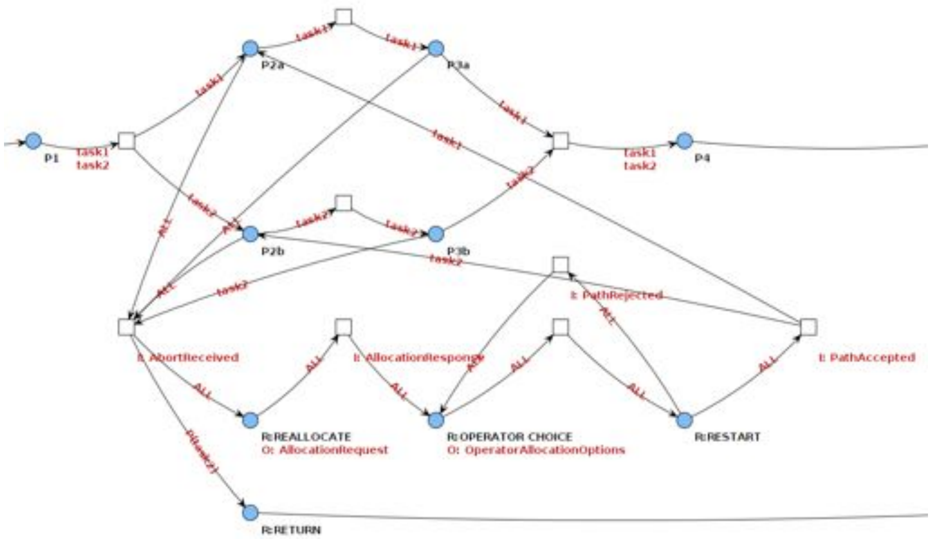


Fig. 5. Abort recovery by complete reallocation

Hierarchical composition: As evident from the preceding sections, team plans can be arbitrarily complex and may feature repetitive sets of actions. To reduce visual clutter and repetitive actions in plan creation, plans can be inserted into other plans as "subplans," similar to a hierarchical Petri Net. A potential hazard this introduces that must be addressed is reuse of predefined parameters and variables if the same play is inserted twice. Hierarchical composition also lends itself well to borrowing features from state charts[7], allowing "phases" of a team plan to share or intelligently duplicate abort recovery mechanisms to further reduce clutter.

4.2 SAMI Markup

To allow interfaces to dynamically adjust to plans, several types of SAMI markup can be added to the plans at development time. Markups are attached to individual events, and like events, can have a number of user-defined values. An important concept regarding SAMI compatible interfaces and plan markup is that they can be developed independently given the markup language. This means the domain expert designing the plans does not need to know anything about the interface except that it can interpret SAMI markup.

Information Markup: The first type of information markup tells the UI about environmental events and robot actions that the operator should be aware of. The second type of information markup tells the UI about the progress of the overall plan, e.g., phases of the plan or that a contingency is being invoked. These messages allow the UI to keep the human aware of the progress of the plan. Each piece of information markup can also be associated with specific objects or areas in the environment to allow the UI to be intelligent about the presentation of the messages.

Directives Markup: Directives can give more general situational awareness help than information messages. For example, a plan developer might understand that it will be useful for the human to have a zoomed out map view or a particular robot's video feed at a particular point in the plan. Allowing the designer to embed this sort of information in the plan allows the interface to help the human maintain situational awareness, a critical aspect of effective control[9].

Options Markup: Options markup can modify the set of options presented to the human and specify *mixed-initiative* parameters that are suited to that particular point in the plan. The first type of mixed-initiative parameters controls the number of options requested from services, with the amount chosen by the plan developer to suit the situation and the likely state of the human at that point in time. The options markup can also tell the UI how to handle taking autonomous action, such as giving the human a period of time to make a decision before selecting an option.

Importance Markup: The importance markup is used with any of the messages to the UI to describe the importance of the message relative to other things. The importance markup uses a temporal function to indicate to the UI how the importance will change over time. The markup can specify a set of assets for which it applies, so that when the human is micro-managing a robot, only the messages important to that robot can be displayed.

4.3 Execution Engine

A SAMI plan created by a developer will use some or all of the above features and to execute the plan the program must understand how all of the pieces connect together according to the SAMI specification language. This includes handling Petri Net token movement, how to handle output events with missing parameters, and what to do when input events arrive with variable values. Additionally, the execution engine provides a run-time visual of the Petri Net and the location of tokens to give the operator a view of the current state of the plan. Below is a brief outline of the key ideas of the execution engine.

Variable management: The input event fields contain information used by output events and must be accessibly stored for future reference. This is achieved with a plan-scoped lookup table, which uses a "variable name" string to retrieve the values of input event fields when the event occurs at run-time.

Parameter instantiation: Output events contain fields with information that will be used in service invocations or proxy commands, which can be acquired three different ways. A field in an output event can be defined when the plan is designed, when the plan is loaded at run-time, or can be linked to a variable that will be set at run-time by an input event. Output event fields linked to a variable are instantiated with the lookup table values when a token enters its associated place before the event is sent to its event handler.

Event handlers: The SAMI plan provides a domain specific event configuration file that specifies an event handler class to use for each supported output event. Those that invoke information services must also define service request handlers and service response listeners which remap the domain specific output event's fields into the generic format recognized by the service.

Event occurred listeners: Each active plan has a class that listens for when input events have occurred so it knows to retrieve the input event's field values. The event occurrence also triggers a check to see if the transition's requirements have now been met.

Handling tokens: In addition to input event requirements, transitions also have token requirements, which can include generic tokens, proxy tokens, and task tokens. Proxy tokens are complicated as plan developers do not know the specifics of the robot team that will be executing the plan. For example, when the fuel contingency plan starts, it has a proxy token for each UAV in the environment in the nominal state. When a low fuel message arrives with a specific UAV as its relevant proxy, the language must be able to specify that a proxy or task token for that UAV is an edge requirement. Furthermore, it needs to be able to specify that the proxy or task token that satisfied that requirement also get passed along specific outgoing edges when the transition triggers. To address this, the language uses the concept of "null" proxy tokens, which are indicated by a "P" in the sample SAMI models. They are used on incoming edges when a proxy or task token containing a specific proxy is an edge requirement, but the identity of that proxy is unknown until runtime and thus cannot be labeled in advance. Null proxy tokens are used in conjunction with input events that will have a relevant proxy, such as fuel messages. When a null proxy token is on an incoming edge of a transition that is being checked for trigger requirements, it checks that, for every input event with a relevant proxy, there is a token in the preceding place that contains that proxy. This requirement can be met by a proxy token or a task token. If a null proxy token is also present on a transition's outgoing edge, the tokens that satisfied the incoming edge's null proxy tokens are added to the place following the outgoing edge.

5 Specification Tool

The SAMI language is based on Petri Nets, which are designed and displayed graphically, but the language can be used to represent highly complex ideas and plans. To reconcile the attractiveness of graphical plan representation while maintaining readability during plan development, a mission specification tool was built, which is shown in Figure 6. It has a number of features which allow for organization of complex plans and intelligent assistants that ensure the plan is valid and complete.

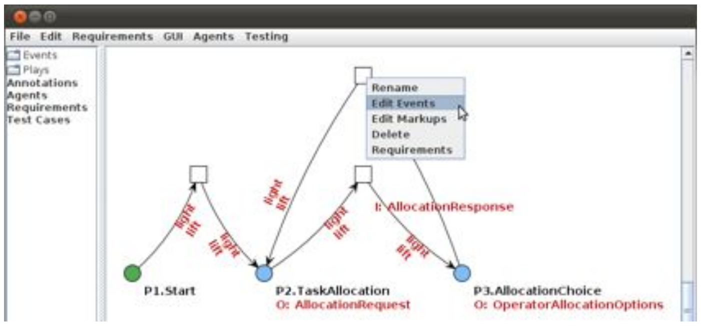


Fig. 6. Plan specification tool

Organization: As a developer may author plans for multiple domains, the specification tool and execution engine use the concept of a "library" to organize domain specific aspects of the SAMI architecture. On the developer side, a domain library contains only the input and output events, assets, and tasks that are relevant to that domain. Developed plans for an operator are also saved within the library so that, when using the execution engine, only plays applicable to the current domain are presented. As mentioned in Section 4, large plans or plans with repetitious portions can make use of "subplans", which a developer can "zoom" into to view only the subplan structure. Using aspects of state charts to allow for "grouping" of places and vertices that share some common actions, such as abort recovery, is a current topic of interest.

Intelligent Assistants: To further aid development, the specification tool makes use of two types of intelligent assistants: architecture assistants and domain assistants. *Architecture* assistants enforce rules of the Petri Net structure and can catch a number of errors including

- Illegal or missing start or end places
- Unconnected places or transitions
- Edges with unfulfillable token requirements
- Branches with common tokens that could cause race conditions
- Subplans that use a variable name in a parent plan

In some of these situations the assistants can suggest fixes or the developer can choose a template for the assistant to apply to fix the mistakes. *Domain* assistants warn the developer if there are contingency plans that are not being used or places in the plan missing abort recovery plans. One such assistant could allow the developer to select a place with an existing abort recovery mechanism and use it as a template for places without a mechanism.

6 Example Use

For initial tests in taking designed SAMI plans and executing them in an environment, the VBS2 high fidelity battlefield simulator was used. VBS2 features a powerful scripting language and large library of UAVs, ground vehicles, humans, and many other assets which make it an excellent platform for heterogeneous, multi-robot team simulation. The events, handlers, and services seen in Figures 2 and 3 were implemented to interface with the VBS2 simulation and the plans were run to verify that the language performed as intended in both an expected execution path resulting in success and a variety of paths resulting in failure. Before the plan begins, 4 UAVs are created in VBS2 and their proxies are registered with the SAMI engine. Thee operator then selects the pickup plan from the SAMI library using the “load” dialog and must fill in parameters which were not specified by the developer. For this plan, that consists of (1) the number of resource allocation alternatives to be presented, (2) the number of path planning alternatives to be presented, and (3) the location of the payload.

Once these items have been defined and saved to the relevant events, the play begins in *P1.Start* with one copy of each token that is registered in the system: the generic token, the proxy tokens for each of the 4 UAVs, and the task tokens for the light and lift tasks. This meets transition *T1-2*'s requirements and the light and lift task tokens are moved to *P2.Task Allocation*.

Now the engine enters *P2* and processes the get resource allocations (RAs) output event, GetRAs, whose only parameter is the number of return options that the operator specified when the plan was loaded. The RA handler is invoked with the output event and the list of tokens in *P2*, which begins to form the request that will be sent to the RA service. The service request requires a list of tasks to allocate and proxies to consider in addition to the number of options. The handler looks through the list of tokens it was invoked with and adds the tasks from any task tokens to the service request. The task and proxy objects referenced by the tokens contain knowledge about their requirements and capabilities, respectively. The handler then adds the list of system proxies registered with the engine to the service request along with the number of options and sends it to the service. Once the request has been sent, the input events on any transitions leading away from *P2* are registered with the event mapper and any needed event occurred listeners are created. When the RA server finishes generating the set of plans, they are received by the RA handler which proceeds to move the values into an RAsReceived event and send it to its information event handler. The event is received and sent to the event mapper, which stores the received values input event in the variables input event then sends the variables input event to the event occurred listener. The listener retrieves the stored values and writes them to the variable lookup table then triggers a transition check that results in *T2-3* executing.

The light and lift tokens from *P2* are moved into *P3* and the GetRAChoice event is processed. It requires a list of resource allocations, which was assigned the same variable name as the RAsReceived. The UI handler accepts the GetRAChoice event and begins to translate the data into a format recognized by the UI service. In this case it will use a service request that takes in a list of panels containing the allocations and allows one or none to be selected. The identity of the selected component or a rejection indicator is sent back in the service response and the UI handler generates either a RAAccepted or RARejected input event accordingly. If an allocation was accepted, the UI handler also assigns the allocation, and transition *T3-4* executes. Otherwise, transition *T3-2* executes, which removes the tokens from *P3* and places them back into *P2*, triggering the resource allocation process again.

T3-4 separates the plan into two parallel parts. In the case of the top half, the GetPaths with the payload location and desired number of options is sent to the path planning (PP) handler along with the lift token. The service request copies the payload location and option count, adds the current location of the proxy on the lift token in the simulation, and then sends the service request.

P6a.Goto Payload's FlyPath event is instantiated with the path from the variable the PathAccepted event's path was mapped to, and the proxy handler is invoked with the event and the lift token. The domain-specific handler sends the path to the lift token's proxy, and the robot begins to follow the path. When the path is complete, ProxyPathCompleted arrives with the lift token's proxy set as the relevant proxy. Here there is an interesting shortcut in the SAMI plan: two ProxyPathCompleted events, one for each task's proxy, have been merged into one. This is possible since the ProxyPathCompleted event is parameter-less, unlike the previous events which stored allocations or paths, and the only difference will be the proxy relevant to the event. So there are actually two ProxyPathCompleted events on *T6-7*, which will only trigger when an event for each of the incoming proxies has been received. This could be represented equivalently as seen in Figure 7, but adds clutter.

When the two task tokens arrive in *P7.Pickup*, they are both sent to the proxy handler with a parameter-less AssignTask event. This uses the same shortcut as in *T6-7*, but this time with a parameter-less output event. This sends the domain commands to each proxy which will result in the robot attempting to execute the task in the simulation. In the even of success, ProxyTaskCompleted is generated, otherwise a ProxyTaskFailed is generated.

Once again using the shortcut representation, the plan enters *P8a.Success* once ProxyTaskCompleted input events have arrived for both the lift proxy and light proxy. If one of the proxies fails, its task token is pulled from *P7* through the use of a null proxy token "P", which triggers *T7-8b*. The retrieved task token

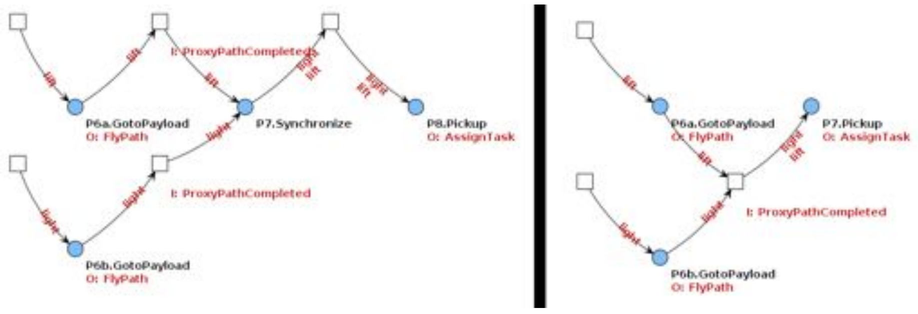


Fig. 7. Two equivalent SAMI statements

is sent to *P8b.Failure*, where it is used to generate a SendAbort output event and trigger abort handling actions.

7 Conclusions and Future Work

This work developed and implemented a novel, generic language for jointly specifying multi-robot coordinated tasks and operator interaction. In addition to allowing the design of team plans with specific instructions to maintain operator situational awareness and manage mixed initiative actions, the language makes it possible to separate the team plan development from the UI development, allowing experts in each of the domains to develop their portions of the system independently.

There are many areas for further research and in future work, such as conducting experiments to test the effectiveness of this system. One experimental design will analyze the usefulness of these plays with SAMI markups using participants as operators. A second experiment will test the specification tool's effectiveness by having participants develop plans. Performance areas of interest include ease of use, which is related to the amount of time and effort that is expended to fully develop a plan, and intuitiveness, which related to the degree of similarity between the user's desired run-time behavior the plan's actual behavior.

8 Acknowledgments

This work is funded by the Air Force Research Laboratory, Contract Number FA8650-11-C-6224

References

1. A. S. Macbeth J. C. Clare and M. L. Cummings. Mixed-initiative strategies for real-time scheduling of multiple unmanned vehicles. In *American Control Conf.*, 2012.

2. M.L. Cummings, Mariela Buchin, Geoffrey Carrigan, and Birsen Donmez. Supporting intelligent and trustworthy maritime path planning decisions. *International Journal of Human-Computer Studies*, 68(10):616–626, October 2010.
3. E. De Visser, R. Parasuraman, A. Freedy, E. Freedy, and G. Weltman. A comprehensive methodology for assessing human-robot team performance for use in training and simulation. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 50(25):2639–2643, 2006.
4. Y. Endo, D.C. MacKenzie, and R.C. Arkin. Usability evaluation of high-level user assistance for robot mission specification. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 34(2):168–180, 2004.
5. M.A. Goodrich, D.R. Olsen, J.W. Crandall, and T.J. Palmer. Experiments in adjustable autonomy. In *Proc. of IJCAI Workshop on Autonomy, Delegation and Control: Interacting with Intelligent Agents*, pages 1624–1629, 2001.
6. P.A. Hancock, D.R. Billings, K.E. Schaefer, J.Y.C. Chen, E.J. de Visser, and R. Parasuraman. A meta-analysis of factors affecting trust in human-robot interaction. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 53(5):517–527, 2011.
7. David Harel. Statecharts: a visual formalism for complex systems. *Science of Computer Programming*, 8(3):231–274, 1987.
8. Kurt Jensen. Coloured petri nets. *Petri nets: central models and their properties*, pages 248–299, 1987.
9. D. Kaber and M. Endsley. The effects of level of automation and adaptive automation on human performance, situation awareness and workload in a dynamic control task. *Theoretical Issues in Ergonomics Science*, 5(2):113–153, 2004.
10. CA Miller and R Parasuraman. Designing for flexible interaction between humans and automation: Delegation interfaces for supervisory control. *Human Factors*, 49(1):57–75, 2007.
11. R. Parasuraman. Designing automation for human use: empirical studies and quantitative models. *Ergonomics*, 43(7):931–951, 2000.
12. R. Parasuraman, K.A. Cosenzo, and EJ de Visser. Adaptive automation for human supervision of multiple uninhabited vehicles: Effects on change detection, situation awareness, and mental workload. *Military Psychology*, 21(2):270–297, 2009.
13. R. Parasuraman, S. Galster, and C. Miller. Human control of multiple robots in the roboflag simulation environment. In *Systems, Man and Cybernetics, 2003. IEEE Intl. Conf. on*, volume 4, pages 3232–3237 vol.4, oct. 2003.
14. James Lyle Peterson. Petri net theory and the modeling of systems. *PRENTICE-HALL, INC., ENGLEWOOD CLIFFS, NJ 07632, 1981, 290*, 1981.
15. Milind Tambe. Towards flexible teamwork. *arXiv preprint cs/9709101*, 1997.
16. H. Wang, A. Kolling, N. Brooks, S. Owens, S. Abedin, P. Scerri, P. Lee, S. Chien, M. Lewis, and K. Sycara. Scalable target detection for large robot teams. In *Proc. of the 6th Intl. Conf. on Human-robot Interaction*, HRI '11, pages 363–370, 2011.
17. D. Xu, R. Volz, T. Ioerger, and J. Yen. Modeling and verifying multi-agent behaviors using predicate/transition nets. In *ACM Intl. Conf. Proc. Series*, volume 27, pages 193–200, 2002.
18. V. A. Ziparo, L. Iocchi, D. Nardi, P. F. Palamara, and H. Costelha. Petri net plans: a formal model for representation and execution of multi-robot plans. In *Proc. of the 7th Intl. Joint Conf. on Autonomous Agents and Multi-Agent Systems*, AAMAS '08, pages 79–86, 2008.

Augie: A Dialogue-Augmenting Agent for Improved Health Care Communication

Roni Stern[1], Ofra Amir[1], Barbara J. Grosz[1], Shira H. Fischer[1], and Lee M. Sanders[2]

[1] Harvard University, Cambride MA 02138, USA,
roni.stern@gmail.com, ofra.amir@gmail.com, grosz@eecs.harvard.edu,
shira_fischer@bidmc.harvard.edu

[2] Center for Policy, Outcomes and Prevention, Stanford University, Stanford, CA 94305, USA
leesanders@stanford.edu

Abstract. Effective physician-patient communication is essential for successful health care, but has proved difficult to achieve. This paper describes a design for a novel type of personal-assistant software agent that intervenes appropriately to improve the effectiveness of such conversations. This "dialogue-augmenting agent" supports physicians and patients by listening to their conversation and identifying key concepts mentioned by the physician that may not be clear to the patient. It also tracks topics of conversation to make sure that the issues a patient intends to raise are eventually brought up in the discussion. The agent determines the value and timing of an interruption either to prompt the patient to ask a question or the physician to explain a concept further. If this value exceeds the cost of interruption, the agent augments the dialogue by sending a prompt to the appropriate dialogue participant. The paper defines the components of a dialogue-augmenting agent, identifies the key factors in computing the value of an interruption in this setting, and describes the dialogue capabilities needed to build such an agent.[3]

Keywords: Interruption management, Autonomous Agents, Dialogue systems

1 Introduction

Effective communication is crucial to successful collaboration [4]. A variety of studies in health care have established the need for improving the effectiveness of patient-physician communication [17]. Potentially serious effects of misunderstandings can occur for example when patients do not understand correctly medication prescriptions and thus fail to follow treatments as prescribed. The source of miscommunication vary: physicians may have inaccurate or incomplete

[3] This research is part of larger project that accords with the vision decried in "Collaborative Health Care Plan Support" which appears in the challenges and visions track of the technical program for this AAMAS conference.

models of each others' expertise or their familiarity with details of a particular case or of patients' health literacy. Communication difficulties may be exacerbated by patients' emotional states or health crises [11]

To help improve the quality of health care communications, we are developing a software agent, Augie, that is able to listen in on a dialogue, detect when parts of the dialogue may not be clear to a dialogue participant, and interrupt to address this potential misunderstanding. This paper considers the case of Augie agents accompanying patients to a doctor's appointment and helping the patient and provider understand each other better.

The design and implementation of Augies rests on several plausible simplifications. Such agents can be helpful even if they catch only some potential misunderstandings, without needing to understand all of a dialogue. They can improve over time, by learning about particular areas of health care and about particular patients and physicians. The design thus assumes Augies will track patient-physician dialogues listening for names of diseases, treatments and tests, using online medical ontologies, such as MeSH [13]. We will use the term *medical concepts* to refer to these. Augies will be helpful further by asking patients before appointments to list the issues they wish to address at the appointment, and then tracking those issues. Eventually, we plan for Augies to access health care "plans" and electronic health records (EHR) to help further in refining their recognition of important concepts to track. Therefore, implementing Augies requires only well constrained dialogue capabilities such as those already developed for capturing structured data from clinical encounters [10].

Building Augies is part of a broader project that aims to develop intelligent, autonomous multi-agent systems for supporting health care for children with complex conditions.[4] In this work we describe the architecture of an Augie and abstract away from the exact physical interface between an Augie and the person it supports. One can assume, as an example, an audio interface that allows Augie to speak to the patient it supports, not unlike a fellow human who whispers in a friend's ear.

The next section of the paper describes related work on which we will build. Subsequent sections present the Augie architecture, and the theoretical model of the interruption management component of Augie, building on our prior efforts in this area [9, 15, 8].

2 Related Work

A variety of prior work on human-computer dialogue and intelligent user interfaces is relevant to Augie. Dialogue systems research provides models of dialogue structure, coherence and intention recognition [6, 3, 1, inter alia] as well as techniques for modeling beliefs and intentions of other agents [19, 4, 5, 12, 16, inter alia]. Work in natural-language generation has developed techniques for producing text that is natural and appropriate in context [14, 7, inter alia].

[4] This project accords with the vision described in the paper Collaborative Health Care Plan Support in the challenges and visions track of the technical conference.

These prior efforts, however, have focused on developing models and techniques for an agent participating in a one-on-one conversation or a system performing offline analysis of a conversation [18]. Augies, are in essence "over hearers" of a conversation. As such secondary participants, they will need to reason about the dialogue and its participants from "outside", and from this perspective to track and model the beliefs and intentions of the dialogue participants as well as their interpersonal dynamic and past encounters. In addition, For instance, interrupting patients too soon may decrease their confidence or independent thinking about health care issues.

Prior work has explored the usefulness of a software agent analyzing and intervening in human interaction in a classroom educational setting [2]. In this work, a software agent called "Pierce" was intended to encourage collaboration among students working on a class assignment together. The students interacted with one another using a chat system, and Pierce intervened when it detected that the collaboration was not effective, encouraging a student to ask a question or asking a student about his thoughts. Various learning techniques were used to detect when a collaboration was not effective, using speech acts as the main features. Then, predefined rules were used to decide which collaboration encouraging action to perform. Furthermore, the students were only allowed to use specific keywords when starting a sentence, greatly simplifying the speech act recognition problem. There are several key differences between Pierce and the work discussed in this paper. Instead of using a predefined set of rules to determine which dialogue augmenting action to perform, we intend to use a more general decision theoretic approach, building on prior work on interruption management [9, 15, 8]. Furthermore, the patient-physician dialogue is not constrained to specific keywords. Lastly, we are aiming specifically for the health care domain, where the dialogue is not between peers, and there is more of a teacher-student relationship.

3 Augmented Dialogue Setting and Goals

To describe Augie's capabilities we will refer to the patient as Bob and to the physician as Alice. Bob has an Augie that accompanies him to Alice. The roles of Augie are as follows:

- Detect the topics and medical concepts discussed as the dialogue progresses.
- Assess whether Bob understands the medical concepts discussed.
- Verify that Bob understands or clarify to Bob a medical concept he needs to understand.
- Check that Bob does not forget to mention the topics he planned to raise.

To demonstrate how an Augie would act, consider the dialogue shown in Figure 1. It is an output of a prototype of the Intelligent Listening Framework (ILF), a system designed to "capture relevant data from a doctor-patient encounter into a well-structured encounter note" [10]. Consider first the content of the dialogue (ignoring the colored annotations). Many patients may not know

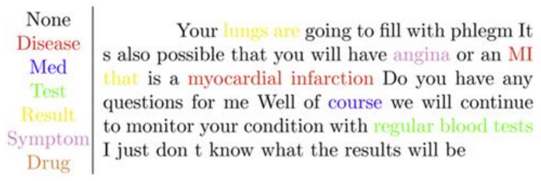

None
Disease
Med
Test
Result
Symptom
Drug
Your lungs are going to fill with phlegm It s also possible that you will have angina or an MI that is a myocardial infarction Do you have any questions for me Well of course we will continue to monitor your condition with regular blood tests I just don t know what the results will be

Fig. 1. An annotated dialogue output from the Intelligent listening framework [10]

that the term *MI* (myocardial infarction) refers to a condition more commonly known as "heart attack". Thus, an Augie might clarify to Bob that "MI means heart attack" or ask Bob "Do you understand what MI means?", implicitly suggesting that Bob asks Alice for clarifications about "MI" if it is not clear to him.

Note that by limiting the list of medical concepts that Augie will consider to a predefined set from a given knowledge base (e.g., MeSH [13]), we enable implementing Augies with considerably simplified NLP capabilities while still capturing an important part of patient-physician dialogue.

3.1 Dialogue Augmentation Actions

In this work we focus on the following three possible types of dialogue-augmenting actions:

1. **Clarify:** Presenting a short sentence that clarifies a medical concept that was discussed in the dialogue. For example, if Alice is talking about "tachycardia" then a "clarify" augmentation action could be to tell Bob, "tachycardia means high blood pressure".
2. **Verify:** Posing a question to Bob to verify that he understood a medical concept. This is intended to focus Bob as well as to raise the possibility of Bob asking Alice for more information. Using the same example as above, a "verify" action could be to ask Bob 'Do you know what is tachycardia?".
3. **Remind:** Reminding Bob about a topic that he planned to raise in the dialogue and has not done so. Augie enables Bob to input such a list of planned topics before the session with Alice starts. For example, if Bob planned to talk about his aching back, then Augie might tell Bob: "Have you talked to Alice about your back?".

4 Designing a Dialogue Augmenting Agent

The design architecture of Augie is shown in Figure 2. This design is composed of the following components: belief handler, dialogue analyzer, augmentation action generator, interruption manager and dialogue augmenter. We next describe each of these components and the interaction between them.

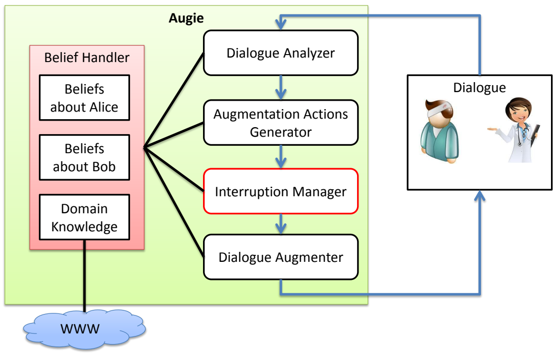


Fig. 2. Augie architecture and basic building blocks.

4.1 Belief Handler

Throughout the dialogue, Augie maintains three sets of **beliefs**: beliefs about what Bob understands, beliefs about what Alice is attempting to convey and its own beliefs in a domain knowledge base. Belief tracking of dialogue participants is known to be a challenge in dialogue systems [20]. We thus limit the required beliefs that need to be tracked for building an Augie as follows. The beliefs about Alice that Augie maintains are the medical concepts that Alice is mentioning. The beliefs about Bob that Augie maintains are estimates of Bob's understanding of the medical concepts raised by Alice. The domain knowledge beliefs are medical information, such as possible diagnoses and treatments for various conditions. This domain knowledge may be hard coded into Augie by its programmer or obtained automatically using information retrieval and extraction techniques from available resources, e.g., in the web. All of these beliefs initially come from a domain modeler or obtained automatically. These beliefs change over time given the medical concepts discussed in the dialogue.

4.2 Dialogue Analyzer and Action Generator

The **dialogue analyzer** component of Augie is responsible for listening to the dialogue and extracting the medical concepts and topics that were discussed. As mentioned above, only medical concepts from a given knowledge base are considered. Thus, implementing the dialogue analyzer does not require complete dialogue analysis, which is a very challenging AI problem. For example, using simple keyword spotting techniques might be sufficient to identify discussed topics and medical concepts. Also, the cost of misdetection in Augie is not high. The annotation created by ILF in the dialogue in Figure 1 can, for example, serve as a reasonable baseline for a dialogue analyzer. While not perfect, the annotations created by ILF in Figure 1 enabled detecting the medical concepts "MI", "Myocardial Infraction" and "regular blood tests" from a spoken dialogue.

The **augmentation actions generator** component is responsible for proposing dialogue augmentation actions. The **augmentation actions generator** will generate for each medical concept identified in the dialogue both a "Clarify" and a "Verify" action. In addition, a "Remind" action is generated for all the topics Bob gave to Augie before the dialogue started.

This set of generated actions are passed to the **interruption manager**, which performs a cost-benefit analysis to decide if one or more of the generated augmentation actions will be helpful. In the next section we describe the interruption manager component in greater detail. Augmentation actions chosen by the interruption manager to be performed are executed by the **dialogue augmenter** component, potentially helping Alice and Bob to better communicate with one another.

4.3 Interruption Manager

The interruption manager uses a cost-benefit analysis as in prior work on interruption management [9, 8, 15]. As described above, the "Clarify" and "Verify" augmentation actions are designed for a specific medical concept. We use the notation $MConcept(A)$ to denote the medical concept that an augmentation action A concerns. For example, if A is the action "Do you understand what MI means?", then $MConcept(A)$ would be "MI". Computing the value of performing A requires models of three measures related to $MConcept(A)$: *impact*, *current clarity* and *clarity after augmentation*. The *impact* of $MConcept(A)$ is the expected future benefit to Bob of $MConcept(A)$ being understood. The *current clarity* of $MConcept(A)$ is the probability that Bob understands this concept; it is affected by the agent's belief about Bob and the dialogue context. *Clarity after augmentation* is a measure of the expected clarity of $MConcept(A)$ after A is performed. In addition, each action is associated with a cost, denoted $cost(A)$, which represents the cost of interrupting the dialogue and performing A.

The impact, clarity and cost measures are combined to a single utility measure as follows:

$$\begin{aligned}&Utility(A) =\\&\quad impact(MConcept(A)) \cdot clarityAfter(A, MConcept(A))\\&- impact(MConcept(A)) \cdot clarity(MConcept(A))\\&- cost(A)\end{aligned}$$

The action with the highest utility is performed, if its utility is positive. Estimating the impact, clarity and cost measures is challenging, and we discuss next how we propose to do so in the first stage of this research project.

In our current implementation, "Remind" augmentation action will be executed at the end of the dialogue, reminding Bob about all the topics that was not mentioned in the dialogue.

5 Approach and Current Stage

Our approach for developing Augie is to implement and evaluate it in stages, starting with simulated patient-physician dialogue.

In the first stage, human subjects will take the role of the patient (Bob) and hear a doctor's monologue with and without Augie to help them. Impact values for the interruption manager will be set manually in consult with our collaborating physicians. Clarity of medical concepts mentioned in this monologue will be determined empirically, evaluating the clarity of every concept with a questionnaire that will be given to a group of subjects.

In the second stage, we will use the information gathered in the first stage to use a more general clarity measure, using Machine Learning to learn the relation between various features of the monologue and the clarity of the concepts that were discussed. Note that these features will include features used in previous work on discourse coherence [6, 1].

In the third stage, we will go beyond the monologue setting, having a human subjects also take the role of the physician (Alice). "Bob" and "Alice" will be given a script: "Bob" will be given a list of symptoms, and "Alice" will be given a diagnosis and proposed treatment. They will interact for a limited amount of time via a chat system, and the level of understanding "Bob" achieves will be evaluated with and without an Augie supporting it.

Finally, we intend to implement Augie in a clinical setting to test its helpfulness in reality. This will be done by continuing our current collaboration with several physicians and starting with a small group of patients of a specific type.

5.1 Expected Results to be Presented by the Workshop

We have begun implementing the first stage described above and expect to present results for the first three stages by the workshop.

6 Conclusion and Current Work

Dialogue is an important part of effective inter-human collaboration and in particular in health care. In this work we presented a design for Augie, an agent that listens to a dialogue between a patient and a physician and augments it. We describe a design for Augie, identifying the key building blocks, and described a theoretical model for the interruption management part of it.

Acknowledgments. The research reported in this paper was supported in part by grant IIS-0705406 from the U.S. National Science Foundation. We are grateful to Ece Kamar and Stuart M. Shieber for helpful discussions on this work.

References

1. R. Barzilay and M. Lapata. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34, 2008.
2. B. Goodman, F. Linton, G. Zarrela, and R. Gaimari. Using Machine Learning to Predict Trouble During Collaborative Learning. *User Modeling and User-Adapted Interaction*, 15:85–134, 2005.

3. D. Griol, L.F. Hurtado, E. Segarra, and E. Sanchis. A statistical approach to spoken dialog systems design and evaluation. *Speech Communication*, 50:666 – 682, 2008.
4. B.J. Grosz and S. Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86(2):269–357, 1996.
5. B.J. Grosz and S. Kraus. The evolution of SharedPlans. *Foundations of Rational Agency*, 14:227–262, 1999.
6. B.J. Grosz, S. Weinstein, and A.K. Joshi. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225, 1995.
7. J. Hunter, Y. Freer, A. Gatt, E. Reiter, S. Sripada, C. Sykes, and D. Westwater. BT-Nurse: computer generation of natural language shift summaries from complex heterogeneous medical data. *Journal of the American Medical Informatics Association*, 18(5):621–624, 2011.
8. E. Kamar, Y.K. Gal, and B.J. Grosz. Modeling information exchange opportunities for effective human-computer teamwork. *Artificial Intelligence*, 2012.
9. Ece Kamar, Ya'akov Gal, and Barbara J. Grosz. Incorporating helpful behavior into collaborative planning. In *AAMAS*, pages 875–882, 2009.
10. J.G. Klann and P. Szolovits. An intelligent listening framework for capturing encounter notes from a doctor-patient dialog. *BMC medical informatics and decision making*, 9(Suppl 1):S3, 2009.
11. Gavin Lloyd, Dave Skarratts, Neil Robinson, and Cliff Reid. Communication skills training for emergency department senior house officers a qualitative study. *J. Accid. Emerg. Med.*, 17:246–250, 2000.
12. Y. Ma, A. Raux, D. Ramachandran, and R. Gupta. Landmark-based location belief tracking in a spoken dialog system. In *SIGDIAL Conference*, pages 169–178, 2012.
13. National Library of Medicine (NLM). MeSH (Medical Subject Headings). http://www.ncbi.nlm.nih.gov/mesh, 2012.
14. O. Rambow, S. Bangalore, and M. Walker. Natural language generation in dialog systems. In *Proceedings of the International Conference on Human Language Technology Research*, HLT, pages 1–4, 2001.
15. D. Sarne and B. J. Grosz. Determining the Value of Information for Collaborative Multi-Agent Planning. *Journal of Autonomous Agents and Multi-Agent Systems*, To appear 2012-2013.
16. E. Selfridge, I. Arizmendi, P.A. Heeman, and J.D. Williams. Integrating incremental speech recognition and POMDP-based dialogue systems. In *SIGDIAL Conference*, pages 275–279, 2012.
17. John M. Travaline, Robert Ruchinskas, and Gilbert E. D'Alonzo. Patient-physician communication: Why and how. *J. of the American Osteopathic Association (JAOA)*, 105(1), 2005.
18. G. Tür, A. Stolcke, L.L. Voss, S. Peters, D. Hakkani-Tür, J. Dowding, B. Favre, R. Fernández, M. Frampton, M.W. Frandsen, C. Frederickson, M. Graciarena, D. Kintzing, K. Leveque, S. Mason, J. Niekrasz, M. Purver, K. Riedhammer, E. Shriberg, J. Tien, D. Vergyri, and F. Yang. The CALO meeting assistant system. *IEEE Transactions on Audio, Speech & Language Processing*, 18(6):1601–1611, 2010.
19. F. Wang and K. Swegles. Modeling user behavior online for disambiguating user input in a spoken dialogue system. *Speech Communication*, 55(1):84 – 98, 2013.
20. Jason D. Williams. Challenges and opportunities for state tracking in statistical spoken dialog systems: Results from two public deployments. *J. Sel. Topics Signal Processing*, 6(8):959–970, 2012.

The Power of Modeling Human Adversary Behaviors in Security Games

Thanh H. Nguyen[1], Rong Yang[1], Amos Azaria[2], Sarit Kraus[2], Milind Tambe[1]

[1]University of Southern California, Los Angeles, CA 90089
{thanhhng, yangrong, tambe}@usc.edu
[2]Bar-Ilan University, Ramat Gan 52900, Israel
{azariaa1, sarit}@cs.biu.ac.il

Abstract. Recent deployments of Stackelberg security games (SSG) have led to two competing approaches to handle boundedly rational human adversaries: (1) integrating models of human (adversary) decision-making into the game-theoretic algorithms, and (2) applying robust optimization techniques that avoid adversary modeling. A recent algorithm (MATCH) based on the second approach was shown to outperform the leading modeling-based algorithm even in the presence of significant amount of data. Is there then any value in using human behavior models in solving SSGs? Through extensive experiments with 547 human subjects playing 11102 games in total, we emphatically answer the question in the affirmative, while providing the following key contributions: (i) we show that our algorithm, SU-BRQR, based on a novel integration of human behavior model with the subjective utility function, significantly outperforms both MATCH and its improvements; (ii) we are the first to present experimental results with security intelligence experts, and find that even though the experts are more rational than the Amazon Turk workers, SU-BRQR still outperforms an approach assuming perfect rationality (and to a more limited extent MATCH); (iii) we show the advantage of SU-BRQR in a new, large game setting and demonstrate that sufficient data enables it to improve its performance over MATCH.

Keywords: Game Theory, Human Behavior, Quantal Response, Robust Optimization, Bounded Rationality, Decision-making

1 Introduction

The recent multiple deployments of Stackelberg Security Games (SSG) assist security agencies ("defenders") to optimally allocate their limited resources against human adversaries [20,1,12]. While these deployments have often assumed that the adversary is a perfectly rational player, who maximizes expected value, it is well understood that such an assumption is not ideal for addressing human adversaries [2]. As a result, researchers have been pursuing alternative approaches to handle adversary's bounded rationality in SSGs [17,22,18].

Two competing approaches have emerged to address human bounded rationality in SSGs. One approach integrates models of human decision-making into algorithms for computing defender strategies; the other adopts robust optimization techniques to intentionally avoid adversary modeling. The BRQR algorithm [22], based on modeling adversary decision-making with the Quantal Response (QR) [14] model, leads to significantly better defender strategies than any previous leading contenders. However, the more recent robust algorithm MATCH [18] outperforms BRQR. It is indeed surprising that despite the long history of modeling success of QR, MATCH still performs better, even when significant amount of data were used to tune the key parameter in QR and no tuning was done to MATCH's key parameter.

Thus, there is now an important open question of whether there is any value in adversary modeling in SSGs. Our first contribution in answering this question builds on the significant support for QR [8,3]: we hypothesize that QR's stochastic response is crucial in building a human decision-making model. Where we part company with the original QR model however is in its assumption that human stochastic response is based on expected value. Instead, we propose a new model based on integration of a novel *subjective utility function* (SU) into QR, called the SUQR model. We show that the SUQR model, given learned parameters (from limited data), has superior predictive power compared to the QR model. We then derive the SU-BRQR algorithm, similar to BRQR, to compute the defender strategy assuming the adversary response follows the SUQR model. We evaluate SU-BRQR's performance by conducting two sets of experiments using an online game with Amazon Mechanical Turk (AMT) workers and show that: (i) SU-BRQR significantly outperforms MATCH in previously used settings; (ii) SU-BRQR usually outperforms (and always performs at least as well as) improved versions of MATCH such as ones offering it the same SU functions or tuning its key parameter.

SU-BRQR's parameters were learned from previously available (albeit limited) game data; we now test SU-BRQR in domains without the benefit of such *a-priori* data. Indeed, while some domains of SSG application, e.g., deterring fare evasion [23] or forest protection [10], could provide significant amounts of data to tune SU-BRQR, would we be better off with MATCH or other algorithms in applications that do not? Our second contribution answers this question by conducting experiments with a group of security intelligence experts, where we do not have any previous modeling data. These experts, who serve as proxies for real-world adversaries, serve in the best Israeli Intelligence Corps unit or are alumna of that unit, and are found to be more rational than the AMT workers. Against these experts, SU-BRQR with its earlier learned parameters, significantly outperforms both an algorithm assuming perfect adversary rationality [16] and (to a more limited extent) MATCH. Finally, our third contribution tests SU-BRQR in a new large game with AMT workers. We show that SU-BRQR with previously learned parameters still outperforms MATCH; and learning from more data, SU-BRQR performance can be further improved.

2 Background and Related Work

SSGs are defender-attacker games where the defender attempts to allocate her ("she" by convention) limited resources to protect a set of targets, and the adversary plans to attack one such target [4,20]. In SSGs, the defender first commits to a mixed strategy assuming that the adversary can observe that strategy. Then, the adversary takes his action.

Let T be the number of targets and K be the number of defender resources. The payoffs of both players depend on the attacked target and whether that target is covered by the defender. When the adversary attacks a target t, he will receive a reward R^a_t if the target is not covered by the defender; otherwise, he will receive a penalty P^a_t. In contrast, the defender will get a penalty P^d_t in the former case and a reward R^d_t in the latter case. We assume, as usual, $R^a_t, R^d_t > 0$ and $P^a_t, P^d_t < 0$. Let x_t be the coverage probability of the defender on target t. The defender's expected value at target t can be calculated as:

$$U^d_t = x_t R^d_t + (1 - x_t) P^d_t$$

Similarly, the expected value for the attacker is given by:

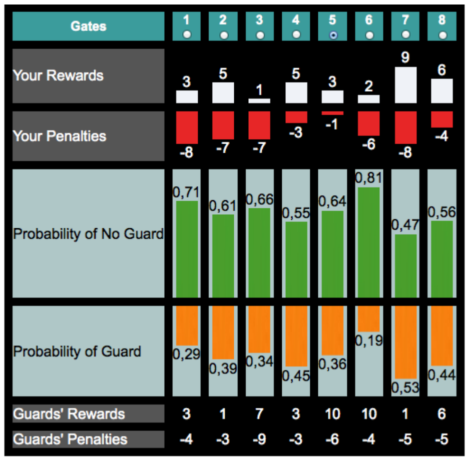
$$U^a_t = x_t P^a_t + (1 - x_t) R^a_t$$

Traditionally, the algorithms to compute the defender strategy in SSGs have assumed a perfectly rational adversary, who tries to maximize his expected value given the defender's strategy[4,16,11]. However, in real-world problems, the adversary's decision may be governed by his bounded rationality [13,5] due to effects such as task complexity and the interplay between emotion and cognition, which may cause him to deviate from the optimal action.

Recent research has therefore focused on developing algorithms to address the adversary's bounded rationality. In particular, BRQR[22] and MATCH[18] are the two leading contenders for handling adversary bounded rationality in SSGs. BRQR subscribes to modeling human decision making; it computes an optimal strategy for the defender assuming that the adversary's response follows the QR model. The QR model predicts a stochastic distribution of the adversary response: the greater the expected value of a target the more likely the adversary will attack that target. QR's key parameter λ (non-negative) represents the level of rationality in adversary's response: as λ increases, the predicted response by the QR model converges to the optimal action of the adversary. In contrast, instead of using a human behavior model, MATCH computes a robust defender strategy by guaranteeing a bound on the defender's loss in her expected value if the adversary deviates from his optimal choice. More specifically, the defender's loss is constrained to be no more than a factor of β times the adversary's loss in his expected value. The key parameter β describes how much the defender is willing to sacrifice when the adversary deviates from the optimal action.

A comparison of these two algorithms by [18], using over 100 payoff structures, showed that MATCH significantly outperforms BRQR. They also showed that even with sufficient data, with carefully re-estimated λ of the QR model, and no effort to estimate MATCH's β parameter, MATCH still outperformed BRQR.

A Simulated Security Game: A simulated online SSG, called "The guards and treasures" has previously been used as the platform for human subject experiments [22,18]. We will also use it in our experiments. The game is designed to simulate the security scenario at the LAX airport, which has eight terminals that can be targeted in an attack. Figure 1 shows the interface of the game.



Gates	1	2	3	4	5	6	7	8
Your Rewards	3	5	1	5	3	2	9	6
Your Penalties	-8	-7	-7	-3	-1	-6	-8	-4
Probability of No Guard	0,71	0,61	0,66	0,55	0,64	0,81	0,47	0,56
Probability of Guard	0,29	0,39	0,34	0,45	0,36	0,19	0,53	0,44
Guards' Rewards	3	1	7	3	10	10	1	6
Guards' Penalties	-4	-3	-9	-3	-6	-4	-5	-5

Fig. 1: Game Interface

Before playing the game, all the subjects are given detailed instructions about how to play. In each game, the subjects are asked to select one target to attack, given the following information: subject's reward and penalty at each target, the probability that a target will be covered by the guard, and the reward and penalty of the defender at each target (more details in [22]).

3 The SUQR Model

The key idea in subjective expected utility (SEU) as proposed in behavioral decision-making [19,6] is that individuals have their own evaluations of each alternative during decision-making. Recall that in an SSG, the information presented to the human subject for each choice includes: the marginal coverage on target t (x_t); the subject's reward and penalty (R_t^a,P_t^a); the defender's reward and penalty (R_t^d, P_t^d). Inspired by the idea of SEU, we propose a subjective utility function of the adversary for SSG as the following:

$$\hat{U}_t^a = w_1 x_t + w_2 R_t^a + w_3 P_t^a \tag{1}$$

The novelty of our subjective utility function is the linear combination of the values (rewards/penalty) and *probabilities*. (Note that we are modeling the decision-making of the general population not of each individual as we do not have sufficient data for each specific subject). While unconventional at first glance, as shown later, this model actually leads to higher prediction accuracy than the classic expected value function. A possible explanation for that is that humans might be driven by simple heuristics in their decision making. Other alternatives to this subjective utility function are feasible, e.g., including all the information presented to the subjects ($\hat{U}_t^a = w_1 x_t + w_2 R_t^a + w_3 P_t^a + w_4 R_t^d + w_5 P_t^d$), which we discuss later.

We modify the QR model by replacing the classic expected value function with the SU function, leading to the SUQR model. In the SUQR model, the probability that the adversary chooses target t, q_t, is given by:

$$q_t = \frac{e^{\lambda \hat{U}^a_t}}{\sum_{t'} e^{\lambda \hat{U}^a_{t'}}} = \frac{e^{\lambda(w_1 x_t + w_2 R^a_t + w_3 P^a_t)}}{\sum_{t'} e^{\lambda(w_1 x_{t'} + w_2 R^a_{t'} + w_3 P^a_{t'})}} \tag{2}$$

The problem of finding the optimal strategy for the defender can therefore be formulated as:

$$\max_x \sum_{t=1}^{T} \frac{e^{\lambda(w_1 x_t + w_2 R^a_t + w_3 P^a_t)}}{\sum_{t'} e^{\lambda(w_1 x_{t'} + w_2 R^a_{t'} + w_3 P^a_{t'})}} (x_t R^d_t + (1 - x_t) P^d_t)$$

$$\text{s.t.} \sum_{t=1}^{T} x_t \leq K, 0 \leq x_t \leq 1 \tag{3}$$

Here, the objective is to maximize the defender's expected value given that the adversary chooses to attack each target with a probability according to the SUQR model. Constraint (3) ensures that the coverage probabilities on all the targets satisfy the resource constraint. Given that this optimization problem is similar to BRQR we use the same approach as BRQR to solve it [22]. We refer the resulting algorithm as SU-BRQR.

Learning SUQR Parameters Without loss of generality, we set $\lambda = 1$. We employ Maximum Likelihood Estimation (MLE) [9] to learn the parameters (w_1, w_2, w_3). Given the defender strategy **x** and N samples of the players' choices, the log-likelihood of (w_1, w_2, w_3) is given by:

$$logL(w_1, w_2, w_3|x) = \textstyle\sum_{j=1}^{N} log[q_{t_j}(w_1, w_2, w_3)]$$

where t_j is the target that is chosen in sample j and q_{t_j} (w_1, w_2, w_3) is the probability that the adversary chooses the target t_j given the parameters (w_1, w_2, w_3). Let N_t be the number of subjects attacking target t. Then we have:

$$logL(w_1, w_2, w_3|x) = \textstyle\sum_{t=1}^{T} N_t log[q_t(w_1, w_2, w_3)]$$

Combining with equation (2),

$$\begin{aligned} logL(w_1, w_2, w_3|x) &= w_1(\textstyle\sum_{t=1}^{T} N_t x_t) + w_2(\textstyle\sum_{t=1}^{T} N_t R^a_t) \\ &+ w_3(\textstyle\sum_{t=1}^{T} N_t P^a_t) - N log(\textstyle\sum_{t=1}^{T} e^{w_1 x_t + w_2 R^a_t + w3 P^a_t}) \end{aligned}$$

$logL(w_1, w_2, w_3|x)$ can be shown to be a concave function: we can show that the Hessian matrix of $logL(w_1, w_2, w_3|x)$ is negative semi-definite. Thus, this function has an unique local maximum point and we can hence use a convex optimization solver to compute the optimal weights (w_1, w_2, w_3), e.g., *fmincon* in Matlab.

Prediction Accuracy of SUQR model As in some real-world security environments, we would want to learn parameters of our SUQR model based on limited data. To that end, we used the data of 5 payoff structures and 2 algorithms MATCH and BRQR (10 games in total) from [18] to learn the parameters of the new SU function and the alternatives. In total, 33 human subjects played these 10 games using the setting of 8-targets and 3-guards from our on-line game. The parameters that we learnt are:

Table 1: Prediction Accuracy

QR	3-parameter SUQR	5-parameter SUQR
8%	51%	44%

(w_1, w_2, w_3)=$(-9.85, 0.37, 0.15)$ for the 3-parameter SU function; and $(w_1, w_2, w_3, w_4, w_5)$ = $(-8.23, 0.28, 0.12, 0.07, 0.09)$ for the 5-parameter function.

We ran a Pearson's chi-squared goodness of fit test [7] in all the 100 payoff structures in [18] to evaluate the prediction accuracy of the two proposed models as well as the classic QR model. The test examines whether the predicted distribution of the players' choices fits the observation. We set $\lambda = .76$ for QR model, the same as what was learned in [22]. The percentages of the payoff structures that fit the predictions of the three models (with statistical significance level of $\alpha = 0.05$) are displayed in Table 1. The table clearly shows that the new SUQR model (with the SU function in Equation (1)) predicts the human behavior more accurately than the classic QR model. In addition, even with more parameters, the prediction accuracy of the 5-parameter SUQR model does not improve. Given this result, and our 3-parameter models demonstrated superiority (as we will show in the Experiments section), we leave efforts to further improve the SUQR model for future work.

4 Improving MATCH

Since SUQR better predicts the distribution of the subject's choices than the classic QR, and as shown later, SU-BRQR outperforms MATCH, it is natural to investigate the integration of the subjective utility function into MATCH. In particular, we replace the expected value of the adversary with subjective utility function. Therefore, the adversary's loss caused by his deviation from the optimal solution is measured with regard to the subjective utility function.

$$\max_{x,h,\eta,\gamma} \quad \gamma \tag{4}$$

$$\text{s.t.} \sum_{t\in T} x_t \leq K, 0 \leq x_t \leq 1, \qquad \forall t \tag{5}$$

$$\sum_{t\in T} h_t = 1, h_t \in \{0,1\}, \qquad \forall t \tag{6}$$

$$0 \leq \eta - (w_1 x_t + w_2 R_t^a + w_3 P_t^a) \leq M(1-h_t) \tag{7}$$

$$\gamma - (x_t R_t^d + (1-x_t)P_t^d) \leq M(1-h_t) \tag{8}$$

$$\gamma - (x_t R_t^d + (1-x_t)P_t^d) \leq$$
$$\beta \cdot (\eta - (w_1 x_t + w_2 R_t^a + w_3 P_t^a)), \qquad \forall t \tag{9}$$

We refer to this modified version as SU-MATCH, which is shown in Equation (4)-(9) where h_t represents the adversary's target choice, η represents the maximum subjective utility for the adversary, γ represents the expected value for the defender if the adversary responds optimally and M is a large constant.

Constraint (7) finds the optimal strategy (target) for the adversary. In constraint (8), the defender's expected value is computed when the attacker chooses his optimal strategy. The key idea of SU-MATCH is in constraint (9). It guarantees that the loss of the defender's expected value caused by adversary's deviation is no more than a factor of β times the loss of the adversary's subjective utility.

Selecting β for MATCH: In MATCH, the parameter β is the key that decides how much the defender is willing to lose if the adversary deviates from his optimal strategy. Pita et al. set β to 1.0, leaving its optimization for future work. In this section, we propose a method to estimate β based on the SUQR model.

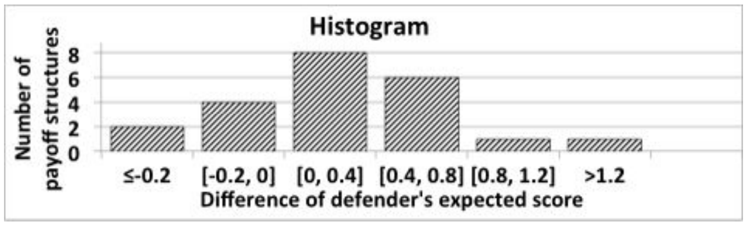
```
Initialize $\gamma^* \leftarrow -\infty$;
for i = 1 to N do
    $\beta \leftarrow Sample([0, \texttt{MaxBeta}], i)$, $x \leftarrow$ MATCH($\beta$);
    $\gamma \leftarrow \sum_t q_t U_t^d$;
    if $\gamma \geq \gamma^*$ then
        $\gamma^* \leftarrow \gamma, \beta^* \leftarrow \beta$;
return $(\beta^*, \gamma^*)$;
```

In this method, N values of β are uniformly sampled within the range (0, MaxBeta). For each sampled value of β, the optimal strategy x for the defender is computed using MATCH. Given this mixed strategy x, the defender's expected value, γ, is computed assuming that the adversary will respond stochastically according to the SUQR model. The β leading to the highest defender expected value is chosen. In practice, we set MaxBeta to 5, to provide an effective bound on the defender loss, given that penalties/rewards of both players range from -10 to 10; and N to 100, which gives a grid size of 0.05 for β for the range of $(0, 5)$. We refer to the algorithm with *carefully selected β as MATCHBeta*.

5 Experimental Results

5.1 Results with AMT Workers, 8-target Games

Our first experiment compares SU-BRQR against MATCH and its improvements, in the setting where we learned the parameters of the SUQR model, i.e., the 8-target and 3-guard game with the AMT workers. In this 8-target game setting, for each game, our reported average is over at least 45 human subjects. The experiments were conducted on the AMT system. When two algorithms are compared, we ensured that identical human subjects played both on the same payoff structures. Participants were paid a base amount of US \$1.00. In addition, each participant was given a bonus based on their performance in the games to motivate them. Similar to [18]'s work, we ensured that players were not choosing targets arbitrarily by having each participant play two extra trivial games (i.e., games in which there is a target with the highest adversary reward and lowest adversary penalty and lowest defender coverage probability). Players' results were removed if they did not choose that target.



(a) All comparison data

	SU-BRQR	Draw	MATCH
$\alpha = .05$	13	8	1

(b) Results with statistical significance

Fig. 2: SU-BRQR vs MATCH, AMT workers, 8 targets

We generated the payoff structures based on covariance games in GAMUT [15]. In covariance games, we can adjust the covariance value $r \in [-1, 1]$ to control the correlation between rewards of players. We first generate 1000 payoff structures with r ranging from -1 to 0 by 0.1 increments (100 payoff structures per value of r). Then, for each of the 11 r values, we select 2 payoff structures ensuring that the strategies generated by each candidate algorithm (e.g., SU-BRQR and versions of MATCH) are not similar to each. One of these two has the maximum and the other has the median sum of 1-norm distances between defender strategies generated by each pair of the algorithms. This leads to a total of 22 payoff structures. By selecting the payoffs in this way, we explore payoff structures with different levels of the 1-norm distance between generated strategies so as to obtain accurate evaluations with regard to performance of the tested algorithms. We evaluate the statistical significance of our results using the bootstrap-t method [21].

SU-BRQR vs MATCH This section evaluates the impact of the new subjective utility function via a head-to-head comparison between SU-BRQR and MATCH. In this initial test, the β parameter of MATCH was set to 1.0 as in [18]. Figure 2a first shows all available comparison results for completeness (without regard to statistical significance). More specifically, we show the histogram of the difference between SU-BRQR and MATCH in the average defender expected reward over all the choices of the participants. The x-axis shows the range of this difference in each bin and the y-axis displays the number of payoff structures (out of 22) that belong to each bin. For example, in the third bin from the left, the average defender expected value achieved by SU-BRQR is higher than that achieved by MATCH, and the difference ranges from 0 to 0.4. There are 8 payoffs that fall into this category. Overall, SU-BRQR achieves a higher average expected defender reward than MATCH in the 16 out of the 22 payoff structures.

In Figure 2b, the second column shows the number of payoffs where SU-BRQR outperforms MATCH with statistical significance ($\alpha = .05$). The number of payoff structures where MATCH is better than SU-BRQR with statistical significance is shown in the fourth column. In the 22 payoff structures, SU-BRQR outperforms MATCH 13 times with statistical significance while MATCH defeats SU-BRQR only once; in the

Table 2: Performance comparison, $\alpha = .05$

	SU-MATCH	MATCHBeta	SU-MATCHBeta
MATCH	3, 11	1, 6	1, 8
SU-BRQR	8, 2	8, 2	5, 3

remaining 8 cases, no statistical significance is obtained either way. This result stands in stark contrast to [18]'s result and directly answers the question we posed at the beginning of this paper: there is indeed value to integrating models of human decision making in computing defender strategies in SSGs, but use of SUQR rather than traditional QR models is crucial.

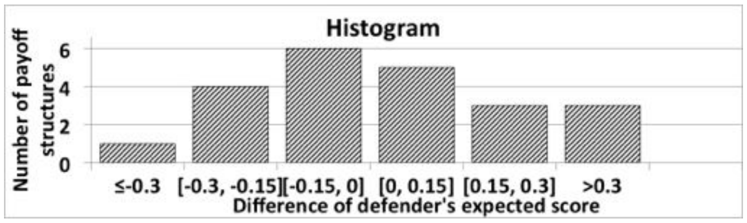
SU-BRQR vs Improved MATCH In Table 2, we compare MATCH and SU-BRQR against the three improved versions of MATCH: SU-MATCH, MATCHBeta, and SU-MATCHBeta (i.e., MATCH with both the subjective utility function and the selected β) when playing our 22 selected payoff structures. Here, we only report results that hold with statistical significance ($\alpha = .05$). The first number in each cell in Table 2 shows the number of payoffs (out of 22) where the row algorithm obtains a higher average defender expected reward than the column algorithm; the second number shows where the column algorithm outperforms the row algorithm. For example, the second row and second column shows that MATCH outperforms SU-MATCH in 3 payoff structures with statistical significance while SU-MATCH defeats MATCH in 11.

Table 2 shows that the newer versions of MATCH achieve a significant improvement over MATCH. Additionally, SU-BRQR retains a significant advantage over both SU-MATCH and MATCHBeta. For example, SU-BRQR defeats SU-MATCH in 8 out of the 22 payoff structures with statistical significance, as shown in Table 2; in contrast, SU-MATCH is better than SU-BRQR only twice.

Although SU-BRQR in this case does not outperform SU-MATCHBeta to the extent it does against MATCH (i.e., SU-BRQR performs better than SU-MATCHBeta only 5 times with statistical significance while SU-MATCHBeta is better than SU-BRQR thrice (Table 2)), SU-BRQR remains the algorithm of choice for the following reasons: (a) SU-BRQR does perform better than SU-MATCHBeta in more cases with statistical significance; (b) selecting the β parameters in SU-MATCHBeta can be a significant computational overhead for large games given that it requires testing many values of β. Thus, we could just prefer SU-BRQR .

6 Results with New Experimental Scenarios

All previous experiments are based on the 8-target and 3-guards game, which were motivated by the LAX security scenario [20]. In addition, the games have been played by AMT workers or college students. To evaluate the performance of the SUQR model in new scenarios, we introduce two new experimental settings: in one the experiments are conducted against a new type of human adversary, i.e., security intelligence experts; and in the other, we change the game to 24 targets and 9 guards.



(a) All comparison data

	SU-BRQR	Draw	MATCH
$\alpha = .05$	6	13	3

(b) Results with statistical significance

Fig. 3: SU-BRQR vs MATCH, security experts

6.1 Security Intelligence Experts, 8-target games

In this section, we evaluate our algorithm with security intelligence experts who serve in the best Israeli Intelligence Corps unit or are alumna of that unit. Our purpose is to examine whether SU-BRQR will work when we so radically change the subject population to security experts. We use the same 22 payoff structures and the same subjective utility function as in the previous experiment with AMT workers. Each result below is averaged over decisions of 27 experts.

SU-BRQR vs DOBSS DOBSS [16] is an algorithm for optimal defender strategies against perfectly rational opponents. DOBSS performed poorly in 8-target games against AMT workers[17,22]. However, would DOBSS perform better in comparison to SU-BRQR against security experts? Our results show that SU-BRQR is better than DOBSS in all 22 tested payoff structures; 19 times with statistical significance. Thus, even these experts did not respond optimally (as anticipated by DOBSS) against the defender's strategies.

SU-BRQR vs MATCH Figure 3a shows that SU-BRQR obtains a higher expected defender reward than MATCH in 11 payoff structures against our experts. Furthermore, SU-BRQR performs better than MATCH in 6 payoff structures with statistical significance while MATCH is better than SU-BRQR only in 3 payoff structures with statistical significance (Figure 3b). These results still favor SU-BRQR over MATCH, although not as much as when playing against AMT workers (as in Figure 2).

Nonetheless, what is crucially shown in this section is that changing the subject population to security experts does not undermine SU-BRQR completely; in fact, despite using parameters from AMT workers, SU-BRQR is still able to perform better than MATCH. We re-estimate the parameters (w_1, w_2, w_3) of the SU function using the data of experts. The result is: $w_1 = -11.0, w_2 = 0.54$, and $w_3 = 0.35$. This result shows that while the experts evaluated all the criteria differently from the AMT workers they gave the same importance level to the three parameters. Because of limited access to experts, we could not conduct experiments with these re-estimated parameters; we will show the impact of such re-estimation in our next experimental setting.

Bounded Rationality of Human Adversaries We now compare the AMT workers and security experts using the traditional metric of "rationality level" of the QR model. To that end, we revert to the QR-model with the expected value function to measure how close these players are to perfect rationality. In particular, we use QR's λ parameter as a criterion to measure their rationality. We use all the data from AMT workers as well as experts on the chosen 22 games in previous experiments to learn the λ parameter. We get $\lambda = 0.77$ with AMT workers and $\lambda = 0.91$ with experts. This result implies that security intelligence experts tend to be more rational than AMT workers (the higher the λ, the closer the players are to perfect rationality). Indeed, in 34 of 44 games, experts obtains a higher expected value than AMT workers. Out of these, their expected value is higher than AMT workers 9 times with statistical significance while AMT workers is higher only once ($\alpha = .05$). Nonetheless, the lambda for experts of 0.91 suggests that the experts do not play with perfect rationality (perfect rational $\lambda = \infty$).

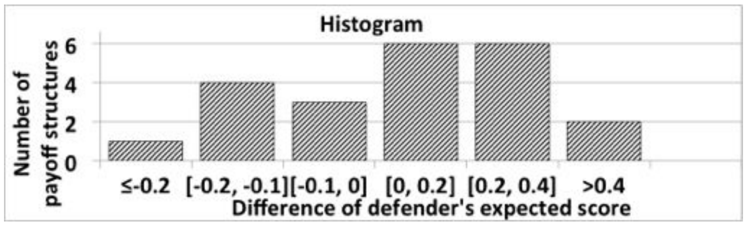
6.2 AMT Workers, 24-target Games

In this section, we focus on examining the performance of the algorithms in large games, i.e., 24 targets and 9 defender resources. We expect that the human adversaries may change their behaviors because of tedious evaluation of risk and benefit for each target. Three algorithms were tested: SU-BRQR, MATCH, and DOBSS. We first run experiments with the new subjective utility function learned previously using the data of the 8-target game.

SU-BRQR vs MATCH with Parameters Learned from the 8-target Games Figure 4a shows that SU-BRQR obtains a higher average defender expected value than MATCH in 14 out of 22 payoff structures while MATCH is better than SU-BRQR in 8 payoff structures. These averages are reported over 45 subjects. In addition, as can be seen in Figure 4b, SU-BRQR performs better than MATCH with statistical significance 8 times while MATCH outperforms SU-BRQR 3 times. While SU-BRQR does perform better than MATCH, its superiority over MATCH is not as much as it was in previous 8-target games.

We can hypothesize based on these results that the learned parameters of the 8-target games do not predict human behaviors as well in the 24-target games. Therefore, we re-estimate the values of the parameters of the subjective utility function using the data of the previous experiment in the 24-target games. The training data contains 388 data points. This re-estimating results in $w_1 = -15.29, w_2 = .53$, $w_3 = .34$. Similar to the experts case, the weights in 24-target games are different from the ones in 8-target games but their order of importance is the same.

SU-BRQR vs DOBSS with Re-estimated Parameters Since DOBSS has not been tested in the 24-target setting, we test it as a baseline. SU-BRQR outperforms DOBSS



(a) All comparison data

	SU-BRQR	Draw	MATCH
$\alpha = .05$	8	11	3

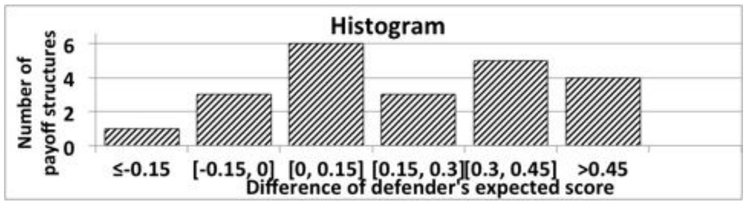
(b) Results with statistical significance

Fig. 4: SU-BRQR vs MATCH, 24 targets, original

with statistical significance in all 22 tested payoff structures illustrating the superiority of SU-BRQR over a perfectly rational baseline.

SU-BRQR vs MATCH with Re-estimated Parameters In this experiment, we evaluate the impact of the new subjective utility function with the re-estimated parameters on the performance of SU-BRQR in comparison with MATCH.

Figure 5a shows that SU-BRQR outperforms MATCH in 18 payoff structures while MATCH defeats SU-BRQR in only 4 payoff structures. Moreover, it can be seen in Figure 5b that SU-BRQR defeats MATCH with statistical significance 11 times while MATCH defeats SU-BRQR only once with statistical significance. In other words, the new weights of the subjective utility function indeed help improve the performance of SU-BRQR . This result demonstrates that a more accurate SU function can help improve SU-BRQR's performance.



(a) All comparison data

	SU-BRQR	Draw	MATCH
$\alpha = .05$	11	10	1

(b) Results with statistical significance

Fig. 5: SU-BRQR vs MATCH, 24 targets, re-estimated

7 Summary

This paper demonstrates the importance of integrating models of human decision making in computing defender strategies in SSGs using a novel *subjective utility function* (in contrast to the traditional computation of expected utility). The paper provides three groups of experiments in total involving 547 human subjects playing a total of 11102 games. These experiments result in the following contributions: (i) we show that our SU-BRQR algorithm, which involves a novel integration of QR behavior model with subjective utility function, significantly outperforms both MATCH and its improved versions; (ii) we are the first to present experimental results with security intelligence experts, and find that even though the experts are more rational than the Amazon Turk workers, SU-BRQR performs better than its competition against these experts; (iii) we show the advantage of SU-BRQR in a new, larger game setting (even with parameters from previous settings) and demonstrate that additional data can further boost the performance of SU-BRQR over MATCH.

8 Acknowledgement

This research was supported by MURI under the grant # W911NF-11-1-0332 and by the Google Inter-university center for Electronic Markets and Auctions, ARO under the grants # W911NF0910206 and # W911NF1110344.

References

1. N. Basilico, N. Gatti, and F. Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *AAMAS*, pages 57–64, 2009.
2. C. Camerer. *Behavioral game theory: Experiments in strategic interaction*. Princeton University Press, 2011.
3. S. Choi, D. Gale, and S. Kariv. Social learning in networks: a quantal response equilibrium analysis of experimental data. *Review of Economic Design*, pages 1–23, 2012.
4. V. Conitzer and T. Sandholm. Computing the optimal strategy to commit to. In *EC*, pages 82–90, 2006.
5. J. Conlisk. Why bounded rationality? *Journal of economic literature*, pages 669–700, 1996.
6. B. Fischhoff, B. Goitein, and Z. Shapira. Subjective utility function: A model of decision-making. *American Society of Information Science*, 32(5):391–399, 1981.
7. P. Greenwood and M. Nikulin. A guard to chi-squared testing, 1996.
8. P. A. Haile, A. Hortacsu, and G. Kosenok. On the empirical content of quantal response equilibrium. *The American Economic Review*, 98(1):180–200, 2008.
9. T. Hastie, R. Tibshirani, and J. Friedman. The elements of statistical learning 2nd edition, 2009.
10. M. Johnson, F. Fang, and M. Tambe. Patrol strategies to maximize pristine forest area. In *AAAI*, 2012.
11. D. Korzhyk, V. Conitzer, and R. Parr. Complexity of computing optimal stackelberg strategies in security resource allocation games. In *AAAI*, pages 805–810, 2010.
12. J. Letchford and Y. Vorobeychik. Computing randomized security strategies in networked domains. In *AARM Workshop In AAAI*, 2011.

13. J. March. Bounded rationality, ambiguity, and the engineering of choice. *The Bell Journal of Economics*, pages 587–608, 1978.
14. R. McKelvey and T. Palfrey. Quantal response equilibria for normal form games. *Games and economic behavior*, 10(1):6–38, 1995.
15. E. Nudelman, J. Wortman, Y. Shoham, and K. Leyton-Brown. Run the gamut: A comprehensive approach to evaluating game-theoretic algorithms. In *AAMAS*, pages 880–887, 2004.
16. P. Paruchuri, J. Pearce, J. Marecki, M. Tambe, F. Ordonez, and S. Kraus. Playing games for security: an efficient exact algorithm for solving bayesian stackelberg games. In *AAMAS*, pages 895–902, 2008.
17. J. Pita, M. Jain, M. Tambe, F. Ordóñez, and S. Kraus. Robust solutions to stackelberg games: Addressing bounded rationality and limited observations in human cognition. *Artificial Intelligence*, 174(15):1142–1171, 2010.
18. J. Pita, R. John, R. Maheswaran, M. Tambe, and S. Kraus. A robust approach to addressing human adversaries in security games. In *ECAI*, pages 660–665, 2012.
19. L. J. Savage. *The Foundations of Statistics*. Dover Publications, 1972.
20. M. Tambe. *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned*. Cambridge University Press, 2011.
21. R. Wilcox. *Applying contemporary statistical techniques*. Academic Press, 2002.
22. R. Yang, C. Kiekintveld, F. Ordonez, M. Tambe, and R. John. Improving resource allocation strategy against human adversaries in security games. In *IJCAI*, pages 458–464, 2011.
23. Z. Yin, A. Jiang, M. Johnson, M. Tambe, C. Kiekintveld, K. Leyton-Brown, T. Sandholm, and J. Sullivan. Trusts: Scheduling randomized patrols for fare inspection in transit systems. In *IAAI*, 2012.

Autonomous Agent for Deception Detection

Amos Azaria[1], Ariella Richardson[2], and Sarit Kraus[1,3]

[1] Department of Computer Science, Bar-Ilan University, Ramat Gan 52900, Israel
[2] Dept. of Industrial Engineering Jerusalem College of Technology, Israel
[3] Institute for Advanced Computer Studies University of Maryland, MD 20742

Abstract. Autonomous agents can be of assistance in detecting and reducing deception in computerized forums and chat-rooms. Although some deception detection methods exist they heavily rely on audio and visual information. Our focus is on text-based environments where there is no data of this type to use. We have developed DIG, an innovative machine learning-based autonomous agent, which joins a group of players as a regular member and assists them in catching a deceiver. We introduce "the pirate game" as a platform for deploying this agent. Our experimental study shows that although humans display difficulty detecting deception, DIG is not only capable of finding a deceptive player, but it also helps increase the entire group's success.

1 Introduction

Many activities in our everyday lives involve sharing opinions with peers through computer-mediated communication. People participate in forum discussions on important topics such as: how to raise their children, what medication they should use and how to improve their business. In web-based applications it is common practice to elicit information through structured questionnaires. It would be nice to assume that all the people participating in these discussions have a common, honest goal and that malicious participants are spotted by moderators. However, this is often not true. Pedophiles manage to infiltrate kids' chat rooms, commercial products are pushed in forums by dealers posing as regular users, and business forums are probably full of advice that actually assists their competitors. Computer-mediated communication has provided a modern venue for deception [16].

In his book "Telling Lies" [5], Ekman states that "It is not a simple matter to catch lies" (pg. 80). Ekman explains (pg. 82) that when people are dishonest they choose their words with care, however, they tend to give away the lie through body language and tone. This knowledge from Psychology has motivated studies on automatic detection of lies based on video and audio data [4,13,7,2].

In many situations, such as the chat rooms and forums we mentioned, there is no audio or visual information. Does that mean that impostors and liars can remain undetected? Newman et al. [11] and Toma and Hancock [15] found that certain words are chosen more often by liars. Zhou et al. use text-based computer mediated environments to study what linguistic based cues are used more in deceptive texts than in non-deceptive texts [17] and how modality affects human deception detection in chat-based environments [16]. However they do not actually use their findings to build a model for deception detection with machine learning.

In our work we investigate scenarios where several participants attempt to collectively detect a deceptive member. We also study how and if they may be assisted by an autonomous agent. Previous studies have explored deception detection in multi-agent environments [14] where agents must determine whether other agents are lying. Our research is different, as we deploy an agent that detects deception in human subjects. The agent joins a group as a regular member and uses machine learning to detect suspicious behavior. Although the agent has no enforcement capabilities in the forum, it participates in the group decision of who the liar is and is also able to raise the attention of other participants to malicious and dishonest activity. This is an important contribution as it implies that anyone can deploy an agent into a chat environment and use the agent to detect deception while the discussion is active.

We designed a game to deploy and evaluate our agent in a text-based environment. In this game there are several credible participants and one dishonest participant (a pirate). In the first phase of the game the participants conduct a textual discussion with an attempt to uncover the liar, and later they cast their votes as to whom they think is the liar. The game is played in two settings. One with a human set of participants, the other with a mixture of human participants and an autonomous agent that plays as a regular participant. We evaluate the contribution of the agent to the credible participants.

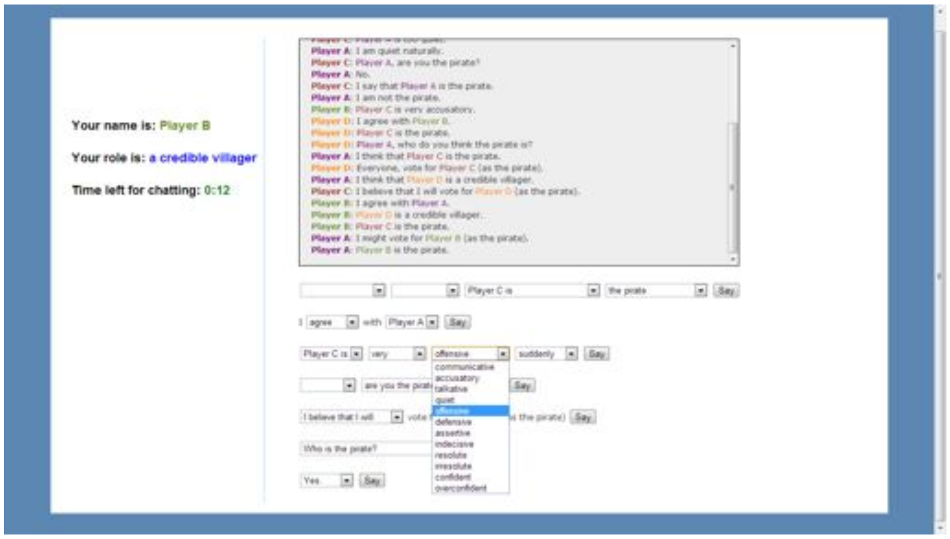
We use machine learning on the data collected from human participants to determine whether a player is honest or not. The agent uses this information to catch the pirate. We also apply machine learning methods in order to learn when players fall under suspicion. This information is also used by our agent in order to minimize the suspicion that it raises.

There are two versions to our game: the second version encourages the pirate to be more active than the first. This setting is similar to one where a salesman is pushing a commercial product. In both versions of the game, the credible participants did not perform better than chance at spotting the pirate. However, once one of the credible participants was replaced by the agent that we built, the group had significantly greater success in finding the pirate.

The two main contributions of this paper are that we provide a method for detecting deception within text-based chat environments, and that we manage to build an agent capable of assisting deception detecting within a human group.

2 The Pirate Game

We will now introduce the Pirate Game. There are four players and two roles in the game. Three players are the honest players, the "credible villagers", and the fourth player plays the deceptive participant, the "pirate". All players are informed of their own role but not of anyone else's. The participants are told that they are a group of villagers who went on a journey to find a treasure. They have found a treasure of coins and can split them. However, one of the participants is a pirate and can steal the coins unless he is detected. In order to detect the pirate a discussion phase is held. After the discussion, all credible villagers cast votes as to whom they think the pirate is (the votes are concealed until all players cast their votes). If there is a majority of votes for the pirate he is "caught" and the money is split between the credible players. Otherwise the pirate receives the full payoff. At the beginning of the game each player is told his role, and the discussion phase begins. The discussion is composed of structured sentences (examples are presented in Figure 1). The interface allows the composition of approximately $4,000$ sentences.



Your name is: Player B

Your role is: a credible villager

Time left for chatting: 0:12

Player A: I am quiet naturally.
Player C: Player A, are you the pirate?
Player A: No.
Player C: I say that Player A is the pirate.
Player A: I am not the pirate.
Player B: Player C is very accusatory.
Player D: I agree with Player B.
Player D: Player C is the pirate.
Player D: Player A, who do you think the pirate is?
Player A: I think that Player C is the pirate.
Player D: Everyone, vote for Player C (as the pirate).
Player A: I think that Player D is a credible villager.
Player C: I believe that I will vote for Player D (as the pirate).
Player B: I agree with Player A.
Player B: Player D is a credible villager.
Player B: Player C is the pirate.
Player A: I might vote for Player B (as the pirate).
Player A: Player B is the pirate.

Fig. 1. A screen-shot of the pirate game in progress

We use structured sentences rather than free text for several reasons. First of all the structured sentences encourage participants to use meaningful sentences. This also encourages players to speak and be active. We were able to easily categorize the sentences into different types for analysis without the need for incorporating NLP techniques that were not the focus of this study. Another helpful attribute of structured language is that differences in typing speed between players do not affect how suspicious players may seem to others. Finally, the controlled environment also makes it easy to add an agent that plays similarly to other players and keeps the agent inconspicuous. This does not mean that an agent cannot be deployed into an unstructured environment, but adding "social credentials" to the agent were not the focus of this work. We may investigate this in the future.

We also implemented a second variation of the game where the pirate is told that one of the credible villagers will turn him over to the village ruler if he escapes with the money. In this case neither the pirate nor the other players receive any payoff. However if the pirate is not caught and manages to convince the other players to cast at least one vote against this villager, he will be considered unreliable (to the village ruler) and the pirate will gain all of the coins. This setting encourages the pirate to be active in the game. We call this version of the game the "informer version" to differentiate it from the "basic version".

3 Formal Model

In the formal model the Pirate Game consists of k players $P = \{p_1, p_2, ..., p_k\}$, one of which is a pirate. We assume $k \geq 4$. The pirate player identity is only known to the pirate himself. The game includes two phases: the first phase consists of a communication phase where all players can discuss their strategy. The communication in this phase is cost-less and unbinding ("cheap talk" [6]). The second phase is a voting phase where all players except the pirate may simultaneously cast their vote $v(p)$ where $v : P \to \{\{p_1\}, \{p_2\}, ..., \{p_k\}, \phi\}$ and ϕ indicates an empty vote. The pirate is assumed to always cast an empty vote ϕ. If the majority of the votes are cast against the pirate, all players but the pirate receive a point and the pirate receives zero points. Otherwise the pirate receives a point and the other players receive zero points. More formally, we define a function that returns the player's role:

$$r(p) = \begin{cases} 0 & \text{if p is the pirate} \\ 1 & \text{else} \end{cases} \tag{1}$$

We will use $\overline{r(p)} = 1 - r(p)$. The reward function for each player $u(p)$ is given by:

$$u(p) \;=\; \overline{r(p)} \;+\; (\text{-}1)^{\overline{r(p)}} \cdot \mathbb{1}\Big\{\Big(\sum_{i=1}^{k} \sum_{p' \in v(p_i)} \overline{r(p')}\Big) \;\geq\; \Big\lceil \frac{1 + \sum_{i=1}^{k} |v(p_k)|}{2} \Big\rceil\Big\} \tag{2}$$

where $\mathbb{1}\{\}$ is the indicator function.

3.1 Equilibrium Strategies

Assuming all players are perfectly rational, the communication phase doesn't reveal any information on the pirate's identity, since the pirate may act as if he were a credible player.

Pure strategy equilibrium: Given a player p' the strategy $\forall p | r(p) = 1, v(p) = \{p'\}$ is in equilibrium, since any deviation from the equilibrium by a single player will not change the final result. Assuming p' is random, this strategy assures an expected utility of $\frac{1}{k}$ for the credible players (and $1 - \frac{1}{k}$ for the pirate). Although it is in equilibrium, agreeing upon the player to vote for (p'), requires the communication phase to allow simultaneous messaging. Assuming simultaneous messaging, all players must choose a random number $x(p)$ between 1 and k, simultaneously publish it, and then vote for player p_l where $l = \sum_{p \in P} x(p) \pmod{k}$. Although the pirate may choose a non-random number, since he has no knowledge of the numbers chosen by the other players, the result remains random. This method was also proposed in [3].

Mixed Equilibrium: Following is a mixed equilibrium for the game $\forall p | r(p) = 1, \forall p' | p' \neq p, v(p) = \{p'\}$ with probability $\frac{1}{k-1}$. The expected utility for the credible players using this equilibrium is given by the following binomial distribution mass function:

$$\sum_{j=\lceil \frac{k}{2} \rceil}^{k-1} \binom{j}{k-1} (\frac{1}{k-1})^j \cdot (1 - \frac{1}{k-1})^{k-1-j}$$

When $k = 4$, the above mixed equilibrium yields a slightly greater expected utility than the pure equilibrium mentioned before (0.26 vs. 0.25). Being symmetric towards all players, the mixed equilibrium doesn't require any prior communication. However, as k increases, the mixed equilibrium yields a very low expected utility for the credible players. For example, when $k = 7$ the expected utility of the credible players goes down to 0.01.

Proposition 1 *When $k = 4$ and the credible players' strategy is: $\forall p, v(p) = \phi$ with probability η and $\forall p' | p' \neq p, v(p) = \{p'\}$ with probability $\frac{1-\eta}{k-1}$; $\eta = 0$ yields the greatest expected utility.*

Proof. With a probability of $\binom{i}{k-1}\eta^i(1-\eta)^{k-1-i}$, i players will cast an empty vote. Denote by m the expected utility for all credible players when $\eta = 0$ and $k = 4$.

We therefore receive the following expected utility for the credible players:

$$\sum_{i=0}^{k-1}\binom{i}{k-1}\eta^i(1-\eta)^{k-1-i}\cdot$$
$$\sum_{j=\lceil\frac{k-i}{2}\rceil}^{k-1-i}\binom{j}{k-1-i}(\frac{1}{k-1})^j\cdot(1-\frac{1}{k-1})^{k-1-j}$$

Assigning $k = 4$ we get:

$$(1-\eta)^3\cdot m + 3\eta(1-\eta)^2\cdot\frac{1}{9} + 3\eta^2(1-\eta)\cdot\frac{1}{3} =$$
$$m - 0.45\eta + 1.12\eta^2 - 0.93\eta^3$$

$-0.45\eta + 1.12\eta^2 - 0.93\eta^3$ *is negative for any $0 < \eta \leq 1$ as it is continuous, has no root in $(0, 1)$ (its only real root is in 0) and is negative in 0.5.*

3.2 Alternative Models

Informer version: We also consider a model with a slight modification to the pirate's utility function, where there exists a player $\bar{p}$ whose identity is known only to the pirate. The pirate's utility function is identical to Equation 2, except if $\bar{p}$ doesn't receive even a single vote, then the pirate's utility is 0 regardless of the other votes. Formally, if $\sum_{i=1}^{k}\sum_{p'\in v(p_i)} r(p') = 0$, then the pirate's utility is 0. This change doesn't affect the equilibria, as they do not depend on the pirate's actions.

Voting pirate: One might consider an alternative to the given model by allowing the pirate to vote as well. This may seem more intuitive (or realistic), but significantly reduces the credible villagers' probability of catching the pirate which is already low. In addition to allowing the pirate to vote, one might also consider requiring that the pirate need only receive the plurality of votes (rather than the majority), where in case of a tie, the chosen player is defined by a toss of a coin. This drastically changes the equilibria for all games. As long as all players vote and no player votes for himself, almost any mixed strategy is in equilibrium with an expected utility of $\frac{1}{k}$ for the credible players (assuming the pirate may vote as well). This is due to the fact that in every game one and only one player is chosen, therefore, for symmetric reasons, there is a probability of exactly $\frac{1}{k}$ that this player is the pirate. We did not consider this model as a coin toss for tie breaking seemed unintuitive to the subjects.

4 Related Work

Of all existing games, the "Pirate Game" is most similar to the Mafia games (also known as "werewolf"). The Mafia games have been used as a platform for studying communication, coordination and functionality in situations where participants have different amounts of information, and have also raised much interest in theoretical analysis [9,3]. The pirate game is a simpler game which is closer to actual situations which take place in real life.

The mafia games have also been used for detecting deceptive participants using audio and video. Chittaranjan and Hung [4] use non-verbal audio cues such as: the length of speaking intervals, the number of speaking turns, interruptions and pitch. They achieve an F-Measure of 0.62 for deception detection. In a later work Raiman, Hung and Englebienne [13] combine these audio features with low quality visual data, in which facial expressions cannot be recognized, but gestures can. Combining the audio data with the video results in an improved detection rate with an F-measure of 0.76. Our experiment is run under the assumption that the conversation is performed over the web and visual and audio data are unavailable. Therefore our agent bases its deception detection component on chat text.

A similar study was performed by [7] who use digital audio tape to detect deceptive speech. They use both prosodic features (which do not use the actual words) such as pitch, loudness, duration, etc. and lexical features such as denials and flags for positive and negative emotion words (a total of 20 features). Each set was investigated separately as well as using a combined set of features. The combination provided the best results with an accuracy of 64.4%.

Another study on deceptive language has been performed by [10]. Data consists of paragraphs written by subjects who were asked to write their opinion on a topic and then write the opposite opinion. Naive Bayes and Support Vector Machines were used to classify words in this data set. Words that are connected to the self such as "I, friends, self" are often used in the honest text. In contrast, detached words such as "you, other, human" appear more often in the deceptive text. When topics were evaluated separately, an average detection rate of 70.8% was found. When testing the portability of the classifiers across topics, accuracy dropped down to 59.8% on average. A similar study analyzed text which is presented in on-line dating profiles [15]. This is a setting where the subjects have time to revise their descriptions, and make sure they sound honest. Yet again deceptive participants use less self references, and use more negations. 63% of the profiles were correctly classified using logistic regression.

Research in computer mediated environments [16] uses a text-based environment where deceptive senders had to persuade receivers to make decisions they knew to be incorrect. This experiment tested whether automatic extraction of linguistic cues using

NLP contributes to differentiation between deceptive and non-deceptive texts. In their work, though subjects were motivated to play the game, there was no special motivation for the deceptive players to be deceptive. The deceptive senders were found to be more expressive than honest senders. This is contrary to previous work that shows that liars tend to be less expressive. This result was explained by the fact that previous studies were conducted in an interview type scenario where a liar had to answer in real time without planning or editing, as opposed to this experiment where liars can edit and think about the message before they send it. This work was later extended [17] to a chat-based environment where the effects of the type of environment on deception detection by human subjects were studied. They found that humans found viewing the messages assisted detection more than viewing video of the actual typing of the liars. However the accuracy of the human deception detection in all settings was lower than 50%. Although the text-based domain we use is similar to the domains used by Zhou et al. our research differs substantially from these studies. We use machine learning to differentiate between deceptive and non-deceptive text as opposed to these studies that either collect statistics on the existence of automatically extracted phrases or use humans for detecting deception.

Mancilla-Caceres et al. [8] developed a game in order to identify children who act as bullies within a social network. The game involves a restricted set of resources and two tasks. One is collaborative and the other competitive. The communication between participants is through text messages. Mancilla et al. manually classify the messages exchanged and, using machine learning, are able to detect the bullies with an accuracy of slightly above 60%, however they state that the classifier finds a large number of false positives and plan to improve on this. This game is reminiscent of our game as it uses text messages and players are given a set of resources. However our game is aimed at assessing dishonest behavior and not aggressive behavior. To the best of our knowledge we are the first to deploy an autonomous agent in any such environment.

5 The DIG Agent

We present the Deception In Group detector and catcher Agent (DIG). DIG is composed of four components. The first is used to detect the deceiver, the second to avoid raising suspicion and to assure that others will find the DIG's words trustworthy. The third component directs the other credible players to DIG's suspect. The fourth component is in charge of answering simple questions.

The first two components require data and are built using machine learning techniques. Detecting the deceiver requires classified data which includes different users' texts and whether or not they played the role of a deceiver. In addition to the role played by each user, the second component requires an indicator as to whether this user raised suspicion (was voted as the pirate). DIG then uses sentences that reduce suspicion and avoids using sentences that raise suspicion. The third component is activated towards the end of the game when DIG states whom it thinks is deceiving with a certainty level (in the sentences) rising as the game comes to an end. For example, in the pirate game, towards the end of the game DIG says "I think that player A is the pirate" and later on it says "I insist that player A is the pirate". The fourth component, which answers simple questions, was manually programmed according to common questions and answers found in the data. A random delay was added to each sentence said by DIG.

6 Experiments

All of our experiments were performed using Amazon's Mechanical Turk service (AMT) [1][1]. Participation in all experiments consisted of a total of 320 subjects from the USA, of which 47.8% were females and 52.2% were males. The subjects' ages ranged from 18 to 67, with a mean of 32 and median of 30. All subjects had to pass a short quiz to assure that they understood the rules and had to practice the usage of the structured sentences before they could play. We ran experiments with the two versions of the game ("informer" and "basic"), each with two different setups. We ran the game with only human players and then with an agent playing the role of one of the credible villagers (the agent is never the pirate). The subjects weren't told about the automated agent and therefore assumed all players were humans.

Participants had to play 5 games. Each game was played with different participants, so no deductions based on participants' behavior can be made between games. The subjects were paid 62 cents for participating in the study. They gained 12 cents for every time they were a credible villager in a group that manged to catch the pirate and 36 cents if they were a pirate which manged to escape with the gold. A stake of 36 cents for a single game is relatively high in AMT (58% of total payoff for 5 games). This is in order to increase the players' incentive to play seriously, and is also influenced by Ekman's statement: [5] (pg. 59) "There is a simple rule: the greater the stakes, the more the detection apprehension".

The players enjoyed the game very much as they gave it on average a score of 4.01 on a scale of 1 to 5 scale, along with feedback such as: "It was the best survey or game I have done...", "This was really fun... Thanks for the good time!" and "That was so much fun! ... I could play it all day!". Interestingly, the subjects found the "informer version" of the game more enjoyable than the "basic version" - 4.22 vs. 3.79 ($p < 0.01$). This can be explained by the fact that in the "informer version" both the pirate and the credible villagers have a clearer task: the pirate needs to incriminate a certain player, and the credible villagers need to find who is trying to incriminate a different player.

6.1 DIG Composition

We used 41 features, some of which are mentioned in [16] and the rest were dedicated to our problem. Since we wanted our approach to be scalable and suitable also for a free chat environment, we used more general features such that other sentences (which do not appear in the structured sentences) may be mapped to these features as well. Some of the features also depend on the order of the sentences and not solely on whether a sentence was said (or how many times it was said). For the learning phase, each player is an instance which must be classified as a pirate or a credible villager. Due to lack of space we only present 10 selected features here.

[1] For a comparison between AMT and other recruitment methods see [12].

- **Fraction of talking:** the fraction of sentences said by the player out of the total number of sentences said (in current game).
- **Accusations:** the number of accusing sentences used by the player.
- **Consistency:** indicates the level of accusation consistency towards the other players. A player who always accused the same other player will have a high value, while a player who accused all three other players equally will have a low value.
- **Characteristics:** indicates whether a player referred to a different player's characteristics such as being too quiet, talkative accusatory etc.
- **First sentence:** indicates whether the player was first, second, third or fourth to say his first sentence.
- **First Accusation:** indicates whether the player was first, second, third or fourth to accuse a different player of being the pirate.
- **Self-justification:** indicates how many times the player stood up for himself.
- **Alter justification:** indicates how many times the player stood up for a different player.
- **Agreeing:** indicates whether a player agreed with other players (and how many times).
- **Agree to accusation:** indicates whether a player agreed with another player when that other player accused someone (and how many times).

Recall that two of DIG's components are based on machine learning. The first is building a model that identifies whether a player is a pirate or a credible villager. We used the results obtained from the all-human games (with no agent) as our learning data. We used an SVM classifier with a linear kernel and with the feature set mentioned above. For the "basic version" we used 204 examples and for the "informer version" we used 272 examples. We obtained an accuracy of 72.1% and an F-Measure of 0.69 for the "basic version" of the game, and similarly an accuracy of 72.8% and an F-Measure of 0.7 for the "informer version". This is better than the mafia game [4] when only audio data was used, and not as good as the mafia game [13] that combines audio and visual data. However, it is known that audio and visual data are strong deception predictors [5] (pg. 80) and therefore our results are very satisfying. During a game, DIG feeds all chat messages in the game into the SVM classifier and calculates its opponents' confidence value for being a pirate. DIG predicts that the player with the greatest confidence value is the pirate.

The second component requires building a model to determine which players appear suspicious to the other players, that is, which players received votes and which didn't. We use this component to supply DIG with information regarding which sentences to use and which to avoid. The model of this component was less accurate (using the same feature-set above), with only 56.4% accuracy and an F-Measure of 0.56 in the "basic version". This result implies that we aren't very successful at predicting whom other players suspect, possibly because the subjects themselves might have voted almost randomly. The accuracy in the "informer version" was slightly higher with 61.0% and an F-Measure of 0.61, implying that in the "informer version" we were more successful at identifying whom the other players suspect. Based on this model we instructed DIG to say "I am a credible villager" at the beginning of the game. DIG used "Alter justification" (stood up for other players) towards the least suspicious player. DIG started to accuse a player only towards the end of the game (which also results in a more accurate classification). We designed DIG to avoid asking other players whether they were pirates, as this was also shown to raise suspicion. DIG clearly said whom it would vote for “I will vote for...” rather than using the phrase “I believe that I will vote for...” as, once again, while the first is shown to lower the other players’ suspicion, the latter is shown to raise it.

6.2 Experimental Results

Figure 2 presents the success rate of the credible villagers at catching the pirate in both versions of the game, with and without the agent. As can be seen in the figure, in both versions of the game the groups including the DIG agent significantly outperform the groups that didn’t include the DIG agent (using chi square test, with $\alpha = 0.05$). Note that the performance of the human players without the agent are very close to the expected utility of random voting equilibrium mentioned in ’The Formal Model’ section, which is 0.26.

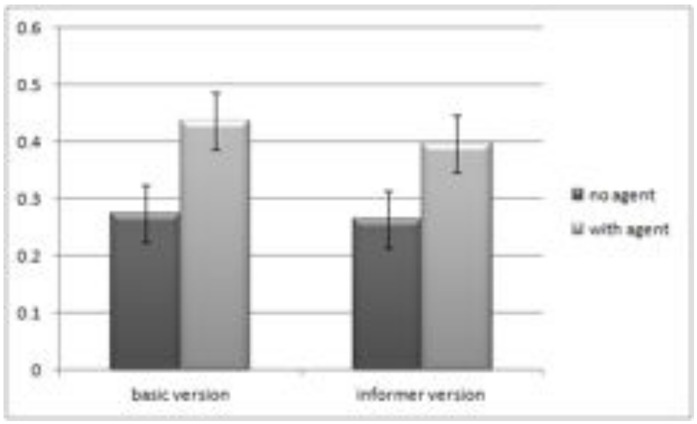


Fig. 2. Success rate in catching the pirate. Compares both versions of the game, with and without an agent.

Table 1 summarizes the voting results. As can be seen in the table not only did DIG help the group by casting more accurate votes, but DIG also seems to improve the number of correct votes cast by the humans in its group. We would also like to mention that very rarely did subjects cast an empty vote (only in 4% of the cases). This complies nicely with Proposition 1.

We end this section by testing the ability of DIG to avoid suspicion. Recall that one of DIG’s properties is to choose sentences that reduce suspicion. We measure suspicion by counting the number of votes cast by the other credible villagers on DIG. In the “basic version”, DIG received 32.2% of the votes which is still a little below average (which is 33.3%) and therefore may be reasonable. However, unfortunately, 37.6% of

game version	agent	correct votes	agent correct votes	human correct votes
basic	no agent	35.1%	-	35.1%
basic	with agent	41.9%	48.3%	38.6%
informer	no agent	33.8%	-	33.8%
informer	with agent	42.4%	46.6%	41.6%

Table 1. Pirate Game Voting Results (random vote = 33.3%)

the votes in the "informer version" were cast against DIG. We explain this by the fact that DIG tried to encourage the other humans to vote for the player which DIG detected as the pirate, as required by DIG's third component. This act in the "informer version" probably raised its suspicion (as players might have incorrectly assumed that DIG is the pirate and that it is trying to incriminate the informer). Another reason that could have caused suspicion in both versions is that people might have noticed that DIG doesn't play as a completely normal player, as DIG's fourth component was not the focus of this study. We believe that had we not selected DIG's sentences with care using the second component, the suspicion would have been even greater and plan to investigate this in future work.

7 Conclusions and Future Work

We have presented "the pirate game" as a platform that enables the study of deception in a group using a computerized text-based environment. We presented two versions of the game, the "basic version" where the deceiving participant could hide and an "informer version" where the deceptive player was encouraged to be active. In both versions we found that humans couldn't detect a deceptive player within the group, as their success rate was similar to the success rate achieved when all players cast a random vote.

We introduced DIG, an agent that uses machine learning to build a successful strategy for deception detection. DIG was successful at detecting the deceptive player in both versions of the game. DIG provided two contributions to the group of players. The first is that as we are looking at a group task, the ability of DIG to cast a correct vote increases the ability of the group. The second contribution of DIG is the indication that other players have higher detection rates when the agent is part of the game.

In future work we intend to pursue a method for the agent to select its sentences so that it increases its probability of detecting the pirate. The agent will need to find a question for each situation that when answered may either increase or decrease the probability that the agent's current suspect is the real pirate. This is a very challenging issue since each sentence may change the behavior of the rest of the game.

8 Acknowledgment

This work was supported in part by ERC grant $\#267523$, the Google Inter-university center for Electronic Markets and Auctions, MURI grant number $W911NF-08-1-0144$ and ARO grants $W911NF0910206$ and $W911NF1110344$.

References

1. Amazon. Mechanical Turk services. http://www.mturk.com/, 2012.
2. N. Bhaskaran, I. Nwogu, M.G. Frank, and V. Govindaraju. Lie to me: Deceit detection via online behavioral learning. In *Automatic Face Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 24 –29, march 2011.
3. Mark Braverman, Omid Etesami, and Elchanan Mossel. Mafia: A theoretical study of players and coalitions in a partial information environment. *Annals of Applied Probability*, 18(3):825–846, 2008.
4. Gokul Chittaranjan and Hayley Hung. Are you a werewolf? detecting deceptive roles and outcomes in a conversational role-playing game. In *ICASSP*, pages 5334–5337, 2010.
5. Paul Ekman. *Telling Lies: Clues to Deceit in the Marketplace, Politics, and Marriage*. W. W. Norton & Company, 1991.
6. J. Farrell and M. Rabin. Cheap talk. *The Journal of Economic Perspectives*, 10(3):103–118, 1996.
7. Martin Graciarena, Elizabeth Shriberg, Andreas Stolcke andFrank Enos, Julia Hirschberg, and Sachin Kajarekar. Combining prosodic, lexical and cepstral systems for deceptive speech detection. In *Proc. IEEE ICASSP*, 2006.
8. Juan Fernando Mancilla-Caceres, Wen Pu, Eyal Amir, and Dorothy Espelage. Identifying bullies with a computer game. In *AAAI*, 2012.
9. Piotr Migdal. A mathematical model of the mafia game. *CoRR*, abs/1009.1031, 2010.
10. Rada Mihalcea and Carlo Strapparava. The lie detector: Explorations in the automatic recognition of deceptive language. In *ACL/AFNLP (Short Papers)*, pages 309–312, 2009.
11. Matthew L Newman, James W Pennebaker, Diane S Berry, and Jane M Richards. Lying words: predicting deception from linguistic styles. *Pers Soc Psychol Bull*, 29(5):665–75, 2003.
12. G. Paolacci, J. Chandler, and P. G. Ipeirotis. Running experiments on Amazon Mechanical Turk. *Judgment and Decision Making*, 5(5), 2010.
13. Nimrod Raiman, HayleyHung, and Gwenn Englebienne. Move, and i will tell you who you are: detecting deceptive roles in low-quality data. In *Proceedings of the 13th international conference on multimodal interfaces*, ICMI '11, pages 201–204, New York, NY, USA, 2011. ACM.
14. Eugene Santos and Deqing Li. On deception detection in multiagent systems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 40(2):224 –235, march 2010.
15. Catalina L. Toma and Jeffrey T. Hancock. Reading between the lines: linguistic cues to deception in online dating profiles. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, CSCW '10, pages 5–8, New York, NY, USA, 2010. ACM.
16. Lina Zhou, Judee K. Burgoon, Jay F. Nunamaker, and Doug Twitchell. Automating linguistics-based cues for detecting deception in text-based asynchronous computer-mediated. *Group Decision and Negotiation*, 13:81–106, 2004.
17. Lina Zhou and Dongsong Zhang. Typing or messaging? modality effect on deception detection in computer-mediated communication. *Decision Support Systems*, 44(1):188 – 201, 2007.

Information Sharing for Care Coordination

Ofra Amir[1], Barbara J. Grosz[1], Roni Stern[1], and Lee M. Sanders[2]

[1] Harvard School of Engineering and Applied Sciences, Cambridge MA 02138, USA
oamir@seas.harvard.edu, grosz@eecs.harvard.edu, roni.stern@gmail.com
[2] Center for Policy, Outcomes and Prevention, Stanford University, Stanford, CA 94305, USA
leesanders@stanford.edu

Abstract. Teamwork and care coordination are of increasing importance to health care delivery and patient safety and health. This paper describes our initial work on developing agents that are able to make intelligent information sharing decisions to support a diverse, evolving team of care providers in constructing and maintaining a shared plan that operates in uncertain environments and over a long time horizon.

1 Introduction

The health care literature argues compellingly that teamwork is of increasing importance to health care delivery, and improved care coordination is essential to improving patient safety and health. The lack of effective mechanisms to support health care providers in coordinating care is a major deficiency of current health care systems [12]. This work is part of a broader project that aims to develop intelligent, autonomous multi-agent systems that work as a team supporting a diverse, evolving team of providers caring for children with complex conditions[3]. The agents will support providers in formulating a shared "care plan" that operates on multiple time scales and in uncertain environments, deploying that plan in their delivery of care, and monitoring and revising it as needed.

Figure 1 illustrates the complex environment in which agents supporting care for a child with a complex condition would operate. As can be seen in the figure, the care team is diverse and broad in scope, including not only physicians but also other types of care providers (e.g., therapists) and others who work with the child in various settings (e.g. teachers). The group of providers may change significantly over the years, whether as a result of personnel changes or because the child's condition or developmental stage raise different needs. Thus, caregivers' involvement with the child may be continuous or intermittent, long or short term, as represented by the horizontal lines in the figure showing time periods in which a caregiver is participating in the child's care. Providers differ in their expertise and address different aspects of a child's condition. Furthermore, the care team for these children seldom all come together in one place. The figure

[3] This project accords with the vision described in the paper "Collaborative Health Care Plan Support" in the challenges and visions track of the technical conference.

also highlights a distinguishing challenge of care for these children: their developmental stages (see horizontal center of figure) affect and may be affected by treatment, adding uncertainty to plan development and typically necessitating plan revision. While these factors make plan coordination and management very complex, it is crucial to the quality of care that this group acts as a team. To do so requires effective mechanisms for information sharing between team members. The focus of this thesis is developing supporting agents that are capable of making intelligent information sharing decisions.

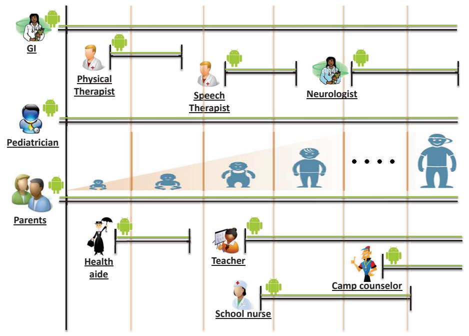


Fig. 1. Agents in the General Care Context

2 Challenges in Health Care Coordination

Our planned research on information sharing for care team coordination aims to develop agents capable of assisting care providers in ensuring that their individual treatments plans mesh and that those actions directed at short-term goals are compatible with longer-term goals. Insufficient communication among team members can lead to missing required treatment actions or conflicting actions. Agents could support caregivers with multiple responsibilities, and therefore limited time, by identifying from the large, heterogeneous body of information each has individually that portion most relevant to share, and determining those caregivers with whom it is important to share it. They would also track follow-up efforts to help ensure that treatment activities—including those for information-sharing—are carried out. The care for children with complex conditions has several distinguishing characteristics that make plan support challenging, including the following:

An evolving team: The various care providers differ in their expertise, knowledge about a child's condition, and concern with a child's longitudinal care plan. The team changes over time, as new providers may join, existing members may leave, and some may be active only intermittently. These characteristics differ radically from those of prior MAS work that has considered issues of forming teams and developing coordinated or collaborative plans.

Uncertain, evolving action sets: The commonly made "closed world" assumption, i.e., the set of actions and goals are constant over time, does not hold in long-term care planning, as the child changes developmentally over time, and new medical treatments may come into play. Planning needs to accommodate new actions (e.g., new treatments and therapies) and remove actions that are no longer relevant.

Conflicting goals and multiple time scales: As the care plan is executed, conflicting goals may arise, either from limited resources or from contradictory effects of actions at different time scales. For instance, an action might help achieve a short term goal but conflict with a long-term goal. Providers often fail to detect such conflicts until after their impact on a child occurs. Furthermore, in addition to achieving goals specified in the care plan, there are maintenance goals [5] that need to be considered.

The development of agents that are effective in such settings introduces several significant challenges. New **information exchange** capabilities are required for evolving groups to function as a team. Furthermore, as actions are added and removed or when conflicting goals are identified, agents will need to **alert the right set of providers at the appropriate time**. In addition, the changes in the team and action sets often require **re-planning that takes into account the long-term care plan**, which is characteristically different from re-planning for execution failures. Agents supporting such care teams will themselves operate in different contexts and have different beliefs about the state of the world and what each other and the people whom they support are doing. They will need strategies for resolving differences among contexts and for learning other agents' contexts, as well as for determining care providers' intended actions and beliefs.

3 Related Work

Prior multi-agent systems efforts e.g., Electric Elves [3], CALO [20], RADAR [7] have addressed the development of multi-agent and planning technologies for personal assistant agents that enable people to better accomplish their tasks in office environments and military settings. Since the primary goal of these projects was to develop a personal assistant agent, they focused largely on the support of a single individual. Some prior work, including Coordinators [19], has addressed collaboration among personal assistant agents, but there are key differences between these efforts and the CASPER goal. First, in prior work, the collaborating agents all operated within a single organization and shared a common vocabulary. The heterogeneity of agents in the health care domain is far greater, as they vary in levels of medical literacy and backgrounds. Furthermore, care givers typically work for multiple organizations. Second, prior work has considered task allocation and the generation of hierarchical plans for agents that, though distributed, are tightly coupled in their efforts [11]. In the health care domain, different providers operate semi-independently and have many competing tasks (e.g. caring for other patients). Their plans are only loosely coupled, but doing the right thing when they interact is essential to plan success. Furthermore, the underlying assumptions of most of these systems are unlikely to hold for CASPERs given the evolving, longitudinal nature of health care plans as new team members, treatments, actions and even goals are typically introduced to the system and there is the possibility for long-term and maintenance goals to interact complexly with short term goals.

Existing techniques for collaborative multi-agent planning are not fully able to handle essential characteristics of the care plan setting and often do not take into account communication mechanisms and costs. Classical planning and agent frameworks such as BDI agents [13] assume a closed world where the operators and goals are defined and fixed from the start. Dec-POMDP models [2] address uncertainties about action outcomes and about states, but are intractable for long horizon plans. Furthermore, they are not suited to incorporate new actions and agents as the care plan evolves, because they assume that a complete model of states and transitions is given in advance and known by all agents. Theories of teamwork and collaboration [8, 4, 17] support collaborative multi-agent planning, but assume a fixed action library and set of agents. A range of recent work on information exchange and communication algorithms for multi-agent settings has defined models for communication within teamwork [10, 15, 14, 6, 18], but these models have been implemented and evaluated only in environments much simpler than the care coordination domain. They also typically tightly limit communications options and make modeling assumptions, including constant team membership, which do not hold in this domain.

4 Approach

Our planned approach will build on the work described in Section 3, as well as that on modeling collaboration [8], helpful behavior [9] and interruption management [10, 16]. It will proceed in stages from simpler to more complex settings to address the information-sharing challenges of the care coordination domain. We are currently evaluating information sharing agents in a domain introduced by Roth et al. [15] to evaluate collaborative agents. In this game, the agents need to infer whether they are in Colorado or Wyoming based on terrain observations, and meet in an agreed location. Although this domain is much simpler than the care coordination setting, it shares some of its characteristics, including costly communication, agents who differ in their observations and uncertainty regarding the state they are in.

Recent work has used this domain to evaluate a computer agent interacting with a human teammate [6]. In their setting, the human and agent assume the same role, and are treated as equal collaborators. In contrast, in the healthcare domain, agents are unlikely to replace caregivers in making complex treatment decisions, and therefore we envision the agents as taking on a supporting role and assisting caregivers by reducing the burden of making information sharing decisions. Therefore, we will use a game setting in which human players make movement decisions and are assisted by agents which decide what observations to share with the other player.

In the second stage, we will evaluate agents in a more complex game setting which has more of the characteristics of the care coordination scenario. To this end we have designed a game in the ColoredTrails framework [1], inspired by discussions with a pediatrician who works with children with complex conditions. Figure 2 shows the game board we have designed. In this game, human players (denoted P1, P2, and P3 in the figure) are located in different positions on the board. There are several goals located on the board, denoted G1 to G4. The players need to agree on a set of shared non-conflicting goals to pursue (goals of the same shape in the figure). This scenario corresponds to caregivers choosing between alternative treatment options in the healthcare domain. There are several sources of uncertainty in the game. First, there is uncertainty about the utility of each goal. In each turn each of the players can move towards one of the goals. Players then obtain more accurate information about the utility of the goal they are approaching. This is analogous to physicians who have some estimate of how well a treatment would work, but can only assess whether it works for a particular patient as they start it. Second, players cannot observe the positions and actions of their collaborating player. This is similar to the healthcare setting in which care providers are typically unaware of all actions performed by others. Players can choose to communicate information to other; however, as in the Colorado/Wyoming game, in this game communicating information is associated with a cost, corresponding to the time required by physicians when they share information or are presented with new information. Finally, we plan to include "distracter" goals for each player. These goals contribute to each player's personal utility. They are included to make the scenario more realistic by simulating the situation in which each physician not only is concerned with caring for a particular child, but also has additional tasks relating to other patients. The game will be played by people, who would be responsible for choosing movement actions. These players would be supported by agents that make information sharing decisions by reasoning about other players' beliefs and the importance of sharing each observation. This abstract environment will enable us to test different representations and agent designs for supporting information sharing in various settings.

In both the Colorado/Wyoming domain and the more complex Care Coordination game we will perform extensive experiments to test the developed agents designs, algorithms and representations. These experiments will use different agent designs, as we plan to test several approaches including POMDP based decision making algorithms (e.g., POMDP, Dec-POMDP or NED-POMDP [10]), and representations for shared plans such as probabilistic recipe trees [9]. We will test different settings of support from the agents, including settings in which the agent makes information sharing decisions without consulting the person playing and setting in which the agent only recommends what information to share and the person can reject or accept that suggestion. We will also run experiments varying the number of players to increase complexity. Agents' performance will be measured based on the utility achieved by human players assisted by agents as compared to games in which people make both movement and information sharing decisions. In addition, we will measure the time savings for people that are a result of removing their responsibility to communicate with each other.

Finally, we will incorporate the decision-making mechanisms in agents designed to operate in the care coordination domain and test their abilities to support caregivers. We have been working with a pediatrician collaborator and his team to determine the needs of caregivers and learn what currently available systems do and do not provide. The need for systems to support care coordination was identified through interviews with parents and caregivers. We envision providing each caregiver, including the parents of the patients, an assistive agent. These agents be integrated with the existing electronic medical records systems, to gather the relevant information for the various caregivers and patients' families. In this real-world setting agents' performance will be evaluated according to several measures, such as user satisfaction, time savings and helpfulness of the information provided by the agents.

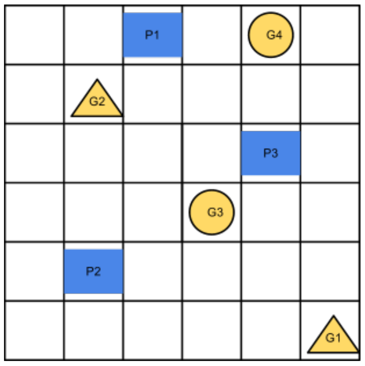


Fig. 2. A game design to test information sharing agents

5 Conclusion

Teamwork and coordination are crucial for providing high quality care, especially for children with complex conditions which have a large, diverse, team of providers responsible for their care. To work as a true team, caregivers need to be aware of relevant treatments carried by others and of information relevant to their care of the child. In this work we propose the development of agents that support such caregivers by making information sharing decisions, ensuring that caregivers have the information required to generate, monitor and revise a shared care plan. Beyond care for children with complex conditions, such agents have the potential to improve the delivery of health care for many patient populations, especially those in which multiple health issues interact (e.g., for cancer and for the elderly). In addition, similar issues arise for plan support in other long-term, complex team environments. For instance, relief plans following natural disasters (e.g., the earthquake in Haiti) involve diverse teams including local and international medical staff, social workers, educators and others. Some rescuers are involved for only a short time or intermittently. Furthermore, relief plans operate over multiple time scales (e.g., short term rescue endeavors, longer term re-establishment of educational systems).

Acknowledgments. The research reported in this paper was supported in part by grant IIS-0705406 from the U.S. National Science Foundation. We are grateful to Kobi Gal and Ece Kamar for many helpful discussions.

References

1. The colored trails project. `http://www.eecs.harvard.edu/ai/ct`.
2. D. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, pages 819–840, 2002.
3. H. Chalupsky, Y. Gil, C. Knoblock, K. Lerman, J. Oh, P. D.V., T. Russ, and M. Tambe. Electric elves: Agent technology for supporting human organizations. *AI Magazine*, 23(2):11–24, 2002.
4. P. Cohen and H. Levesque. Intention is choice with commitment. *Artificial intelligence*, 42(2):213–261, 1990.
5. S. Duff, J. Harland, and J. Thangarajah. On proactivity and maintenance goals. In *AAMAS*, pages 1033–1040, 2006.
6. A. Frieder, R. Lin, and S. Kraus. Agent-human coordination with communication costs under uncertainty. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pages 1281–1282, 2012.
7. D. Garlan and B. Schmerl. The RADAR architecture for personal cognitive assistance. *Intrnational Journal of Software Engineering and Knowledge Engineering*, 17(02):171–190, Apr. 2007.
8. B. Grosz and S. Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86(2):269–357, 1996.
9. E. Kamar, Y. Gal, and B. Grosz. Incorporating helpful behavior into collaborative planning. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 875–882, 2009.
10. E. Kamar, Y. Gal, and B. Grosz. Modeling information exchange opportunities for effective human-computer teamwork. *Artificial Intelligence*, 2012.
11. F. Oliehoek. Decentralized POMDPs. In *Reinforcement Learning: State of the Art, Adaptation, Learning, and Optimization*. Springer Berlin Heidelberg, 2012.
12. L. L. I. R. on Care Integration. *Order From Chaos: Accelerating Care Integration*. National Patient Safety Foundation, 2012.
13. A. Rao, M. Georgeff, et al. BDI agents: From theory to practice. In *The international conference on multi-agent systems (ICMAS)*, pages 312–319. San Francisco, 1995.
14. M. Roth, R. Simmons, and M. Veloso. Reasoning about joint beliefs for execution-time communication decisions. In *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 786–793, 2005.

15. M. Roth, R. Simmons, and M. Veloso. What to communicate? execution-time decision in multi-agent POMDPs. *Distributed Autonomous Robotic Systems 7*, pages 177–186, 2006.
16. D. Sarne and B. J. Grosz. Determining the Value of Information for Collaborative Multi-Agent Planning. *Journal of Autonomous Agents and Multi-Agent Systems*, To appear 2012-2013.
17. E. Sonenberg, G. Tidhar, E. Werner, D. Kinny, M. Ljungberg, and A. Rao. Planned team activity. *Artificial Social Systems*, 890, 1992.
18. P. Stone, G. Kaminka, S. Kraus, and J. Rosenschein. Ad hoc autonomous agent teams: collaboration without pre-coordination. In *In Proceedings of the 24th AAAI Conference on Artificial Intelligence*, 2010.
19. T. Wagner, J. Phelps, V. Guralnik, and R. VanRiper. An application view of coordinators: Coordination managers for first responders. In *AAAI*, pages 908–915, 2004.
20. N. Yorke-Smith, S. Saadati, K. Myers, and D. Morley. The design of a proactive personal agent for task management. *International Journal on Artificial Intelligence Tools*, 21(01), Feb. 2012.

SimplyPut: Leveraging a Mixed-Expertise Crowd to Improve Health Literacy

Edith Law[1], Barbara Grosz[1], Lee M. Sanders, M.D.[2], and Shira H. Fischer[3]

[1] Center for Research on Computation and Society
SEAS, Harvard University
elaw@seas.harvard.edu,grosz@eecs.harvard.edu
[2] Division of General Pediatrics
Stanford University
leesanders@stanford.edu
[3] Division of Clinical Informatics
Harvard Medical School
shira@post.harvard.edu

Abstract. A substantial amount of health information, online or in print, is much too technical for the general patient population to understand. Our project aims to create a system that combines intelligent agents with a mixed-expertise crowd to translate medical documents into text that can be digested and interpreted correctly by parents of chronically ill children.

Keywords: Interactive Human Computation, Health Literacy, Health Communication, Mixed-Expertise Crowd, Crowdware, Text Simplification and Summarization

1 Introduction

Parents are now able to draw on a wealth of information on the Web to help them manage the health of their children. For almost any disease, a variety of websites, wikis, and forums describe such key disease factors as causes, symptoms, prognosis, methods of treatment and prevention. This information is, however, not accessible to a large segment of the population: some have low literacy, others are unable to interpret correctly the results of studies, and yet others cannot determine the meaning of a general study for their individual child's care. A recent study [14] evaluating the health literacy of parents reveals that almost 30% of parents have below basic or basic level of health literacy. In general, about 20 percent of adult Americans have 5th grade or lower reading level [3], which is far below the level required to understand most print or online medical and healthcare literature. Low parental health-literacy levels are correlated with an inability to adhere to instructions and take appropriate actions, which ultimately negatively affects the health outcomes of their children. Similar challenges are faced by adult patients with lower reading level managing their own health.

To be useful, health information should be both *readable* and *comprehensible.* The readability of a piece of text refers to its difficulty, as measured by properties of the text itself, such as vocabulary level, sentence structure, use of technical jargon, and density of information [7]. In contrast, comprehension measures the extent to which the reader can accurately remember and interpret the information contained in the text. Research in natural-language processing has yielded a variety of methods for text summarization [10], simplification, sentence compression [11], and the prediction of reading levels, some of which have been focused on medical documents [4]. None of these methods have yet been used to transform medical documents at scale, and to-date the techniques have not been able to provide the accuracy and completeness needed for use by patients.

This paper describes research aimed at developing a collaborative multi-agent "human computation" system, called **SimplyPut**, which coordinates people with different levels of medical expertise, along with text-processing algorithms, to address the problem of producing healthcare information appropriate for readers at lower levels of health literacy. This effort builds on our prior work on "crowdware" [15]—applications designed to coordinate a crowd involved in a common task to achieve a goal—and on collaborative multi-agents systems. This system will coordinate a crowd of workers to perform the task of translating paragraphs from a medical journal article into simpler, more compact text that can be more easily understood by those with lower literacy levels. We treat this transformation as a kind of machine translation problem, in that it takes information written in technical medical language and translates it into simple, lay language that conveys the important points for patients (or their parents). This approach is similar to Soylent [2], which deploys crowdsourcing for editing and shortening text.

Ideally, one would be able to find workers who are *bilingual* in the sense that they know both the language of medicine and how to "speak plainly". There are insufficient people with such skills to address the problem at scale. Instead, we proposal a solution involving a *mixed-expertise crowd* comprising non-expert workers (e.g., Mechanical Turk workers, health forum members), semi-experts (e.g., medical students), and experts (e.g., doctors). This team works together, iteratively refining drafts to produce a final translation. Our goal is to develop a system that coordinates human agents from the mixed-expertise crowd and computer agents (e.g., NLP algorithms), intelligently dividing the labor between the two based on their estimated competence at various text transformation tasks. Different from Soylent, our system encourages the active involvement of the requester in guiding the workers and in shaping the final solution. Involving the requester (i.e., physician) is particularly important in the medical domain, where the translated material must not only be accurate and complete, but also serve the particular needs of the patients and their caretakers, making them aware not only of what the material says, but also of what it *means* for them.

This work is part of a broader project that aims to develop intelligent, autonomous multi-agent systems that work as a team supporting a diverse, evolving team of providers caring for children with complex conditions.[4] In this paper,

[4] This project accords with the vision described in the paper Collaborative Health Care Plan Support in the challenges and visions track of the technical conference.

we describe our vision for this research; specifically, we characterize the health literacy problem in greater detail, present observations from a pilot experiment, describe the initial system design, and discuss the anticipated challenges.

2 Characterizing the Problem

To understand the nature of the problem and the desired solution, we conducted a pilot experiment in which we examined differences between experts and non-experts in simplifying medical documents. The study used the following paragraph from a medical journal about the effects of oral corticosteroids on growth for kids with asthma.

> There is little doubt that oral corticosteroids such as prednisolone can have a detrimental effect on growth. Martin et al, in a prospective survey over 14 years, showed that children who had received oral steroids were significantly shorter than either asthmatic children who had not received steroids or non-asthmatic controls. However, this difference in height was only seen at age 14 years, and no difference was apparent by 21 years, indicating that the main effect of oral corticosteroids was to cause growth delay and affect the timing of puberty. The degree of growth retardation has been clearly linked to the frequency of oral corticosteroid use. However, there is also evidence that adult height can be permanently reduced in some children who have received long term oral corticosteroids for asthma.

Four medical doctors, with different types of expertise (neurology, pediatrics, general medicine) were asked to generate a 2-3 sentence summary that was appropriate for conveying to patients the information contained in the paragraph. They were asked also to articulate the process they used to generate the summary. The results are presented in Table 2.

Even in this small study a number of interesting factors emerged: different doctors assumed different specific goals. Some doctors focused on simple reporting of the facts: their process of translation involves simplification of the language, such as explicit steps to remove medical jargon, remove words with too many syllables, reduce the total number of words in the sentence and the density of the information in the text. Others added extra information, such as the definition of oral corticosteroids, to clarify the statements. One doctor went as far as constructing a “take home” message based on the paragraph, by developing two broad categories – “what does this say” and “what does this mean for my child” – and sorting the information in the paragraph into those categories.

Despite their differences, the doctors all clearly adopted a patient-centric approach to designing their summaries. They considered “what might confuse the patient,” “[what] words have different meanings in medicine than in normal parlance,” “what the goal was for the parents to know,” “translating subjects / controls into what this means for their kids,” or substituting in terms that are more medically relevant, such as “side effect” and “dosage.”

Translation	Process
A kind of medicine called "oral corticosteroids" that we use for asthma can negatively affect a child's growth. One study watching children's development over 14 years found that children on this medicine were shorter at age 14 than children who did not receive it. However, this was only at age 14: by the time these children were 21 years old, their height was normal, so the medicine may have just slowed their growth when they were younger and made puberty later. The more medicine the took, the stronger this effect was. However, there are other studies showing that perhaps the medicine may have permanent effects on height if they've taken the medicine for a long time.	I think there's some gestalt, like what about this might confuse the patient, versus the specifics. I am sensitive to words that have different meanings in medicine than in normal parlance, so however well you explain the study, I think it'll still be confusing if the patient doesn't know what steroids are, for example. I read the whole thing, thought about it, and then did the summary. I also thought about what the goal was for the parents to know - that a) there are side effects and b) the research is unclear about whether these are short-term or long-term. That's not very reassuring, but that seemed to be what we had, so that was all we could tell them, but I wanted at least that much to be clear.
Steroid pills, like the kind that children with bad asthma sometimes need to take, carry some bad side effects. One side effect is that kids are shorter, or may go through their growth spurt or puberty later. If only on steroids occasionally, the kids usually catch up, but if the steroid course is prolonged they may be shorter as adults.	Simplify language. Set the discussion up as a 'side effect' of treatment. Switch from population level discussion to one aimed more at a parent with a child, translating subjects/controls into what this means for their kid.
There is some evidence that children who take oral steroids (a class of anti-inflammatory medications) for asthma may experience some slowing of their growth. However, studies disagree on whether this effect is permanent, or whether these children eventually catch up. The frequency of use may be important.	I first read the entire paragraph, which contained sentences presenting several at times conflicting facts about steroids and growth, and identified the "lowest common denominator" with which all the sentences were in agreement, which is that steroids seem to have some effect on growth. I made this the topic of the first sentence in my summary and rewrote this to eliminate jargon. Next, I identified where the various sentences were in disagreement (which is regarding whether effects on growth were temporary or permanent) and made this the second sentence in my summary. Finally, I focused on the one sentence that may account for the discrepancy, which is the one concerning dose effects.
What does this say? If a child takes steroid pills or liquid by mouth, it may cause her to be shorter when she is an adult. This is mainly true for a child who takes steroid pills every day for one month or more. It is also more of a problem for children who take steroids before they are 14 years old. **What does this mean for my child?** If your child needs to take steroid pills for more than one month in a single year, ask her doctor if it may affect her growth. Your child may also need to use a steroid inhaler (puffer). A steroid inhaler is safe. It will not cause her to be shorter when she is an adult. If she does not take the steroid inhaler, she may get very sick.	I followed clear-health communication guidelines (Doak and Doak): - Answer the simplest and most actionable question(s). No more than 3 questions. - Choose no more than 3 messages per question. - Use the fewest possible sentences and the fewest possible words. - Use words with $<$ 3 syllables. Avoid any medical jargon. - Use sentences with $<$ 12 words. - Avoid phrases and complex sentences. - Avoid commas and parentheses. - Use large font size and bullets. - Maximize white space. I added my clinical experience.

Table 1. Doctors' Translations and Process

Translation	Process
A new survey showed that children who received oral steroids were shorter than other children. However, they caught up by age 21, showing that there was a delay in growth and not a true stunting. Long-term use may still stunt growth.	I tried to read a few sentences, then summarize as plainly as possible what the article was trying to say.
There is no proof that steroids administered orally has any detrimental effect on growth. It may stall growth at certain stages of life, thereby delaying puberty, but once the administration of steroid is stopped, growth would progress normally as before.	I had to avoid all possible medical terminology because the patient will not be understand it and so there is no point in elaborating it. Since there is no higher education, the explanation mustn't frighten the patient. It must instill faith in him. The patient must feel confident after the explanation.
Studies show that the use of oral steroid medicine, for treating asthma, can delay puberty in children. These children are shorter at or around 14 years old but eventually catch up by adult age. Although in some cases of prolonged use could cause permanent effects in height.	I just tried to make the paragraph be very simple and to the point. Avoiding difficult, medical jargon. I feel in such a situation the studies do not need to be intro as thoroughly. Just state the main problem, and then eliminate any worry by saying what is normal and can be done.
Studies show that children who use oral steroids were shorter than children who didn't at age 14, but were all the same by age 21. However, some children that used a lot of the steroids were permanently shorter than children who didn't use them as much.	I moved sentence by sentence and simplified terminology and combined sentences where possible.
Certain surveys do show a significant decrease in height in children who take oral steroids for asthma,but such difference in height was only seen at age 14 years, and no difference was apparent by 21 years. But there is no evidence yet to proves that adult height permanently reduces in children who take oral corticosteroids for asthma for a long time.	Went through the whole para first,understood in a normal manner and then typed the jist

Table 2. Turkers' Translations and Process

We ran an equivalent task on Mechanical Turk, in which we asked five workers with no medical expertise to summarize the paragraph in 2-3 sentences as if they were talking to a patient (who has low reading level) about it, and describe the process or steps they took to generate the summary. In general, Turkers seemed to have produced more straight-forward summaries that are sentence-by-sentence translations, involving less re-organizing of content and having fewer elaborations; very few translations address what the information *means* for the patient. Some translations are missing important facts; for example, the second translation failed to report the fact that prolonged usage of corticosteroids can have a permanent effect on height. None of the Turkers substituted in terms that are more familiar to patients, such as "dosage" and "side effect." Our vision is to create a system that enables the requester, e.g., a doctor, to iteratively point out such errors and suggest ways to rephrase the text to be more patient friendly.

3 SimplyPut: A Crowdware System for Translating Technical Documents

We present here our vision of a system called *SimplyPut*, a web application where experts and non-experts (e.g., Mechanical Turk workers) collaborate using a shared interface to translate technical documents for a target audience. In the medical domain, experts are medical professionals, the documents are medical information from journals or websites, and the audience are patients or parents of patients. The system will use a novel approach that leverages a mixed-expertise crowd and will support human-computer collaborations in this context.

To date, most human computation systems have been developed to address problems that allow for a task to be decomposed into operations that can be performed by individual workers, each working independently and having only a local view of the solution. A different approach is required if a task cannot be fully decomposed into individual tasks, i.e., if there are dependencies among the tasks undertaken by different workers. We have developed such an approach in prior work on a crowd-based collaborative interface called Mobi [15], in which workers asynchronously put together a travel itinerary. In Mobi, workers choose the type and amount of contributions they want to make, while the system exerts some control over the computational process by automatically tracking the violated constraints and alerting workers about things that need to be adjusted to meet these constraints. Mobi is essentially a groupware system for crowds – referred to as crowdware. It leverages communication, a shared interface, and mechanisms for keeping track of the goal and progression towards it, to help a group to accomplish a joint task. The underlying idea behind crowdware is an **interactive approach to human computation**: requesters and workers are given the ability to monitor the solution as it is being assembled and influence the computational outcome through intermittent communication with one another. The goal of *SimplyPut* is to allow the requester (e.g., a doctor) to interact with and guide workers with little or no medical expertise to produce a final document that satisfies his or her requirements.

Like Mobi, *SimplyPut* will incorporate a shared interface that different agents, whether human agents or computer agents, will use to collaborate and generate the solution together. By allowing human agents to critique other human agents and computer agents, the system can then adaptively decide on the division of labor [12, 1, 6] between human and computer, assigning to each party tasks at which it is most competent. Getting the division of labor right is a key requirement for collaborative interfaces [1]. Having expert and semi-expert workers amongst the crowd, the system can guarantee that the final document serves its true purpose, which in this case is to communicate to patients and their caretakers what they need to know, no more and no less. Finally, by keeping the requester in the loop, allowing them to monitor the state of the solution and influence the final outcome, we can ensure, to some extent, that the final translation is accurate and complete. We now describe the different aspects of the system and some key challenges.

3.1 Agents

In our system, there will be three different kinds of agents:

Requesters are human agents who are the *users* of the system needing a technical document to be translated.

Workers are human agents responsible for carrying out various text processing and verification tasks, including non-experts with no formal medical training, or semi-experts (e.g., medical students) and experts (e.g., doctors).

NLP Algorithms are computer agents responsible for carrying out various text processing and verification tasks.

We envision that these agents will interact in a variety of ways. The role of the requester is mainly to provide requirements and constraints to the worker agents, and to critique and provide feedback on their work. Human workers transform the text, by adding, removing, re-ordering content, and also providing feedback for other human or automated agents. Automated algorithms can perform text modification operations. Given evaluative feedback from the requester and other human worker agents, these algorithms can also update their behavior by learning from past mistakes or additional training examples. For example, the system can send the output of a sentence compression algorithm to human workers, asking them to identify and correct mistakes, e.g., if the algorithm has mistakenly deleted parts of the sentence that are important, or if it has failed to identify superfluous parts of the sentence. This evaluative feedback from the human worker agents can then serve as new training example for the sentence compression algorithm.

A key challenge in designing this system will be to get the division of labor right, assigning to the computer agents only tasks they can do well, and likewise for workers at lower levels of the expertise hierarchy. We will follow an iterative, learning approach to this problem. Initially we will assign to both computer agents and people the same tasks—e.g., take a sentence and remove unnecessary elements from it—and then have human agents vote on the best results. If a sentence compression algorithm is comparable to human performance, then the system should leverage it over human workers; otherwise, the system will delegate the tasks to human workers.

3.2 Tasks

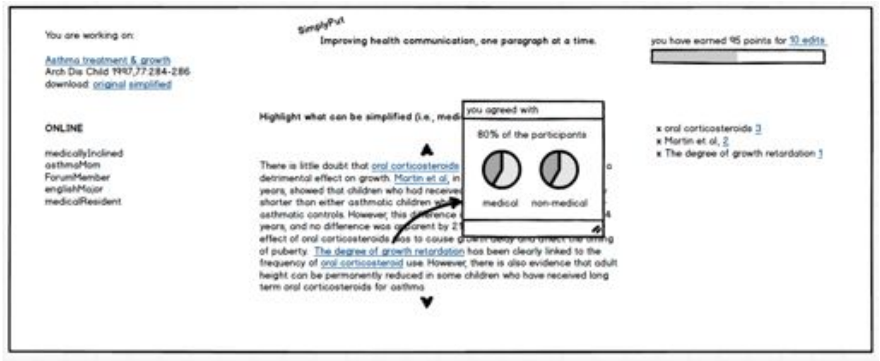
There are a variety of text transformation tasks that human workers and automated algorithms can perform. Some of these tasks are on the sentence level, while others may be on the paragraph level. They will include

Highlight Identify words or phrases that can be simplified (e.g., jargons).
Substitute Replace a word or phrase with easier to understand terms.
Chop Compress text to remove unimportant words or phrases.
Re-Order Reorder parts of the text (e.g., changing passive to active voice).
Elaborate Add definitions or explanations to the text.
Condense Combine multiple paragraphs by retaining only the important facts.
Verify Judge the work of other human or computer worker agents.

These tasks can be combined into a workflow, and assigned to workers by the system or by the requester. In particular, the requester can monitor the current solution, identify problematic areas, and assign new text transformation tasks to fix the problems. For example, the requestor may realize that he wants the sentences to be shorter, in which case, he may assign a *chop* task with the requirement that each sentence be limited to at most n words.

3.3 Interface

Figure 1 illustrates our initial design of SimplyPut. In the middle panel, the system presents to the worker a paragraph and a specific text transformation task. Users can choose to work on the current paragraph, or skip to some other paragraphs (and tasks) by scrolling up or down.


SimplyPut
Improving health communication, one paragraph at a time.
You are working on:
Asthma treatment & growth
Arch Dis Child 1997,77:284-286
download: original simplified
ONLINE
medicallyInclined
asthmaMom
ForumMember
englishMajor
medicalResident
you have earned 95 points for 10 edits.
Highlight what can be simplified (i.e., medi
you agreed with
80% of the participants
medical
non-medical
x oral corticosteroids 3
x Martin et al, 2
x The degree of growth retardation 1

There is little doubt that oral corticosteroids
detrimental effect on growth. Martin et al, in
years, showed that children who had received
shorter than either asthmatic children wh
asthmatic controls. However, this difference
years, and no difference was apparent by 21
effect of oral corticosteroids was to cause
of puberty. The degree of growth retardation has been clearly linked to the
frequency of oral corticosteroid use. However, there is also evidence that adult
height can be permanently reduced in some children who have received long
term oral corticosteroids for asthma

Fig. 1. SimplyPut: An Envisioned Design

The example shown here is a *highlighting* task, which asks the worker to select words or phrases in the text that can be simplified. With each selection, the system provides immediate feedback to the worker, indicating whether his or her selection agrees with what other participants have previously chosen. The worker can download the original article as well as the (partially or completely) simplified version. Finally, the interface displays information about the progress of the translation and the amount of contribution the worker has made.

3.4 Challenges

Evaluation Translations produced by our system should be correct, readable and comprehensible to the intended audience. We can evaluate each translation on its perceived and actual readability [8] using both automated metrics (e.g., Flesch [5] and SMOG [9]) and human judgments. To ensure that our tool is useful in practice, we will also evaluate the comprehensibility of the translations in a variety of ways. A common method for evaluating comprehension is the Cloze procedure [13], which asks evaluators to fill in the blanks of a written passage which has words deleted at a randomly chosen set interval. Alternatively, one can evaluate comprehension using explicit questions. We can, for example, ask the requesting doctor to generate a list of take-home facts that a patient should remember about the text, as well as a list of questions that will test the comprehension of these take-home facts. Each translation will be first reviewed to see how many and which of the take-home facts they retained (which measures its precision and recall), then evaluated on its comprehensibility by asking human evaluators to answer questions (generated by the requesting doctor) about those take-home facts.

Incentives Our system is aimed at bringing together a variety of people—doctors, medical and health communication students, as well as interested citizens (e.g., members of various health forums)—to collaboratively simplify medical documents. While doctors may be interested in disseminating scientific research findings to their patients in an understandable way, other participants may be interested in reading about certain health conditions and diseases, meeting other people with similar interests, educating themselves on how to write in a patient-friendly language, or contributing to the greater good of improving health literacy and communication. A major challenge, therefore, is in designing our system to simultaneously address a variety of intrinsic motivations.

Division of Labor Non-experts may produce lower quality work, but have substantially more time to devote to the endeavour than semi-experts or experts. Natural language processing algorithms may be less competent, but significantly cheaper to deploy than paying humans to perform the same tasks. To be cost-effective, our system needs to optimally divide the work amongst non-experts, experts and automated algorithms to produce translations, while taking into account monetary and time constraints.

4 Conclusion

In this paper, we present the health literacy problem and propose to address it though development of a collaborative crowdware system called SimplyPut. The human and computer agents in this system will collaborate to translate medical documents, making them more accessible and easy to understand for the general patient population. Our system is intended to be as much of a communication vehicle as it is a translation tool. Through this large-scale, collaborative effort to simplify medical information, we hope that people from otherwise disjoint communities—i.e., doctors, students, patients—will begin to understand how to better communicate with one another.

Acknowledgments. We thank Stuart Shieber for numerous discussions about the direction of this project, and for his valuable suggestions and feedback. The research reported in this paper was supported in part by grant IIS-0705406 from the U.S. National Science Foundation.

References

1. T. Babaian, B. Grosz, and S. Shieber. A writer's collaborative aid. In *UIST*, pages 7–14, 2002.
2. M. S. Bernstein, G. Little, R. Miller, B. Hartmann, M. Ackerman, D. Karger, D. Crowell, and K. Panovich. Soylent: a word processor with a crowd inside. In *UIST*, pages 313–322, 2010.
3. C. Doak, L. Doak, and J. Root. *Teaching Patients with Low Literacy Skills.* J.B. Lippincott Company, 1996.
4. N. Elhadad. *User-Sensitive Text Summarization: Application to the Medical Domain.* PhD thesis, Columbia University, January 2006.
5. R. Flesch. A new readability yardstick. *Journal of Applied Psychology*, 32:221–233, 1948.
6. B. Grosz and L. Hunsberger. The dynamics of intentions in collaborative intentionality. *Cognitive Systems Research: Special Issue on Cognition, Joint Action and Collective Intentionality*, 7:2–3, 2005.
7. G. Leroy, S. Helmreich, and J. Cowie. The influence of text characteristics on perceived and actual difficulty of health information. *International Journal of Medical Informatics*, 79:438–449, 2010.
8. P. Ley and T. Florio. The use of readability formulas in health care. *Psychology, Health and Medicine*, 1(1):7–28, 1996.
9. G. McLaughlin. Smog grading – a new readability formula. *Journal of Reading*, 12(8):639–646, 1969.
10. A. Nenkova and K. McKeown. Automatic summarization. *Foundations and Trends in Information Retrieval*, 5(2-3):103–233, 2011.
11. E. Pitler. Methods for sentence compression. Technical report, University of Pennsylvania, 2010.
12. S. M. Shieber. A call for collaborative interfaces. *ACM Computing Surveys*, 28(4es), 1996.
13. W.L. Taylor. Cloze procedure: A new tool for measuring readability. *Journalism Quarterly*, 30:415–433, 1953.
14. H.S. Yin, M. Johnson, A.L. Mendelsohn, M.A. Abraham, L.M. Sanders, and B. P. Dreyer. The health literacy of parents in the united states: a nationally representative study. *Pediatrics*, 124(3):289–298, 2009.
15. H. Zhang, E. Law, K. Gajos, E. Horvitz, R. C. Miller, and D. Parkes. Human computation tasks with global constraints: A case study. In *CHI*, 2012.

AgentSwitch: Towards Smart Energy Tariff Selection

Sarvapali D. Ramchurn,[1] Michael A. Osborne,[2] Oliver Parson,[1] Talal Rahwan,[1]
Sasan Maleki,[1] Steve Reece,[2] Trung D. Huynh,[1] Mudasser Alam,[1] Joel E. Fischer,[3]
Tom Rodden,[3] Luc Moreau,[1] Stephen Roberts[2]

[1] Electronics and Computer Science, University of Southampton, SO17 1BJ, UK
{sdr,op106,tr,sm1g09,tdh,ma606,lavm}@ecs.soton.ac.uk
[2] Pattern Recognition Lab, University of Oxford, Oxford, UK
{mosb,sjrob}@oxford.ac.uk
[3] Mixed Reality Lab, University of Nottingham, Nottingham, UK
{Joel.Fischer,Tom.Rodden}@nottingham.ac.uk

Abstract. In this paper, we present AgentSwitch, a prototype agent-based platform to solve the electricity tariff selection problem. AgentSwitch incorporates novel algorithms to make predictions of hourly energy usage as well as detect (and suggest to the user) deferrable loads that could be shifted to off-peak times to maximise savings. To take advantage of group discounts from energy retailers, we develop a new scalable collective energy purchasing mechanism, based on the Shapley value, that ensures individual members of a collective (interacting through AgentSwitch) fairly share the discounts. To demonstrate the effectiveness of our algorithms we empirically evaluate them individually on real-world data (with up to 3000 homes in the UK) and show that they outperform the state of the art in their domains. Finally, to ensure individual components are accountable in providing recommendations, we provide a novel provenance-tracking service to record of the flow of data in the system, and therefore provide users with a means of checking the provenance of suggestions from AgentSwitch and assess their reliability.

1 Introduction

Energy poverty is a rapidly growing issue across the world due to the significant rise in energy costs over the last few years.[4] Such increments are due to the unprecedented growth in energy demand (e.g., world energy is set to grow by more than 50% by 2030) coupled with dwindling fossil fuels and the high costs of (and resentment against) constructing large renewable energy generation facilities. This is set against a background where ageing nuclear and coal power stations are gradually being decommissioned (and as a result, a recent report suggested widespread blackouts are to be expected in the United Kingdom by 2015 [11]). In the UK energy market, we note that the loss of generation capacity is exacerbated by a misalignment of incentives whereby energy companies have no motivation to innovate nor to adopt cleaner sources of energy, given that they can easily pass on any rising costs they face directly to their customers. Thus, unsurprisingly, retailers have made significant profits even during periods of serious economic downturn[5]. In addition, most consumers (40%-60% in the UK) tend to 'stick'

[4] According to OFGEM UK, half a million households have been put into fuel poverty due to price increases in 2008 alone, while domestic energy bills have nearly doubled since 2003.
[5] One of the largest energy suppliers in the UK noted a rise in profits of more than 20% in the first half of 2012 despite the UK economy being in recession.

to the same energy supplier year in year out and do not spend much time looking for a cheaper deal. This reduces competition in the energy retail market and does not help drive down prices [12,20].[6]

Now, consumers cannot be completely blamed for not finding the best deal, given that energy tariffs are often made explicitly complex and, at times, confusing. For example, tariffs may have multiple tiers (e.g., the first tier may be priced at 20p per kWh, and beyond this the cost drops to 5p/kWh), implement time-of-use pricing (e.g., 5p/kWh between 11pm and 7am), and may include additional one-off discounts (often only for a limited period). To help consumers combat this complexity, a number of third-party online services exist to help consumers submit simple estimates of their yearly consumption and obtain the cheapest tariff (e.g., uswitch.com and moneysupermarket.com). Some other services also claim to help consumers come together as a collective in order to access group discounts from retailers (e.g., which.com, incahoot.com). Crucially, however, these services rely on consumers being able to make a reasonable estimate of their yearly consumption (taking into account varying usage over different seasons and usage at on- and off-peak times) and being able to understand how to take advantage of the various tiers or time-of-use tariffs they offer (e.g., by shifting appliance usage to off-peak times). Moreover, in existing collective purchasing systems, all members of the collective tend to obtain the same contract without considering whether the discounts are fairly distributed across all the members of the collective (e.g., those with unpredictably peaky consumption profiles should be charged more than those with predictably flat profiles, as they tend to cause higher penalties in the balancing market).

Against this background, here we report on the development of a prototype agent-based platform, called AgentSwitch, that integrates state of the art techniques and mechanisms to address the challenging issue of energy tariff selection. AgentSwitch builds upon the data provided by off-the-shelf energy monitoring devices and applies a number of machine learning, optimisation, and coalition formation algorithms in order to solve the energy tariff selection problem. In more detail, this work advances the state of the art in the following ways. First, we develop novel extensions to Bayesian Quadrature (a machine learning technique), in order to generate predictions of yearly consumption at hourly level. These estimates can then be directly used to select the best tariff traditionally available from energy retailers. Second, using the predictions of yearly consumption, we develop a novel mechanism for collective energy purchasing. The mechanism relies on a novel scalable algorithm to compute the Shapley value for coalitional games involving thousands of agents (homes). Third, we present a novel non-intrusive appliance load monitoring (NIALM) algorithm that works on coarse energy data (at five-minute level rather than second-level as is traditionally the case in this field) in order to detect deferrable loads that might benefit from being shifted to off-peak times. This algorithm is shown to outperform the state of the art on standard datasets. Fourth, we implemented a novel provenance service that allows the tracking of data throughout the system in order to provide accountability for its recommendations. As such, AgentSwitch instantiates the foundational tools, substantiated by benchmarks against the state of the art, in order to address a real world challenge for real users.

The rest of this paper is structured as follows. Section 2 details the system architecture underlying AgentSwitch. Section 3 elaborates on our Bayesian Quadrature model to predict yearly energy consumption from limited data. Section 4 then details our group

[6] Indeed, research by the U.S. Dept. of Energy found that most people are likely to spend *no more than two hours a year* setting their preferences for comfort, tariffs, and environmental impact [20]

buying mechanism that includes techniques to form coalitions via clustering of homes with similar attributes, along with a novel scalable Shapley value computation algorithm that allows for approximately fair distribution of energy costs across consumers. Section 5 then presents our load disaggregation algorithm that helps provide better suggestions to users. Section 6 presents our provenance tracking mechanism. Finally, Section 7 concludes the paper.

2 AgentSwitch Architecture

AgentSwitch is implemented as a prototype web application that incorporates a number of key modules (see Figure 1) as follows: (i) an annual load prediction module that uses Bayesian Quadrature (BQ) to predict annual electricity consumption, (ii) a group buying module that identifies and forms coalitions of consumers to take advantage of group discounts from retailers, (iii) a load disaggregation module that uses NIALM techniques to analyse home-level electricity readings to identify opportunities for savings by shifting appliances to off-peak times, (iv) a provenance tracking service that tracks the flow of data in the system, and finally (v) user interfaces to input data and visualise recommendations in such a way that the complexity of the processes underlying AgentSwitch are hidden from view.

Now, in order to provide tariff recommendations to users who sign up to use the service (i.e., allow AgentSwitch to analyse their data to provide recommendations and formulate group buying strategies), AgentSwitch needs to access at least[7] two key sources, namely consumers' electricity consumption readings (to be kept in a database, termed Energy DB, in AgentSwitch for algorithms to analyse) and live electricity tariff specifications from all retailers (for AgentSwitch to match consumption predictions or group consumption against the best tariffs).[8] While the former can be obtained from users' off-the-shelf energy monitors or smart meters that provide average power readings at different levels of granularity, the latter can be obtained from online third-party providers and suppliers (in our case, we used live tariff data from `uswitch.com` with their permission).

In what follows, we individually detail the core computational components of AgentSwitch as presented above. While we omit detailed descriptions of the architectural elements enabling the efficient storage and analysis of data in the Energy DB as well as user interface elements, they constitute important features of AgentSwitch that render it fit for a real-world deployment. Thus, we choose to focus on the core Artificial Intelligence (AI) elements of AgentSwitch in what follows.[9]

[7] Other sources of data such as weather predictions and historical data, users' schedule and travel plans for the year ahead, the users' home types and contents, would all be useful to improve predictions and load disaggregation and could be integrated into our approach. Thus, despite its complexity, AgentSwitch in its current shape represents only the first step towards building fully integrated tariff selection and home energy management systems.

[8] As a first step, we only consider electricity due to the wide availability of such electricity monitors, but this approach could be easily extended to gas or water if the appropriate sensors become available and affordable.

[9] As this is more relevant to the Agents community.

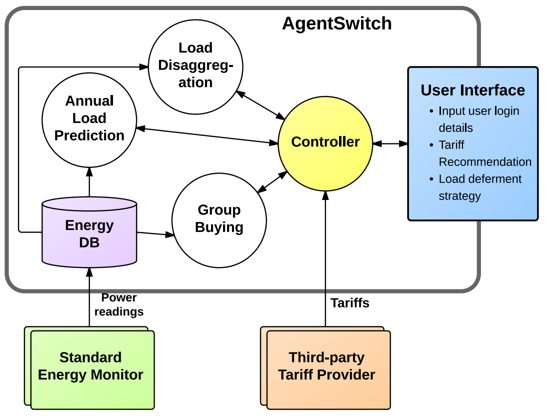


Fig. 1. The architecture of AgentSwitch showing the data flows (arrows) between different modules. The circles represent algorithms, rectangles represent data providers or sinks. All the data flows in the system are tracked by our provenance service.

3 Annual Load Prediction

The choice of an electricity tariff is based on a consumer's annual energy consumption and, to date, most online services would normally use a consumer's monthly usage and correlate this with the national average (for a given type of consumer — large family in a large house, single living in a flat) in order to arrive at an estimated annual consumption (possibly ignoring the division between peak and off-peak consumption). In contrast, here, we present a principled approach to computing such estimates in order to select the best tariff for a given customer. To this end, we employ a Gaussian process (GP) to model power consumption as a function of time. In particular, we apply this Gaussian process model to estimate the integral of power consumption over a year (the total annual energy consumption) so that estimates of time-of-use costs are correctly computed. This technique is known as Bayesian Quadrature (BQ) [3,13,5,16], a model-based means of numerical integration. To date, only very simple GP covariance functions (typically Gaussian) have been used to perform BQ. In this work, we extend Bayesian Quadrature techniques to use a more sophisticated GP covariance, suitable for quasi-periodic signals. Such signals emerge from the weekly cycles of energy consumption that typical domestic consumers exhibit.

3.1 Bayesian Quadrature

Bayesian quadrature [13,16] is a means of performing Bayes-ian inference about the value of a potentially nonanalytic integral,

$$\langle f \rangle := \int f(x)p(x)\mathrm{d}x. \tag{1}$$

For clarity, we henceforth assume the domain of integration $\mathcal{X} = \mathbb{R}$, although all results generalise to $\mathbb{R}^n$. Previous work on BQ assumes a Gaussian density $p(t) := \mathcal{N}(t; \nu_t, \lambda_t)$, although other convenient forms, or, if necessary, the use of an importance re-weighting trick ($q(x) = {}^{q(x)}/_{p(x)}p(x)$ for any $q(x)$), allow any other integral to be approximated.

Quadrature involves evaluating $f(t)$ at a vector of sample points $\mathbf{t}_s$, giving $\boldsymbol{f}_s := f(\mathbf{t}_s)$. Often this evaluation is computationally expensive; the consequent sparsity of samples introduces uncertainty about the function f between them, and hence uncertainty about the integral $\langle f \rangle$.

Previous work on BQ chooses a Gaussian process (GP) [17] prior for f, with mean μ_f and squared exponential (or un-normalised Gaussian) covariance function, suitable for modelling very smooth functions,

$$K_{SE}(t_1, t_2 | h, w) := h^2 \exp -\frac{1}{2}\left(\frac{t_1 - t_2}{w}\right)^2 \tag{2}$$

Here hyperparameter h specifies the output scale over f, while hyperparameter w defines an input scale over t. This covariance can be readily modified to give a periodic covariance, suitable for modelling periodic functions,

$$\begin{aligned} &K_{PSE}(t_1, t_2 | h, w, P) \\ &:= h^2 \exp -\frac{1}{2}\left(\frac{1}{w}\sin\left(\pi\frac{t_1 - t_2}{P}\right)\right)^2 \end{aligned} \tag{3}$$

where h and w are hyperparameters that have interpretations as for the squared exponential, and P is the hyperparameter representing the period.

We use the following dense notation for the standard GP expressions for the posterior mean m and covariance Σ, respectively: $m_{f|s}(t_\star) := m(f_\star | \boldsymbol{f}_s)$ and $\Sigma_{f|s}(t_\star, t'_\star) := \Sigma(f_\star, f'_\star | \boldsymbol{f}_s)$.

Variables possessing a multivariate Gaussian distribution are jointly Gaussian distributed with any affine transformations of those variables. Because integration is affine, we can hence use computed samples $\boldsymbol{f}_s$ to perform analytic Gaussian process inference about the value of integrals over $f(t)$, such as $\langle f \rangle$. The mean estimate for $\langle f \rangle$ given $\boldsymbol{f}_s$ is

$$\begin{aligned} m(\langle f \rangle | \boldsymbol{f}_s) &= \iint \langle f \rangle \, p(\langle f \rangle | f) \, p(f | \boldsymbol{f}_s) \, \mathrm{d}\langle f \rangle \, \mathrm{d}f \\ &= \iint \langle f \rangle \, \delta\Big(\langle f \rangle - \int f(t) \, p(t) \, \mathrm{d}t\Big) \, \mathcal{N}\big(f; m_{f|s}, \Sigma_{f|s}\big) \, \mathrm{d}\langle f \rangle \, \mathrm{d}f \\ &= \int m_{f|s}(t) \, p(t) \, \mathrm{d}t \,, \end{aligned} \tag{4}$$

which, for Gaussian input density and squared exponential covariance, is expressible in closed-form due to standard Gaussian identities [16].

3.2 Applying BQ to Household Energy Consumption Prediction

We applied our model to predict the yearly energy consumption of a test set of UK homes. This application motivated the development of a bespoke GP mean and covariance to permit long-range forecasting. The prior mean function was taken as:

$$\mu(t|a) := c\,\alpha(t), \tag{5}$$

where $\alpha(t)$ is the UK national average energy consumption for time-of-use consumers drawn from data provided by Elexon Ltd. Figure 2 illustrates this national average consumption over the year. c is a hyperparameter acting as a scaling constant. Our motivation for this prior mean was to permit the model to appropriately extrapolate from the patterns of consumption in winter to the remaining seasons in 2012. Then, the covariance chosen was built from terms of the form (2) and (3),

$$\begin{aligned} &K(t_1, t_2|h_a, w_a, w_p, P, h_b, w_b) \\ &:= K_{SE}(t_1, t_2|h_a, w_a)K_{PSE}(t_1, t_2|h = 1, w_p, P) \\ &\qquad + K_{SE}(t_1, t_2|h_b, w_b), \end{aligned} \tag{6}$$

modelling the quasi-periodic behaviour of the signal (e.g. lower energy use at night). Note that this covariance introduces additional hyperparameters. The period P is set *a-priori* to one day; the daily period was by far the most significant repeated pattern in the data. Other hyperparameters (including an additional observation noise variance hyperparameter) are fitted using maximum likelihood for each available data-set.[10] Figure 3 displays an example of GP regression using the model above.

We ultimately wish to compute an integral of power over time in order to estimate energy computation. That is, we need to solve (1) for an input density $p(t)$ that is constant for the desired time period and zero elsewhere. Unfortunately, the complex form of (6) rules out the analytic computation of (4) required for traditional BQ. In order to effect BQ for our problem, we discretise by assuming power is constant for each observed minute, and aggregate to provide observations of total energy consumption in each observed hour. We can then use these within the BQ framework as noisy integral observations of the true power signal. These observations can then be used to directly infer the desired integral over the entire year. The use of the BQ formalism allows for both a mean and variance to be provided for such integrals. Thus, the flexibility of our model allows for inference of the integral of power over any desired time period. Use of the time-of-use tariffs implies different electricity prices in nominal night and day periods, in order to encourage electricity demand at night. As such, in order to provide recommendations to users of the benefits of switching to or from the Economy 7 tariff, we must estimate both annual day-time and night-time energy consumption. This is trivially achieved by our approach.

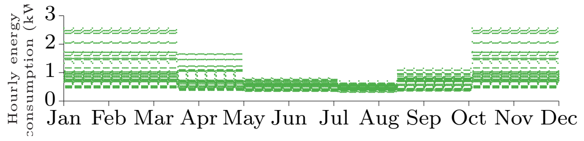


Fig. 2. UK national average hourly energy consumption throughout the year.

3.3 Evaluating BQ

We evaluated the efficacy of our prediction module on half-hourly consumption data collected for a set of 18 UK homes over winter 2011-2012 (our longest metered set of

[10] Note that we do not yet consider correlations between houses. Future work might investigate the emergence of such correlations from observable demographic information, such as household income and the number of household residents.

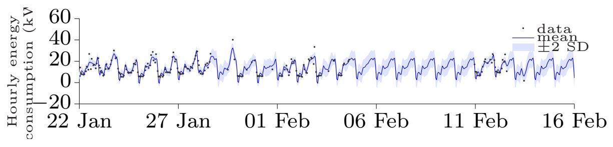


Fig. 3. Example of GP regression for energy consumption given sparse observations.

homes out of a pool of 3000) and divide the data into a set of training data (one third of the total data) and test data (the remaining two-thirds); the goal was to accurately predict the total energy usage across the test period given only training data. To provide comparisons for the BQ method, we implemented two alternative methods. The first predicted that energy usage for the household would be identical to the national average, ignoring the data. The second assumed that energy usage for the test data would continue at exactly the mean usage rate across the training data; this method ignores prior knowledge about the national pattern of energy usage. The root mean square errors (RMSEs) across all households are tabulated in Table 1. These experiments reveal that in combining both prior knowledge and the information contained in data, BQ returns slightly better predictions than either alternative. While the improvement offered by BQ for this limited segment of data are modest, we are hopeful that datasets spanning entire annual cycles (or taking into account localised weather patterns) will permit the full modelling capacity of BQ to achieve more significant gains. We expect that explicitly encoding a non-negativity constraint [14] for energy usage may also improve the BQ model. Note that in addition to providing mean estimates of energy consumed, the probabilistic BQ method also provides *an indication of the uncertainty in such quantities* in terms of the variance in yearly consumption. The provision of such uncertainty estimates is crucial to helping consumers understand the degree of confidence that can be attributed to the savings that a given tariff could provide. Thus, if a consumer has a large consumption variance, she may prefer to choose a tariff which is less risky (e.g., fixed average price rather than one that charges extra for going above a given tier).

Table 1. RMSE (MWh) for predictions of total energy usage.

Nat. Ave.	Mean	BQ
4.57	0.28	**0.27**

By predicting consumption accurately using the techniques above, we are able to choose the best tariff for any given home (resulting in savings of hundreds of pounds depending on the previous tariff). In the next section, we extend the use of such predictions to help create efficient energy purchasing collectives in order to take advantage of group discounts.

4 Collective Energy Purchasing

The creation of consumer collectives is a challenging problem given the number of potential consumers that can be grouped, along with the vast amounts of historical energy consumption data they come with. To date, apart from generic signup mechanisms, there is no principled way to group energy consumers and take advantage of group discounts (e.g., forming groups with flat energy profiles that enable the purchase of energy contracts in the forward market or green groups that prefer power from renewable supplies). A key challenge is the problem of sharing the group discounts "fairly" among group members. In this section, we describe a principled approach to do so based on the concept of the *Shapley value* from cooperative game theory [19], and address the computational issues that arise by presenting a novel, scalable algorithm that is integrated within AgentSwitch, and is shown to significantly outperform the state of the art.

In what follows, we first describe how we model the collective energy purchasing problem as a cooperative game involving coalitions (i.e., non-overlapping groups) of consumers, and discuss the challenges involved in computing the Shapley value. Second, we describe our algorithm for computing the Shapley value. Finally, we evaluate our algorithm using real data.

4.1 Cooperative Game Model

In the general form, given a group of consumers and their estimates, the *value* of a coalition (i.e., a group of energy consumers signed up to AgentSwitch) is the expected cost of the aggregated consumption of all its members, taking into consideration the group discount. Due to the workings of the forward energy market, and the penalties applied to unexpected overconsumption or underconsumption in the balancing market, the consumers that will obtain the highest discounts are those whose consumption estimates have a low variance. In this vein, adopting a similar model to [18,1], we construct the expected cost function for a coalition C based on the expected usage profile over a given period (e.g., day, month, or year), as well as the variances (which captures the uncertainty in demand) of all coalition members. More formally, let A be the set of consumers with an AgentSwitch account, and $C \subseteq A$ be a potential coalition. Moreover, let $f_i(S) \sim \mathbb{N}(m_i(S), \Sigma_i(S,S))$ be the predicted consumption profile for consumer $i \in C$ over times S (as in Section 3.1). Now, for coalition C, the mean predicted consumption is:

$$\mu_C = \sum_{i \in C} \int_S m_i(t) dt \; .$$

Furthermore, the variance of the predicted consumption is:

$$V_C = \sum_{i \in C} \int_S \int_S \Sigma_i(t, t') \; dt \; dt'$$

Based on these, the coalition value reflecting the expected cost of C is:

$$v(C) = p \times \mu_C - k\sqrt{V_C}$$

where p is the unit price, and k is a constant parameter that captures the balancing prices. We can replace the integrals above with summations for discrete times S. Given the above cost function for a coalition of homes, it turns out that the minimum cost is achieved if all consumers come together in the *grand coalition* (i.e., the set of all consumers). With this in mind, the objective of AgentSwitch is to divide the value of the grand coalition, i.e., the expected cost, among its members in a fair way.

We start by noting that the presence of any consumer i in a coalition C causes a reduction in the expected cost that the coalition has to pay. This difference is referred to as the *marginal contribution* of that consumer, and is given by $v(C \cup \{i\}) - v(C)$. According to the Shapley value, the fairest way to divide the group discount is as follows. Each consumer receives a share that is equal to his average marginal contribution to all possible subsets of the rest of the consumers. This division satisfies a number of axioms, each of which being a desirable fairness property.

Unfortunately, due to its combinatorial nature, the Shapley value quickly becomes hard to compute as the size of the grand coalition exceeds tens. The state-of-the-art research on addressing this issue [2], proposes an approximation algorithm based on uniform sampling from the marginal contributions. However, in order to establish a worst-case guarantee on the quality of this approximation, the algorithm requires prior knowledge of the minimum and maximum marginal contributions of each consumer. Moreover, this worst-case guarantee is probabilistic, meaning that with some probability it could fail, which, in our setting, would be undesirable given that it might negatively impact the consumers' willingness to sign up to AgentSwitch.

Against this background, in the next subsection, we present a branch-and-bound algorithm for computing the Shapley value of a given consumer. Initially, this algorithm approximates the Shapley value, and establishes a tight bound, both in linear time. After that, it iteratively improves the approximation until it eventually arrives at the exact value.

4.2 Computing the Shapley Value for Group Discounts

Let us consider a set MC^i containing all the marginal contributions of consumer i. In order to overcome the exponential complexity of calculating the Shapley value, which is an average of the elements of this set, we divide MC^i into a number of subsets based on the size s of the coalitions corresponding to each marginal contribution, and denote each such subset as MC^i_s. The rationale behind this division is that it results in a number of subsets linear in the number of consumers. We find the maximal and minimal elements of each MC^i_s in linear time, and approximate the average of each MC^i_s based on its mid-range. This way, the worst-case error (i.e., the distance between the approximated and the actual value) can also be quantified. Setting the complexity of computing the characteristic function aside, this approximation requires a number of operations linear in the number of consumers.

Once the algorithm has computed the initial approximation, and has established the tight bound, it gradually improves both the approximation and the bound until it reaches the actual Shapley value of the consumer. This is done as follows. For each MC^i_s, the algorithm selects what we call a *branching agent*, and divides MC^i_s into two subsets: those elements whose corresponding coalitions *contain* the branching agent, and those whose corresponding coalitions *do not contain* it. Then, for each one of these two subsets, the algorithm finds the maximal and minimal elements, and uses them to approximate the average value in that subset in the same way that it would initially do with MC^i. If more time is available, the algorithm divides each one of the two parts of MC^i_s further based on yet another branching agent, and so on and so forth. By continuing this process, the algorithm reaches a state where MC^i has been divided 2^{n-1} times. In this case the approximated average of each subset would be equal to its actual average.

Table 2 shows a comparison of the performance of our algorithm, and two state-of-the-art algorithms. As the expected cost function mentioned above happens to be *supermodular*, one of the two algorithms that we benchmark against is the polynomial sampling algorithm proposed by [8] for supermodular games. For the sake of also including the exact Shapley value, which has an exponential time complexity, we have considered 25 randomly chosen consumers from a pool of 3000 UK homes. As can be seen, the total number of marginal contributions that our algorithm requires to approximate the Shapley value of all the 25 consumers with an average error of 0.4% is a significantly small number compared to the 25×2^{24} optimal. The next best approximation is Castro et al.'s [2], which requires to evaluate 145% more marginal contributions than our algorithm to approximate with the same precision. Whereas our algorithm does so with 100% confidence on the reported error, Castro et al.'s confidence is 75%, which can only be improved with more samples (i.e., more computation).

Table 2. Comparison of our algorithm and the state-of-the-art, for 25 randomly chosen consumers from a pool of 3000 UK homes.

Algorithm	**No. MC**	**Avg. Error %**	**Confidence**
AgentSwitch	1200	0.4%	100%
Castro et al.	2950	0.4%	75%
Nowel et al.	25×15000	0.4%	75%
Optimal	25×2^{24}	0	100%

In separate tests, given the linear complexity of our algorithm and considering 3000 homes in the grand coalition, our algorithm took an average of 4 seconds to compute the Shapley value of a home. Given these values, it is then possible to specify a tariff for every single customer, along with possible penalties for deviating from the expected consumption.

We next address the issue of making the most of a given (time-of-use, real-time, or group) tariff by finding out which loads can be deferred to off-peak times given only an aggregate measure of power readings from homes.

5 Appliance Disaggregation

Previous work has shown that household electricity data can be disaggregated into individual appliances, therefore enabling suggestions for optimising a household's energy efficiency to be provided to a household's occupants [15]. The aim of such feedback is to provide well-defined actionable suggestions (e.g., shift your five washing machine loads or two dishwasher loads to off-peak times) with clear savings rather than leave it to consumers to decide which appliances they should shift to make any significant savings [4]. However, disaggregating appliances from a household's electricity consumption in the context of AgentSwitch is a challenging problem for the following three reasons. First, the data sampling rate of once every 5 minutes (as obtained from off-the-shelf meters) is less frequent than that required by similar hidden Markov model (HMM) based approaches [6,15]. Second, the data available to AgentSwitch represents the average power demand over the sampling interval, as opposed to the instantaneous power demand, therefore blurring the changes in the observed power demand when an appliance turns on or off. Third, the number and types of appliances within each household are not known. As a result, the methods proposed in previous literature are not applicable, and instead a new approach is required.

Thus, AgentSwitch focuses on the identification of appliances which both consume a large amount of energy and can be deferred to another time of day with minimal inconvenience to the household occupants. Three common examples of such deferrable appliances are the washing machine, clothes dryer and dishwasher, and we shall refer to the use of such appliances as *deferrable loads*. Since each deferrable load has a high energy consumption, they have the advantage that they can be disaggregated from the remainder of a household's energy consumption, despite the low data sampling rate. Given this, we first construct appliance models from individually metered appliances from houses other than those in which disaggregation will be performed, which we refer to as the training phase. Second, the appliance models are used to identify appliance signatures within the aggregate electricity data, which we refer to as the disaggregation phase. The training phase consists of operation detection, followed by feature extraction and model construction, while the disaggregation phase consists of operation detection, followed by feature extraction and operation classification (i.e., identifying the appliance run). The following sections describe each of these processes.

5.1 Operation Detection

The aim of operation detection is to identify pairs of switch events from raw power data, potentially corresponding to an appliance turning on, followed by the same appliance turning off. A switch event, e_n, is defined by an increase or decrease in the raw power, $\mathbf{p}$, within a range of values defined by the appliance model: $p_{min} < |p_t - p_{t-1}| < p_{max}$.

Possible appliance operations are then identified as positive switch events, e_{start}, followed by a negative switch event, e_{end}, separated by a duration within a range defined by the appliance model: $d_{min} < t_{e_{end}} - t_{e_{start}} < d_{max}$.

5.2 Feature Extraction

A number of features are then extracted from the detected operations. In the training phase, these features are used to build the appliance models, while in the disaggregation phase, these features are used to determine how well a detected operation matches an appliance model. We extract the following seven features from each operation: on power, off power, duration, power range, ratio of high to low power readings, minimum energy and number of peaks in power demand. We refer to these features as $x_1, \ldots, x_7$.

5.3 Model Construction

In the training phase, we construct models of each deferrable appliance by fitting known distributions to the output of the feature extraction module by maximum likelihood. We use a two-dimensional Gaussian probability density function (PDF) for the on and off power, since these features are interdependent, and we use a single-dimensional Gaussian PDF for the appliance duration, power range and ratio of high to low power readings. In addition, we use a single-dimensional Gaussian cumulative density function (CDF) for the minimum energy feature, to provide a smooth lower bound. Last, we use a switch function to match the number of peaks in the power demand. We denote the parameters of these six functions as $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_6\}$. In addition, we also extract boundary values for the appliances' power, p_{min} and p_{max}, and the appliances' duration, d_{min} and d_{max}.

5.4 Operation Classification

In the disaggregation phase, to determine whether a potential appliance operation is a deferrable load, we define a likelihood function which describes the similarity between a potential appliance operation and the previously learned appliance model. The likelihood function is defined by:

$$\begin{aligned}\mathcal{L}(e_{start}, e_{end}; \boldsymbol{\theta}) =& f(x_1, x_2; \theta_1) f(x_3; \theta_2) f(x_4; \theta_3) \\ & f(x_5; \theta_4) F(x_6; \theta_5) g(x_7; \theta_6) \end{aligned} \tag{7}$$

where the function f represents the PDF of a Gaussian distribution parameterised by $\boldsymbol{\theta}$, the function F represents the CDF of a Gaussian distribution parameterised by $\boldsymbol{\theta}$ and g represents a function which returns 1 if x_7 is equal to the number of peaks in the model and 0 otherwise.

The likelihood of an operation is compared to a likelihood threshold, L, to determine its classification as a deferrable load: $\mathcal{L}(e_{start}, e_{end}; \boldsymbol{\theta}) > L$. The likelihood threshold was optimised using sub-metered training data from houses other than those in which the disaggregation is being performed.

5.5 Evaluating Load Disaggregation

In order to test the accuracy of the AgentSwitch disaggregation algorithm, we required a data set for which each appliance's power demand is known, since the homes we monitored (as detailed in Sections 3 and 4) did not have appliance level monitoring and hence cannot be used for this test. Instead, the Reference Energy Disaggregation Dataset (REDD) [7] is such a data set, in which both the household aggregate and individual appliance power demands were monitored. We selected three houses which contained a dishwasher, and downsampled the data to 5 minute average power readings to mimic the UK homes we monitored.

We benchmarked the AgentSwitch approach against a state-of-the-art HMM-based approach [15]. Such approaches predominantly use step changes in power demand and state durations as their primary features. In addition, they represent appliances using state transition models, which describe the probability of a transition from one state (e.g. on) to another state (e.g. off) given the observed power readings.

We compare the accuracy of the AgentSwitch and HMM-based approaches using three metrics: precision, recall and F-score. These metrics respectively represent the fraction of detections which were actually loads, the fraction of loads which were detected and a weighted average of the two.

The detection accuracy of the AgentSwitch and the HMM-based approaches using the REDD data set are shown in Table 3. It can be seen that the AgentSwitch approach outperforms the HMM-based approach. This is due to the nature of the data coming from the UK homes, which represents low-granularity average power readings. Consequently, instantaneous increases in the power demand caused by an appliance turning on or off are blurred across consecutive readings due to the averaging. The HMM-based approach is not robust to such blurring, and consequently suffers a loss in accuracy. Conversely, the AgentSwitch approach utilises additional features even when the on and off event signatures are blurred, and as a result is robust to such effects. So far, we have considered homes in isolation and how to provide feedback to consumers in order to help them maximise their savings given a time-of-use electricity tariff. In our test homes, the savings from deferring the identified loads were not major (limited to less than £50 a year) due to low usage of deferrable appliances at peak times electricity. For

Table 3. Accuracy metrics of dishwasher detection

Approach	Precision	Recall	F-score
AgentSwitch	0.687	0.600	0.596
HMM	0.568	0.438	0.456

higher usage levels at peak times (e.g., large families or flats) and as energy prices rise and more complex time-of-use pricing (e.g., real-time or critical peak pricing) are put in place to combat peaks in demand, we expect such savings to increase substantially. A key step, however, to achieve these savings is to ensure consumers can understand and trust the recommendations provided by AgentSwitch. To support mechanisms and interfaces to do so, we next introduce our provenance-tracking service.

6 Provenance in AgentSwitch

In order to ensure consumers understand and build confidence in the suggestions given by AgentSwitch (as detailed in the previous sections), it is important to provide them with means to track the flow of data and decisions made throughout the system. For example, using such traces, users (or other services acting on their behalf) may be able to identify faulty sensors, incorrect assumptions about yearly consumption, or justify changes to daily routine to make significant savings.

As shown in the previous sections (see Figure 1) various types of data come into/out of AgentSwitch modules and get consumed/transformed at the same time (e.g., electricity consumption data from a third-party provider, post code from user input, tariffs from uSwitch). Thus, the quality of recommendations given by AgentSwitch depends on the quality of data it receives from different sources and the performance of its individual components. As recommendations from AgentSwitch might potentially lead to financial gains or losses, it is crucial to be able to identify the origin of error once a recommendation is deemed to be inaccurate. This, however, is challenging due to the multiple paths of (data) dependencies inherent to such a complex system.

In order to address this issue, the chains of dependencies that lead to a recommendation, i.e., its provenance,[11] were fully tracked in AgentSwitch. Such information enables a systematic approach to pinpointing the sources or agents responsible for the errors and to auditing the data produced within the system. In more detail, the integration of provenance tracking in AgentSwitch is described in Section 6.1. Section 6.2 then demonstrates various use cases of provenance in the application.

6.1 Tracking provenance

Provenance information in AgentSwitch is modelled using the PROV Data Model (PROV-DM) [10] being standardised at the World Wide Web Consortium. In this model, the three main types of element are: entity — a physical or digital thing (e.g., a piece of data), activity — something that occurs and generates or changes entities, and agent — something that bears some form of responsibility for an activity. In addition to these elements, various kinds of relationships between entities, activities, and agents can

[11] Provenance is defined as a record that describes the people, institutions, entities, and activities involved in producing, influencing, or delivering a piece of data [10].

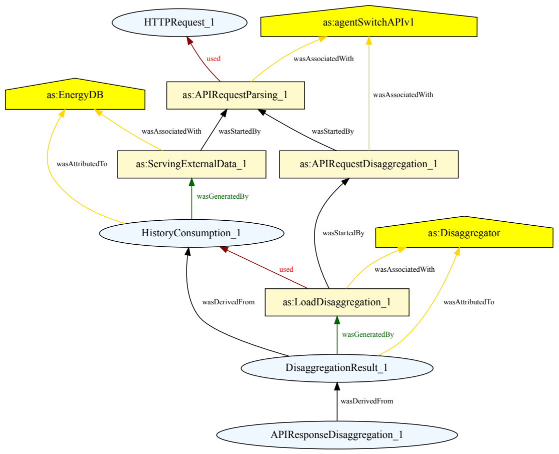


Fig. 4. Provenance graph captured from a disaggregation API call to AgentSwitch. Boxes are activities that took place, pentagons the agents controlling them, and ovals the data entities consumed/produced.

be captured in PROV-DM (see [10] for more details). Figure 4, shows a visual representation of the provenance tracked in a disaggregation API call (to the load detection module described in Section 5). It shows that the disaggregation API response (`APIResponseDisaggregation_1`, a data entity) was derived from the disaggregation result (`DisaggregationResult_1`) attributed to a load disaggregation script in Matlab (`as:Disaggregator`). The activity `as:LoadDisaggregation_1`, in turn, used the electricity consumption data `HistoryConsumption_1` from our Energy DB (`as:EnergyDB`) to produce the load disaggregation result.

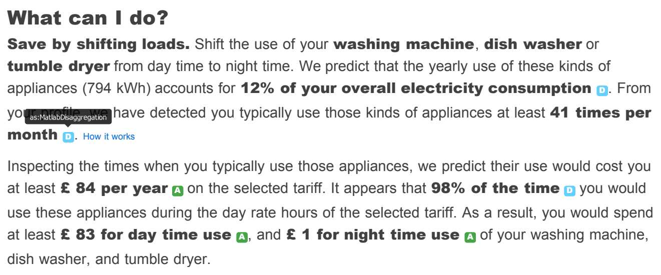
Provenance tracking was integrated independently into each individual component of AgentSwitch. When a component passes its result to the next, the captured provenance also accompanies the data, such as from the AgentSwitch API to the web front-end (see Figure 1). By so doing, the provenance graph for a recommendation evolves at each individual component, and the complete provenance is eventually made available to the web front-end for further use.

6.2 Exploiting provenance

Having had the provenance of activities in AgentSwitch captured, it was possible for the application to inform end users about its underlining processes in a number of ways:

Responsibility management: In a similiar vein to [9], the responsibilities of contributing agents in a recommendation can be queried from its provenance. An example of which is presented in Figure 5 where parts of the load shifting advice are tagged with colour labels indicating the main modules responsible for the respective data. For instance, the AgentSwitch API (A) produced the cost calculation while the load disaggregation service (D) provided the appliance usage information in Figure 5. We can also show that the costs were calculated from the selected tariff and the appliance usage information.[12]

[12] In this respect, a user interface for drilling down provenance graphs that offers an information representation suitable to end users is left as future work.



What can I do?

Save by shifting loads. Shift the use of your **washing machine**, **dish washer** or **tumble dryer** from day time to night time. We predict that the yearly use of these kinds of appliances (794 kWh) accounts for **12% of your overall electricity consumption** D. From your profile, we have detected you typically use those kinds of appliances at least **41 times per month** D. How it works

as:MatlabDisaggregation

Inspecting the times when you typically use those appliances, we predict their use would cost you at least **£ 84 per year** A on the selected tariff. It appears that **98% of the time** D you would use these appliances during the day rate hours of the selected tariff. As a result, you would spend at least **£ 83 for day time use** A, and **£ 1 for night time use** A of your washing machine, dish washer, and tumble dryer.

Fig. 5. The screen-shot of a recommendation with colour tags (i.e. A and D) generated from tracked provenance indicating the agents that were mainly responsible for the various pieces of data shown.

Uncertainty management: In addition to the dependencies between data entities, the provenance was tracked for meta-data about those entities, such as the time span of consumption data, the uncertainty of annual consumption predictions (as presented in Section 3) and load disaggregation results (as described in Section 5). Those uncertainty metrics can be propagated and combined in provenance graphs all the way to the final recommendation, allowing the estimation of its confidence level, which is then presented to the user. This may help end users understand the risks associated with recommendations and therefore adjust their tariff selections as suggested by AgentSwitch[13].

Beside the above benefits to end users, the captured provenance also showed us visually how AgentSwitch actually works, similar to a debug logging system to software developers. It allows the identification of the flow of executions in the system, the points where individual components interface with one another, as well as to manage data dependencies.[14]

7 Conclusions

In this paper we presented the core modules of AgentSwitch, an agent-based platform to enable smart energy tariff selection for domestic energy consumers. We focused on the key advances to the state of the art that the development of modules for AgentSwitch have brought about, including novel extensions to BQ, a novel algorithm for NIALM, and scalable algorithms to compute the Shapley value for energy purchasing collectives. Moreover, we showed how these modules can be packaged into an accountable information infrastructure underpinned by a novel provenance tracking service that allows users to track the flow of their data through individual modules and third-parties. While a working prototype of AgentSwitch, using live data from homes, has been completed, future work will look at integrating other sensor data (weather, consumer preferences) into the prediction and group buying algorithms in order to improve accuracy and the resulting savings.

[13] Again, a drilling-down interface will also be useful in this context, allowing end users to see where such uncertainty stemmed from. At the same time, it may help them gain an understanding of the inner workings of the system and potentially increase their trust in it.

[14] In fact, provenance graphs produced by early versions of AgentSwitch had revealed inefficient code by showing duplicate (thus redundant) API calls, prompting us to improve the system.

Acknowledgments

This work is supported by the EPSRC-Funded ORCHID Project EP/I011587/1. We also thank uSwitch for access to their API.

References

1. E. Baeyens, E. Bitar, P. Khargonekar, and K. Poolla. Wind energy aggregation: A coalitional game approach. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 3000 –3007, dec. 2011.
2. J. Castro, D. Gómez, and J. Tejada. Polynomial calculation of the shapley value based on sampling. *Comput. Oper. Res.*, 36(5):1726–1730, May 2009.
3. P. Diaconis. Bayesian numerical analysis. In S. G. J. Berger, editor, *Statistical Decision Theory and Related Topics IV*, volume 1, pages 163–175. Springer-Verlag, New York, 1988.
4. J. Froehlich, L. Findlater, and J. Landay. The design of eco-feedback technology. In *Proc. CHI '10*, pages 1999–2008. ACM, 2010.
5. M. Kennedy. Bayesian quadrature with non-normal approximating functions. *Statistics and Computing*, 8(4):365–375, 1998.
6. J. Z. Kolter and T. Jaakkola. Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation. In *International Conference on Artificial Intelligence and Statistics*, La Palma, Canary Islands, 2012.
7. J. Z. Kolter and M. J. Johnson. REDD: A Public Data Set for Energy Disaggregation Research. In *ACM Special Interest Group on Knowledge Discovery and Data Mining, workshop on Data Mining Applications in Sustainability*, San Diego, CA, USA, 2011.
8. D. Liben-Nowell, A. Sharp, T. Wexler, and K. Woods. Computing shapley value in supermodular coalitional games. In J. Gudmundsson, J. Mestre, and T. Viglas, editors, *Computing and Combinatorics*, volume 7434 of *Lecture Notes in Computer Science*, pages 568–579. Springer Berlin Heidelberg, 2012.
9. S. Miles, S. Munroe, M. Luck, and L. Moreau. Modelling the provenance of data in autonomous systems. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'07)*, pages 1–8. ACM, 2007.
10. L. Moreau and P. Missier. PROV-DM: The PROV Data Model. Candidate Recommendation, 2012.
11. OFGEM. The retail market review - findings and initial proposals, 2011.
12. OFGEM. Press release: projected tightening of electricity supplies reinforces the need for energy reforms to encourage investment., October 2012.
13. A. O'Hagan. Bayes-Hermite quadrature. *Journal of Statistical Planning and Inference*, 29:245–260, 1991.
14. M. A. Osborne, R. Garnett, S. J. Roberts, C. Hart, S. Aigrain, and N. Gibson. Bayesian quadrature for ratios. *Journal of Machine Learning Research - Proceedings Track*, 22:832–840, 2012.
15. O. Parson, S. Ghosh, M. Weal, and A. Rogers. Non-intrusive Load Monitoring using Prior Models of General Appliance Types. In *26th AAAI Conference on Artificial Intelligence*, Toronto, Canada, 2012.
16. C. E. Rasmussen and Z. Ghahramani. Bayesian Monte Carlo. In S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15. MIT Press, Cambridge, MA, 2003.
17. C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.
18. H. Rose, A. Rogers, and E. H. Gerding. A scoring rule-based mechanism for aggregate demand prediction in the smart grid. In *The 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, June 2012.
19. L. S. Shapley. A value for n-person games. *Contributions to the theory of games*, 2:307–317, 1953.
20. US Department of Energy. The Smart Grid: An Introduction, 2007.

Exploring the challenge of learning well from a sparse training set

Jeffery Raphael[1] and Elizabeth Sklar[1,2]

[1]The Graduate Center and [2]Brooklyn College
The City University of New York, New York, USA
jp_raphael@yahoo.com,sklar@sci.brooklyn.cuny.edu

Abstract. A classic challenge in machine learning is to provide an effective training set for the learner. When a human generates the content of the training set, such as a log of moves from a game or clickstream data from an interactive tool, the problem becomes more acute. The work presented here explores the relationship between the quality of a training set and the lessons learned by agents trained on that set. Two different learning techniques and two different state space reduction methods are applied to learning in a classic game environment (Tetris). Results demonstrate, as expected, that these variations affect the policy acquired by the learner. While nothing unexpected was uncovered from a machine learning standpoint, the exercise has proven useful within the realm of learning from a human teacher. This paper presents the results, analyzed using a variety of metrics, and proposes a methodology for predicting and assessing the performance of a learner based on its learning environment.

1 Introduction

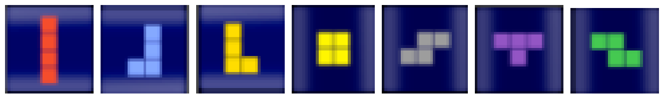
A classic challenge in machine learning is to provide an effective training environment for a learner. Our work focuses on learning how to behave in dynamic interactive environments. Our long-term goal is different from traditional machine learning work in such environments, where most research concentrates on trying to train an *expert*. Instead, our aim is to train *a population of agents* that collectively represent the behavior landscape, from novice to expert, exhibiting the full range of common and peculiar traits. Thus we want to learn to imitate a human trainer and capture characteristics of her behavior.

In the case of supervised learning from static instances, that training environment is typically a set of data points, called *exemplars*, which an agent learns to imitate or classify. A human can generate such a data set by logging actions in a game (i.e., "moves") or interactive tool (i.e., "clickstream"). But here the problem becomes more acute. For any even moderately complex environment, a single human can only provide a limited set of exemplars, both in terms of quantity and with respect to coverage of the complete landscape of possible actions. Related work has explored the generation of comprehensive training sets by aggregating logs of activity of multiple humans [1, 2]. The work presented here investigates a different side of the same question: if it is not possible to create a more robust training set, then how much can we expect a learner to absorb? We examine the question using two different learning techniques and two different state space reduction methods, within a classic game environment (Tetris). Our results demonstrate that these variations indeed affect the policy acquired by the learner. While nothing unexpected was uncovered from a machine learning standpoint, the exercise has proven useful within the realm of learning from a human teacher and proposes metrics and a methodology for assessing and predicting the performance of a learner based on its learning environment.

This paper is organized as follows. First, we describe our training environment and explain the state space reduction methods explored in our study (Section 2). Second, we outline the learning techniques employed (Section 3). Next, Section 4 presents the results of our study. Section 5 highlights related work. Finally, we close with directions for future exploration of our research question.

2 Training environment

This study uses the game of *Tetris* as a training environment. Tetris was chosen because it exhibits the following desirable characteristics for a machine learning testbed. Tetris employs a real-time, dynamic and stochastic, but replicable, environment; only the "next piece" is chosen randomly, and once that is known, all other elements of an episode can be replayed deterministically. User input is facilitated through a small set of possible actions, which means that there is a tractable set of moves to learn. Visual feedback is provided, in response to user actions and predictable environmental state changes. Finally, it is a game, so user behavior is directed and easily motivated; and it is not hard to find human subjects willing to generate data sets by playing games.



label:	*I*	*J*	*L*	*O*	*S*	*T*	*Z*
exemplars:	1081	500	437	773	509	456	519
scan size:	4	3	3	4	3	3	3

Table 1. Training set. See text for explanation of the last two rows.

There are a few differences between traditional Tetris and the version used in our study. All adaptations were made to simplify the learning environment. First, in our version, there are no "levels". In traditional Tetris, players move through increasingly difficult levels of play by earning more points; at each level, the speed with which the tetrominoes falls increases. Second, there are no bonus points given for completing multiple lines. In our version, a player's score is simply the number of lines completed. Finally, in our version, the player does not have access to the "next" piece.

Tetris move	*label*
move Left	l
move Right	r
Drop (fall to bottom of board)	d
rotate Clockwise (90°)	c
rotate Anti-clockwise (90°)	a

Table 2. Legal moves

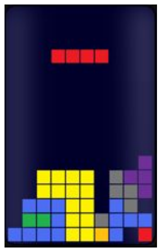
For our study, a limited training set was generated by recording the moves of one player completing three games of Tetris[1]. The three games provided 4275 exemplars, each consisting of a board configuration and response (move made by player). These were stored in a database, using a "game string" format that encodes the label for the current Tetromino piece (middle row of Table 1) and player's moves (right column of Table 2). Any game can be replayed using the game string logged for that game. Table 1 shows the number of exemplars.

2.1 State Space Reduction

The standard Tetris board consists of 200 squares arranged in a 10×20 grid. Assuming that any cell in the grid could be marked as either occupied (by 1/4 of a tetromino) or unoccupied, a grid this size has 2^{200} configurations. Note that this includes board configurations that cannot arise during normal game play, such the top row (line) being fully occupied if all the cells below it are not occupied. For our study, we defined a *state* as consisting of the current board configuration and the current tetromino being played. With seven different tetrominoes, there are a maximum $7 \times 2^{200} \approx 2^{203}$ possible states. Constructing a scheme to represent the board as well as manage its large state space is always an issue when dealing with Tetris. We selected two state space reduction methods for their simplicity and popularity amongst Tetris solutions, as illustrated in Figure 1.

- The *Contours* method encodes the board as a vector of height differences between consecutive columns. The maximum difference between two columns is set to four units. Four units was chosen because the longest tetromino, the I-tetromino, is four units long. With Contours, the board is transformed to a 9-dimensional vector $V = \langle d_1, d_2, \ldots d_9 \rangle$ where $-4 \leq d_i \leq 4$. Thus, Contours yields $7 \times 9^9 \approx 2^{30}$ states.
- The *Heights* method, on the other hand, represents the board using a 10-dimensional vector composed of the height of each column. Hence, Heights produces a $V = \langle h_1, h_2, \ldots h_{10} \rangle$ where $0 \leq h_i \leq 20$ yielding $7 \times 21^{10} \approx 2^{43}$ states.

[1] The results reported here were based on games played by the first author.



$$\text{Contours} = \langle 2, 2, 0, 0, 0, -3, 3, 0, 1 \rangle$$

$$\text{Heights} = \langle 1, 3, 5, 5, 5, 5, 2, 5, 5, 6 \rangle$$

Fig. 1. Tetris board represented as a vector using two state space reduction methods

We employed an additional technique, which we call *Scanning*, to further shorten the size of the vector that represents the board. Scanning uses a variable-sized input vector instead of a fixed-size vector. The size of the vector, or scan size, depends on the tetromino. The bottom row of Table 1 lists the tetrominoes and their scan sizes. Figure 2 shows the seven scans that result from processing the sample board in Figure 1. How a particular scan is selected depends on the learning mechanism, and this is discussed in Section 3.3.

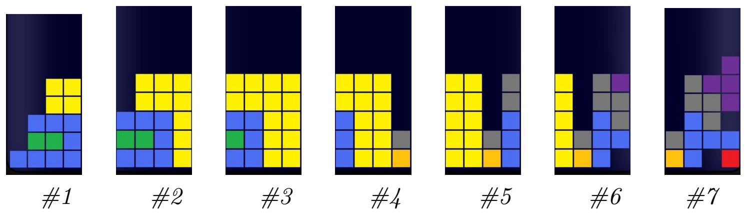


Fig. 2. Scans of the sample board in Figure 1

3 Learning methods

For our study, we compared two machine learning techniques: *artificial neural networks (ANN)* using *backpropagation* to adjust the weights, and *Learning from Demonstration (LfD)* using *Gaussian mixture models (GMM)* and *Expectation Maximization (EM)* to adjust the statistics. As above, we also compared two state space reduction techniques (*contours* and *heights*) and applied *scanning* in both cases. We trained four agents. Each employed a different combination of machine learning technique and state space reduction method, as shown in Table 3.

Each agent comprised 7 strategies (i.e., ANN or GMM), one for each type of tetromino—so in all, 14 ANNs and 14 GMMs were trained. Each agent was evaluated to determine how well it had absorbed the information in its training set and how well that knowledge generalized to a broader state space. Performance was measured using three common Tetris metrics: **score** (number of lines completed), **board height**, and number of **tetrominoes** played; plus two metrics that describe the compactness of the board: **density** and **holes** (explained below).

	contours	*heights*
artificial neural network (ANN)	ANNC	ANNH
learning from demonstration (LfD)	LfDC	LfDH

Table 3. Tetris Agents

There are four major steps that an agent must take to go from the point when it receives the game state (tetromino plus board configuration) to selecting its action (move). The four steps are:

1. Assess the board using the scanning process
2. Encode the scans using contour or height vectors
3. Process each vector using the appropriate ANN or LfD system
4. Select an action, which is expressed as the number of counterclockwise rotations and a column.

After the agent has selected its action, the action is converted into a sequence of commands that the Tetris engine will execute on the tetromino in play. Once this is done, a new tetromino is put in play and the process repeats.

3.1 Artificial Neural Networks

This section describes the parameters that all fourteen ANNs had in common. All the ANNs had three layers (input, hidden and output), and each layer had a bias node. All neurons used a logistic function, $f(x) = 1/(1+e^{-\beta x})$, as the activation function. Network weights were initialized with random values between -0.05 to $+0.05$ inclusive. The number of input nodes varied for each tetromino-based ANN and is discussed below in Section 3.3. The input values were normalized using the *Min-Max normalization formula*. The *mean squared error (MSE)* was used to calculate an ANN's error. Each output neuron represents an action, where that action is both the orientation (expressed as the number of counterclockwise rotations) of the tetromino and the column (within a scan) in which it should be placed. Since the output neurons return a value within the interval $[0, 1]$, the output neuron with the highest output value is set to 1 and all other output neurons are set to 0. The tetromino's scan size, number of unique orientations, and maximum width determine the number of output neurons. For example, the I-tetromino has a scan-size of four, two unique orientations, and a maximum width of four units. Therefore, given any scan of size four, there are five possible solutions; and so there must be five output neurons in the ANN that handle the I-tetromino, one for each possible solution (see Figure 3).

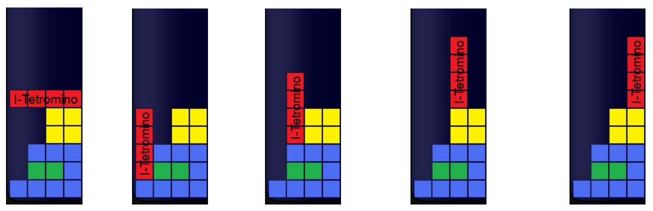


Fig. 3. Sample scans of width four (scan #1 from Figure 2) showing all possible moves for *I*-tetromino

3.2 Learning from Demonstration

This section describes the characteristics that all fourteen LfD systems had in common. The components, or Gaussian distributions, in each LfD system represented different actions (see Table 2). If an input scan was found to belong to a particular component, then it meant that the tetromino should be placed in the column and orientation that that particular component represented. The tetromino's scan size and number of unique orientations determine the number of components needed. Using the same example illustrated in Figure 3, an LfD system designed to handle the *I*-tetromino needs five components. Given a scan (encoded as an input vector), the LfD system returns a *probability density function (PDF)* value for each of its components. This PDF value is a measure of membership of the input vector within each Gaussian distribution. The component with the highest PDF is selected as the agent's response.

3.3 Applying scanning

The main difficulty with applying the scanning technique is to determine which segment of the board to place the tetromino in. Since the Tetris board is 10 columns wide and the scans are only 3 or 4 columns wide, there are 8 choices of which 3-column vector to select and 7 choices of which 4-column vector to select. The choice is made by assessing the board, from left to right, starting at the first column and encoding n columns at a time (where n is the scan size). Each scan is sent to the ANN or LfD system for processing. How a particular scan is selected depends on the learning mechanism.

For LfD systems, the PDF is a natural criteria to measure membership [3]. The PDF can be interpreted as the level of confidence that the LfD system has in an action given a state (in this case a scan). Consequently, the scan which results in the highest PDF is selected.

For ANN systems, the networks as we have defined them do not reflect any level of confidence in a particular response. To work around this issue, ANN agents always select the first scan as their default solution and then decide to select one of the other scans with probability ϕ[2]. This approach had dubious

[2] The results presented here used a value of $\phi = 0.75$.

results, as discussed in Section 6, and different methods of selecting the scan for the ANN are being explored in follow-up work. The number of inputs to the neural network for any tetronimo is thus equal to the width of the corresponding scan for that tetromino.

3.4 Training

The neural networks were trained using the Backpropagation algorithm [4], with weights updated after each exemplar was fed to the network. All the neural networks had the same momentum factor of 0.2, but different learning rates (between 0.20–0.75). For the LfD systems, clustering occurred once, after all demonstrations were collected. The clustering (EM) algorithm that was used to calculate the GMM parameters was from [5].

4 Results

The agents were evaluated in several ways:

– **Grade on training states.** Each agent was shown the same set of exemplars on which it was trained, and a "grade" was assigned based on how many actions were chosen correctly (i.e., matched the human trainer's action).
– **Play training sequence.** Each agent played the same three games that the human trainer played. In other words, each agent played three games in which they were presented with the same sequence of tetrominoes that the human trainer was.
– **Play random sequence.** Each agent played 50 games with a sequence of randomly selected tetrominoes. The human trainer did not play these games, so the trained agents' performance is compared to an agent that selects its moves randomly.

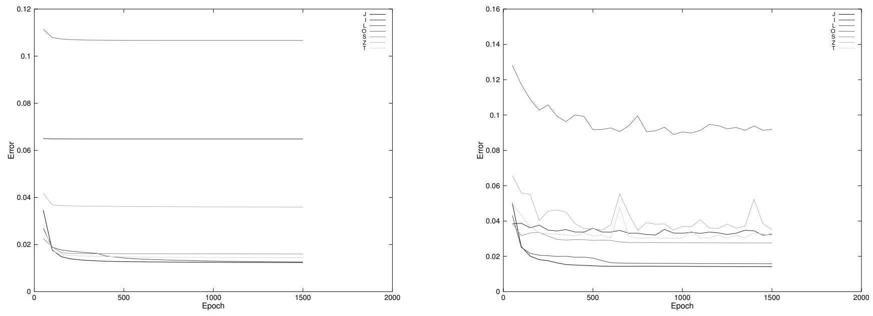
The following metrics were tallied for each agent for the latter two evaluation methods:

– **score**, or average score in the case of the 50-game series
– **height** of the board, as the games progressed for the 50-game series
– number of **tetrominoes** played
– **density** of the board, at the end of the evaluation series
– number of **holes** as the games progressed for the 50-game series

Density is the ratio of the number of filled spaces to the total area. Holes are cells on the board that cannot be reached.

4.1 Results: Grade on training states

We look at the **Grade** assessment in two ways. First, we examine how the error rate dropped during training. The training errors for the ANNs are shown in



(a) ANN contours (b) ANN heights

Fig. 4. Error rates for ANN agents

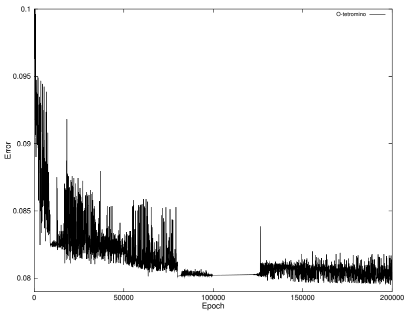
Figure 4. All the ANNs were trained for 1500 epochs. During training, there is an initial decline in the error rate and then it levels off and fails to improve any further, particularly with the contour ANNs. The contour ANN for the J-tetromino showed very little change throughout its training session. For the majority of the ANNs, the error rate remained between 0.01 and 0.07 except for the ANN which handled the O-tetromino. The O-tetromino ANN had the highest error rate (for both state space reduction methods). In order to test the idea that the O-tetromino ANNs would benefit from a longer training session, we tried retraining the height ANN for $200,000$ epochs, as shown in Figure 5a. There was a gradual improvement in the error rate until around the $75,000$th epoch where the error rate reached 0.082. At that point, the error rate continues to fluctuate between 0.079 and 0.081 but does not show any significant improvement for the remainder of the training session.

Another factor which may effect network convergence, besides length of training, are the initial weights. Figure 5b shows the error rates for training 100 instances of the contours L-tetromino ANN for 1500 epochs. At the beginning of each training session, the weights were initialized to random values within the interval $[-0.05, +0.05]$. The results clearly show that, although the initial weights do effect the final network error, the effect is not significant. The error rates ranged from 0.004 to 0.017.

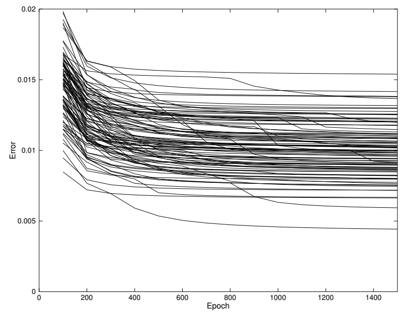
Second, we evaluated the agents by showing them each training state and tallying the percentage of actions they predicted correctly. These overall grades are shown in Table 4. The LfD contour earned the highest grade. In fact, both contour methods (LfD and ANN) outperformed both the height methods.

4.2 Results: Play training sequence

Table 5 contains the results for the agents playing the same sequence of tetrominoes as the three games played by the human trainer. It is important to recognize that this does not produce exactly the same set of states as the training set (i.e.,



(a) O-tetromino trained for $200K$ epochs



(b) 100 training instances

Fig. 5. training stuff

LfD contours	LfD heights	ANN contours	ANN heights
89%	81%	88%	77%

Table 4. Overall grade.

as assessed above), because any time an agent chooses an action different from the human's action, the board configuration will diverge. Thus, even though the sequence of tetrominoes shown was the same, the board configuration component of the state space differed.

Clearly, according to the three metrics shown in the table (score, number of tetrominoes played, and density at the end of the game), none of the agents learned to play as well as the human trainer. The scores for density are not too far behind, however. Indeed, for *game 3*, three of the agents outperformed the human according to the density measure. But the other, more traditional Tetris evaluation metrics tell another story. The LfD contours agent still outperforms the others, though we note that the ANN heights agent is close. According to the number of tetrominoes metric, ANN heights outperformed LfD contours in two of the three games, and its standard deviation is much lower (even though we are only looking at 3 games).

4.3 Results: Play random sequence

The final set of results shows each agent playing 50 games with randomly generated sequences of tetrominoes. Note that the human trainer did not play these 50 games, so we can only compare the agents' performance in relation to the trainer's performance on the 3 training games.

We start with the best result. Figure 6 shows the average number of holes on the board after each tetromino is played across all fifty games. The average number of holes on the board is the only metric that showed a marked difference between the two categories of agents. The LfD agents clearly outperformed the

	human trainer	LfD contours	LfD heights	ANN contours	ANN heights
game 1					
score	219	0	0	0	2
tetrominoes	590	25	25	31	32
board density	0.77	0.48	0.48	0.56	0.49
game 2					
score	185	7	0	0	1
tetrominoes	506	53	24	25	39
board density	0.79	0.68	0.46	0.48	0.70
game 3					
score	450	1	5	0	1
tetrominoes	1129	26	46	32	32
board density	0.53	0.43	0.61	0.58	0.56
mean (stdev)					
score	284.67 (117.73)	**2.67** (3.09)	1.67 (2.36)	0 (0)	1.33 (0.47)
tetrominoes	741.67 (276.02)	**34.67** (12.97)	31.67 (10.14)	29.33 (3.09)	**34.33** (3.3)
board density	0.70 (0.12)	0.53 (0.11)	0.52 (0.07)	0.54 (0.04)	**0.58** (0.09)

Table 5. Playing the training sequence.

ANN agents in reducing the number of holes, hence they were better at packing the tetrominoes. In the begining of the games, prior to the 20th tetromino, both LfDC and LfDH's performance are almost identical to the human player. And later, around the 48th tetromino, LfDC does just as well as the human player in maintaining a small number of holes.

	random player	LfD contours	LfD heights	ANN contours	ANN heights
mean (std) over 50 games					
score	0.02 (0.14)	**2.16** (2.66)	1.28 (1.59)	0.82 (1.01)	0.90 (1.15)
tetrominoes	21.74 (3.27)	**34.98** (7.86)	33.56 (6.93)	31.68 (5.64)	32.08 (5.21)
board density	0.40 (0.06)	0.55 (0.07)	**0.56** (0.08)	0.55 (0.08)	0.55 (0.08)

Table 6. Playing the 50-game evaluation sequence.

Overall, the agents did better than the random agent but worse than the human agent in every metric except for average number of holes. The LfD agents had the two highest scores, shown in Table 6. LfDC averaged 2.16 lines completed per game while LfDH averaged 1.28 lines per game. The two ANN agents had close to the same average score, about one point per game. The inability of the agents to complete lines and therefore manage the height of board was also evident in the average height of the board after each tetromino was played, shown in Figure 7. All the agents exhibited the same kind of *stacking* behavior. As the game progresses, the height of the stack grew linearly.

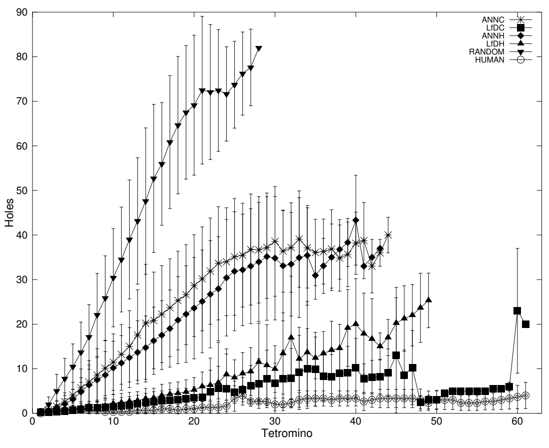


Fig. 6. Avg. number of holes over 50 games played by agents and 3 games played by human. Error bars show one deviation above and below mean.

4.4 Analysis

We analyze our results in two ways. First, we present a domain-specific analysis of the playing strategy learned by the agents. Second, we propose a more general method of analyzing the amount of the state space learned and using that to predict the performance of a learning agent.

Agent Playing Strategy. After studying the results, we concluded that the agents did learn how to play Tetris, but developed a strategy to minimize holes rather than to complete lines. In other words, the agents placed the tetrominoes so that they fit together snugly. In Figure 6 we see that the difference in the number of holes generated by the agents versus random agent is statistically significant. The agents are not completing lines, instead they are steadily building the tetromino stack as they play. This behavior is further supported by the results shown in Table 6 and Figure 7.

There are two possible culprits for this playing style: state representation and the quality of the training data set. Figure 8 shows a T-tetromino and all the possible ways it could be played in a sample three-column scan. A typical human player would most likely select the second example as the correct solution. Other than how to place the T-tetromino, the second example does not reveal much else about the player's long or short term goals. An analysis of the input vectors in the training data showed that tetrominoes were often played in the same manner over and over again with very little variety. For example, it was rare to find the O-tetromino placed in a position that was not a flat space.

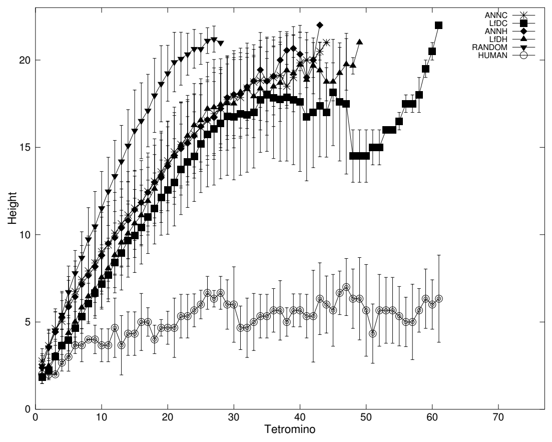


Fig. 7. Avg. height as game progressed over 50 games played by agents and 3 games played by human. Error bars show one deviation above and below mean.

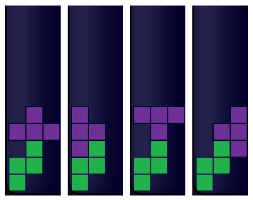


Fig. 8. Possible placement of T-tetromino in scans of size three

State Space Coverage. A closer examination of the exemplars revealed that the agents were actually exposed to only a small portion of the state-space. Table 7 shows a dissection of the training data set for the agents that used the contours method. The LfDC agent was the best scorer, yet on average the training data set represented only 11.87% of the state-space (for heights, the average training set represented 0.43%). This comes as no surprise, a good Tetris player is less likely to explore alternative maneuvers which may not lead to a high score. Evidently, the training examples from the human teacher tended to be refined and not representative of the broader state-space landscape.

We can quantify the *expected amount of knowledge acquisition*. In doing so, we are able to compare the agents' actual results to how well they *should* have performed given the training data set coverage. This *expected performance (EP)* is formulated as:

$$EP = h \times g \times \sigma$$

where h is the human result (for a particular metric), g is the agent's overall grade for the same metric (shown in Table 4), and σ is the average percentage

Tetromino	State-Space	Exemplars in Training Set	Coverage
I	3645	302	8.29%
J	486	55	11.32%
L	486	51	10.49%
O	2187	187	8.55%
S	243	37	15.23%
T	486	58	11.93%
Z	243	42	17.28%

Table 7. State space coverage by tetromino type for *contours*

of the state space that was present during training. For example, the expected performance for the score of the LfDC agent is as follows. The LfDC agent had an overall grade of 89% and its training data set covered 11.87%. Thus, its EP for score was $(284.67 \times 0.89 \times 0.1187) = 30.07$. Table 8 contains the EP values for metrics computed earlier.

	LfDC	LfDH	ANNC	ANNH
score	30.07	0.98	29.74	0.94
tetrominoes	78.35	2.56	77.47	2.44
board density	0.074	0.0024	0.0731	0.0023

Table 8. Expected performance (EP)

The difference between an agent's expected performance metric and actual metric gives us a method of assessing the success (or failure) of the training technique that is independent of the coverage of the training set. Subtracting the expected performance from the actual result produces this expectation comparison, which we refer to as *outlook*:

$$outlook = EP - actual$$

A negative value for outlook tells us that an agent performed worse than expected (i.e., the outlook for training using this method is negative or pessimistic), while a positive outlook means that the agent performed better than expected (i.e., a positive or optimistic outlook). Table 9 contains the outlook values for scoring, tetrominoes played and board density. Thus for the LfDC agent, its outlook is negative (-27.40). However, the *heights* agents (LfDH and ANNH) demonstrated a *positive outlook*. Despite having access to a smaller percentage of the state space, the heights agents performed better than expected across all metrics. This is also the case for the board density metric for all agents, i.e., all agents performed better than expected in the board density metric.

	LfDC	LfDH	ANNC	ANNH
score	-27.40	0.69	-29.74	0.39
tetrominoes	-43.68	29.11	-48.14	31.89
board density	0.46	0.52	0.47	0.58

Table 9. Outlook values

5 Related Work

Learning from human-generated data encompasses a variety of supervised learning methods. We describe several in this section and then explain how our work compares.

Harter & Kozma [6] used a feedforward backpropagation neural network to solve a reduced version of Tetris. Their implementation of Tetris only had a width of 5 units and used 3 different pieces. The neural network was fed the piece type and the board's contour and produced the column to place the piece in and how many times to rotate it. They encrypted the piece type and board contour into a 12-bit input pattern. Human players were used to extrapolate proper piece placement and to generate training data for the neural network. In order to test the solution, the agent was given a random sequence of 10 pieces to place. Performance was evaluated by examining the density and height of the board after placing all 10 pieces. The agent was given 100 random trials. The authors found that humans outperformed the agent and believed it was due to changes in playing strategy that the neural network could not replicate.

Chernova & Velosa [3] developed an interactive policy-learning algorithm where an agent is able to actively request demonstrations when needed. They utilized GMMs as a mapping function to derive a policy. Each GMM, with multiple Gaussian components, represented a single action. They were able to reduce the number of demonstrations the agent required to learn a task, thus improving training time and minimizing teacher effort. To test their algorithm, they trained a Sony Aibo robot to navigate a circular path. The Aibo used its infrared sensor, located in its head, to create a three-dimensional vector representation of its surroundings. Chernova & Velosa [3] compared their algorithm to an agent that learned to navigate the path using a variant of reinforcement learning called Prioritized Sweeping [7]. They found that the reinforcement learning method took much longer than the demonstration-based method.

Knox & Stone [8] developed the "Training an Agent Manually via Evaluation Reinforcement" (TAMER) framework. The TAMER framework provides a scheme for interactively training an agent using *shaping*. In shaping, the trainer only observes the agent and provides simple feedback about each action the agent has taken. Knox & Stone [8] evaluated their TAMER framework using two domains: Tetris and Mountain Car. They used 28 human trainers, 9 for Tetris and 19 for Mountain Car. They compared their Tetris TAMER agent to other Tetris agents developed in [9–12] and their Mountain Car agent to two different $Sarsa(\lambda)$ agents. Knox & Stone [8] found that the Tetris TAMER agent learned an optimal policy much faster than the other agents. The Tetris TAMER agent reached its peak performance by the third game compared to over a hundred games for the other agents. In the Mountain Car experiments, the TAMER agent, on average, outperformed both $Sarsa(\lambda)$ agents for the first five episodes.

We can compare our methodologies to each of these techniques. The LfD systems devised in Chernova *et al.* [3, 13] developed policies interactively. We opted to develop a policy after all the demonstrations had been collected. We also adopted the approach of [3] to LfD systems and utilized GMMs as mapping functions. Knox & Stone [8] describe more formal methods of human agent interactions. Finally, Harter & Kozma [6] did some work that paralleled our work with ANNs and Tetris.

6 Discussion

In this work, we developed four agents to play the game of Tetris. Two of the agents used LfD to learn, while the other two used ANNs. In combination with comparing the performance of these two different learning algorithms, we also compared two different state space reduction methods in order to reduce the complexity of the Tetris state space. We presented our results in two ways, starting with a domain-specific analysis that focuses on details particular to the game of Tetris. Then we proposed a more general way of assessing the performance of a learning agent based on the coverage of the training set used.

To assess the results of learning, we had each agent play fifty Tetris games in order to evaluate their learning mechanism and state space reduction method. Analysis of the fifty games found that the agents learned to play in a very specific manner. The agents played strictly to minimize the number of holes on the board, ignoring line completion and board height. This style of play also meant that the agents performed well on board density. The agents were also able to recall the training set data with a fairly high rate of accuracy, between $77\% - 89\%$. Our experiments revealed that although the agents did not have access to a robust training data set, they were still able to develop a noticeable playing strategy. It is remarkable that the agents demonstrated some qualities of human play, mainly minimizing holes. As we saw in Figure 6, some of the agents were able to minimize holes as well as the human player.

Finally, we proposed a more general method for assessing the performance of a learner based on how well it acquired the knowledge in its training set, as a function of the degree to which that training set is representative of the entire landscape the agent is attempting to learn. We refer to this as "outlook". Computing outlook for the four agents we trained to play Tetris revealed that the agents using the *heights* state space reduction method demonstrated a more positive outlook than agents that used the *contours* state space reduction method. This result is contrary to comparison of the raw metrics, and provides an interesting and hopefully useful contrasting way of analyzing the performance of a learner.

Agent learning using logs of human activity is a viable means of capturing latent knowledge and may even lead to surprising behaviors. More importantly we hope that our work will encourage use of *clickstream* data in a similar capacity. It is clear from our results that knowledge transfer did in fact occur and that our agents developed a playing strategy (in large part because of the state space reduction methods) which captured a single component of the human's playing style. We would like to investigate other state space reduction methods and examine how they effect play. The lack of a state space reduction method that encompassed features that promoted line completion gave rise to agents that preferred to pack and stack the tetrominoes rather than clear lines and earn points. In future work, we also plan to tackle other domains and means of measuring training set quality.

References

1. Sklar, E., Icke, I.: Using simulation to evaluate data-driven agents. Multi-agent Based Simulation IX, Lecture Notes in Artificial Intelligence **5269** (2009)
2. Sklar, E., Blair, A.D., Pollack, J.B.: Chapter 8: Training Intelligent Agents Using Human Data Collected on the Internet. In: Agent Engineering. World Scientific, Singapore (2001) 201–226
3. Chernova, S., Veloso, M.: Confidence-based policy learning from demonstration using gaussian mixture models. In: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems. AAMAS '07, ACM (2007)
4. Hertz, J., Krogh, A., Palmer, R.G.: Introduction to the Theory of Neural Computation. Westview Press (1991)
5. Gupta, M., Chen, Y.: Theory and Use of the Em Algorithm. Foundations and Trends in Signal Processing. Now Publishers (2011)
6. Harter, D., Kozma, R.: Ontogenetic development of skills, strategies and goals for autonomously behaving systems. In: Proceedings of the 5th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI 2001). (2001) 178–181
7. Moore, A.W., Atkeson, C.G.: Prioritized sweeping: Reinforcement learning with less data and less time. Machine Learning **13** (October 1993) 103–130
8. Knox, W.B., Stone, P.: Interactively shaping agents via human reinforcement: The tamer framework. In: The Fifth International Conference on Knowledge Capture. (September 2009)
9. Ramon, J., Driessens, K.: On the numeric stability of gaussian processes regression for relational reinforcement learning. In: In ICML-2004 Workshop on Relational Reinforcement Learning. (2004) 10–14
10. Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-Dynamic Programming. Athena Scientific (1996)
11. Böhm, N., Kókai, G., Mandl, S.: Evolving a heuristic function for the game of tetris. In Abecker, A., Bickel, S., Brefeld, U., Drost, I., Henze, N., Herden, O., Minor, M., Scheffer, T., Stojanovic, L., Weibelzahl, S., eds.: LWA, Humbold-Universität Berlin (2004) 118–122
12. Szita, I., Lörincz, A.: Learning tetris using the noisy cross-entropy method. Neural Computation **18** (December 2006) 2936–2941
13. Chernova, S., Veloso, M.: Interactive policy learning through confidence-based autonomy. Journal of Artificial Intelligence Research **34**(1) (2009) 1–25

Advice Provision in Multiple Prospect Selection Problems

Amos Azaria[1] and Sarit Kraus[1,2]

[1] Department of Computer Science, Bar-Ilan University, Ramat Gan 52900, Israel
[2] Dept. of Industrial Engineering Jerusalem College of Technology, Israel

Abstract. When humans face a broad spectrum of topics, where each topic consists of several options, they usually make a decision on each topic separately. Usually, a person will perform better by making a global decision, however, taking all consequences into account is extremely difficult. We present a novel computational method for advice-generation in an environment where people need to decide among multiple selection problems. This method is based on the prospect theory and uses machine learning techniques. We graphically present this advice to the users and compare it with an advice which encourages the users to always select the option with a higher expected outcome. We show that our method outperforms the expected outcome approach in terms of user happiness and satisfaction.

1 Introduction

It is hard to overestimate the importance of decision-making in life. People make decisions on a daily basis; some are trivial such as whether or not to eat ice-cream, while others are important such as which apartment to buy. “Choice Bracketing”, termed by Read et al. [14], designates the grouping of individual choices together into sets. “Broadly Bracketing” indicates that the decision-maker takes all choices into account when making his decision, while “Narrow Bracketing” indicates that the decision-maker isolates each choice from all other choices. Of course, in order to bracket several decisions, they must all be converted to a single scale (such as money), where a decision that turns out to be a bad choice for one topic may be balanced out by another decision which turns out to be a good choice for a different topic.

It has been shown that people tend to use narrow bracketing and usually treat each decision as if in isolation from all other decisions, which in many cases results in a poor choice [20,14,15,3]. Gneezy and Potters [6] have shown that, in investment games, people who are forced into broadly bracketing by viewing their revenue less often become less risk averse and therefore increase their average revenue (this finding has been replicated by Moher and Koehler [11], however they have failed to replicate it in generalized settings). Unfortunately, these studies did not test users’ satisfaction, only their average total revenue.

A classic experiment that illustrates narrow bracketing was done by Tversky and Kahneman [20]. They asked their subjects the following question:
”Imagine that you face the following pair of concurrent decisions. First examine both decisions, then indicate the options you prefer:

Choice I. Choose between:
A. A guaranteed gain of $240.
B. A 25% chance to gain $1000 and a 75% chance to gain nothing.
Choice II. Choose between:
C. A guaranteed loss of $750.
D. A 75% chance to lose $1000 and a 25% chance to lose nothing."

Since people tend to be risk averse with a positive payoff and risk seeking with a negative payoff, a large majority of subjects (73%) chose both A and D. Only 3% of the subjects chose B and C. Combining A and D yields a 25% chance to gain $240 and a 75% chance to lose $760. However, combining B and C dominates this with a 25% chance to gain $250 and a 75% chance to lose $750. Indeed, when presenting all four combined choices (A+C, A+D, B+C and B+D), no one chose the dominated option (A+D). This experiment shows that although people tend to narrow their bracketing, they make better choices when explicitly broadening it.

We tackle people's tendency to narrow bracketing using an environment where people need to decide among several independent selection problems, whether they prefer a guaranteed outcome or a higher, but uncertain, outcome. We build an agent which first learns peoples' preferences and generates a general human model. The agent then searches the space of all possible combinations that can be chosen by the user and, based on the human model, advises the user which combination would be most beneficial for him.

Many real life situations resemble our problem. The most obvious example is when building an investment portfolio, where some stocks have a higher risk but also offer an opportunity to receive higher interest. On the other hand one can invest in a bond with a lower risk and a lower interest level. Most people combine different stocks and bonds, combining different levels of risk. A similar example might be a manager who wants to market several products, where each product has its own probability for success and failure and its estimated revenue. Our agent may be integrated into a Decision Support System which manages use on regular basis.

Another example might be a GPS guiding a driver who can choose between a longer road with a low probability of heavy traffic and a shorter road with a high probability for heavy traffic. If this decision is made several times on a trip the time consumption associated with choosing a road which turned out to be with heavy traffic may be canceled by future decisions in which the traffic was flowing. Therefore, considering the full route may result in greater satisfaction for the driver than making a separate decision for each and every junction.

For the last example suppose that a professor has several papers that she would like to submit to conferences. When considering each paper separately, the professor may send each paper to a conference to which it is likely to be accepted, however, when considering all papers together, the professor may prefer sending some papers to a highly refereed conference where it is likely that at least a few of them are accepted.

We would like to emphasize, that although we focus on advice provision, the agent may utterly replace the decision-maker. For instance, if the GPS mentioned above is embedded into an autonomous vehicle, it may take the human to his target without asking the human which route to take.

The main contributions of the paper are the following: first, we present the algorithmic challenge that is associated with the human bracketing problem. Second, we developed a multistage procedure for providing humans with a combination of choices in multiple prospect selections aiming at maximizing human satisfaction. The innovation in the preference elicitation of the multistage procedure stems from the use of an off-line learning group. New users do not need to provide any additional information and can receive advice with no learning period. Furthermore, the group elicitation procedure is done using simple problems in which humans can provide their real preferences and address the inability of people to recognize their preferences in complex decision problems. At last, we provide extensive experiments that show the success of our proposed method in terms of human satisfaction.

2 Related Work

Recommendation systems use models for predicting user ratings in order to supply advice. (See Ricci et al. [16] for a recent review). The main application of recommender systems is to find an item (or items) among hundreds or thousands of items which the system expects would be favorable to the user based on his previous preferences. A common case would be a recommendation system which recommends a book or a movie to a user, based on movies or books that the user enjoyed in the past [7].

In a recent work, Azaria et al. [2] propose a method for giving advice to users in an environment where users have several options from which to choose where the decisions are made in sequential order. However, in their work they assume that the users and the agent have different goals and utility functions, while in our work the agent's goal is solely to help the user and thus they both have identical utility functions.

Sarne et al. [17] attempt to facilitate people's decision-making process by modifying the presentation of the problem in an economic search environment. However, in their work they assume that people want to maximize their expected monetary value. We relax this assumption and assume that people have a non-monetary utility function which they try to maximize. In this paper we base this utility function on the Prospect Theory.

3 Prospect Theory

The Prospect Theory was presented by Kahneman and Tversky in [10] and later refined to the Cumulative Prospect Theory in [21]. The Prospect Theory is based on three principles. The first is that people do not take into account their total wealth when accepting or rejecting an uncertain opportunity (as suggested by the expected utility hypothesis [5]), but rather use their current wealth as a baseline and will be happy if they win an amount and become upset if they lose an amount. The second principle is loss aversion, where people hate losing more than they like winning. The third principle is that people have a subjective representation of probabilities and do not interpret probabilities fully rationally, but rather use their own decision weights when deciding whether to reject or accept a gamble. In his book, Kahneman [9] (p.314) gives the following examples: The decision weight that corresponds to a 90% chance is 71.2%, while the decision weight that corresponds to a 10% chance is 18.6%. According to these examples, people are likely to prefer a guaranteed outcome of \$80 than a gamble with a 90% chance of winning \$100, since the latter is only worth \$71.2 to them. Tversky and Kahneman elicited these weights by sequentially asking subjects to choose between a specific lottery and many different guaranteed outcomes. The equivalent to the given lottery for a certain subject was set to the average between the greatest rejected guaranteed outcome and the smallest accepted guaranteed outcome. However, these decision weights depend on peoples personality, their wealth, culture and the scope of the payoff in question. We build a human model by learning the decision weights which best suit our population and scope of payoff based on machine learning techniques.

4 Multiple Prospect Selections Advice Provision

The term *prospect* comes from Kahneman and Tversky, who refer to a lottery where a player has a chance to win (or lose) a certain amount as a prospect. We only use *simple prospects*, where a player can gain a certain outcome with some probability, and zero otherwise. A *prospect selection problem* $s =< x; p, y >$, is a problem where a player needs to choose between a guaranteed outcome of x and a probability of p to win an outcome y (we use the subscripts s_x, s_p and s_y respectively). We use $c \in \{g, u\}$ to denote the user's choice, where g denotes a choice of the guaranteed value (x), and u denotes the uncertain outcome (the prospect: y with probability p). An important point is that a fully rational player (i.e. one who maximizes expected monetary outcome) would always choose the uncertain outcome if $p \cdot y \geq x$, and the guaranteed outcome otherwise. However, as shown by Tversky and Kahneman [21], people do not act fully rationally not because they cannot do so, but because they use different decision weights.

In our work we consider *multiple prospect selection problems* where a player faces k (different) prospect selection problems. As mentioned in the introduction, people fail to broaden their bracketing and treat each selection problem as if it is isolated from the other selection problems. Therefore, people use their original decision weights, which are sub-optimal for multiple selection problems.

Our goal is to build an agent that advises a human player in which of the selection problems to choose the guaranteed outcome and in which to choose the prospect. We denote this advice by $a \in \{g, u\}$. We measure the performance of the agent in terms of human satisfaction from the final result, human satisfaction from the decisions he made, human satisfaction from the advice and the fraction of selection problems where the human followed the advice. We intentionally do not measure success in terms of the raw final stake, since, as mentioned above, people care about how much they have only in context, that is, in comparison to how much they could have had (or lost), and what risks they took or avoided in order to achieve that outcome.

We now present the prospect Selection problem Advice provider for Multiple Problems agent (SAMP). To build this agent we first use machine learning to elicit decision weights from given data. Based on these decision weights we build a general human model that can calculate the utility in human eyes for any combination of choices in a multiple prospect selection problem. Upon demand, SAMP searches for the best combination using this model and presents it to the user.

4.1 Decision Weights Elicitation

The first step in building SAMP, is eliciting human decision weights. Since these decision weights will be used to build a *general* human model, we collect data using subjects from a similar culture and using a similar scope of outcome and probabilities. However, we do not collect personal parameters regarding users' preferences (such as risk attitude). This data is collected from subjects who were asked to choose between a guaranteed outcome and a simple prospect. Note that this is a very simple choice and we therefore assume that humans can provide their real preferences.

Using the data we build a logistic regression classifier. We feed the classifier for each prospect selection problem with the probability of winning in the prospect, the ratio between the prospect outcome and the guaranteed outcome, and all quadratic combinations of the two. The classifier needs to classify the data based on the subjects' choices. Adding quadratic combinations is required since a linear combination of the two features isn't enough to learn a good model. Quadratic functions are used with success as SVM kernels [19]. More formally, the feature vector for the classifier is:

$$v = \{s_p,\ \frac{s_y}{s_x},\ {s_p}^2,\ (\frac{s_y}{s_x})^2,\ s_p \cdot \frac{s_y}{s_x}\}$$

and the classifier is trained on c. We use $\frac{s_y}{s_x}$ rather than s_y because we want the final decision weights learned to be independent of the outcome and solely depend on the probability.

Given a new prospect selection problem feature vector the classifier will output a number between 0 and 1 which, if greater than 0.5, indicates that the user is more likely to choose one option and if smaller than 0.5 indicates that the user is more likely to choose the other. However, since we are interested in learning the decision weights, we would like to know when people are *indifferent* between choosing the guaranteed and the uncertain outcome. We therefore are interested in the cases where the classifier will output exactly 0.5. Since we use a logistic regression classifier, we obtain:

$$\frac{1}{1 + e^{-(w^T \cdot v)}} = 0.5 \tag{1}$$

where w is the vector of weights obtained from the classifier and v is the feature vector described above. This implies: $1 + e^{-(w^T \cdot v)} = 2$, and therefore: $e^{-(w^T \cdot v)} = 1$, which implies: $(w^T \cdot v) = 0$. Since we are interested in finding the decision weights for a given probability, we denote: $z = \frac{s_y}{s_x}$ and solve Equation 4.1 for z. Writing both the feature vector and the weight vector explicitly we obtain:

$$w_0 + w_1 \cdot s_p + w_2 \cdot z + w_3 \cdot s_p + w_4 \cdot z^2 + w_5 \cdot s_p \cdot z = 0 \tag{2}$$

Solving Equation 2 we obtain:

$$z(s_p) = \frac{-w_5 - w_2 \pm \sqrt{(w_5 s_p + w_2)^2 - 4w_4(w_3 {s_p}^2 + w_1 s_p + w_0)}}{2w_4} \tag{3}$$

Given a probability s_p, the actual decision weight $d(s_p)$ is $\frac{1}{z(s_p)}$. Only a single solution is appropriate; in most cases the second solution is either negative or greater than 1. We set $d(1)$ to 1 and $d(0)$ to 0. If in any other case $d(s_p) > 1$ or $d(s_p) < 0$, we set it to 1 or 0 respectively (this should not happen if the classifier was trained on sufficient data).

4.2 Assessing the value of multiple prospects

Using the decision weights obtained above, we can assess a value of a prospect. Given a prospect with a probability s_p of winning an outcome s_y, using the decision weight $d(s_p)$ we obtain that the value of the prospect for the user is simply:

$$u(s_p, s_y) = d(s_p) \cdot s_y \tag{4}$$

The main challenge which remains is to assess the value of multiple (k) prospects. If we were simply to sum the value of all prospects individually, we would fail by using the exact concept which we are trying to overcome, i.e. narrow bracketing. The first step in assessing the value of multiple prospects is calculating the final probability for each of the possible outcomes. Note that there may be up to $n = 2^k$ possible outcomes. These outcomes must be sorted from low to high. We use the following algorithm to efficiently (linear in output) obtain all possible outcomes (along with their probabilities):

Input: Pml - A list of selection problems ($s =< x; p, y >$) and user's selections on these problems ($c \in \{g, u\}$).

Output: OP - A hushmap holding all possible outcomes as keys and their probabilities as values.

```
1:  $OP[0] \leftarrow 1$
2:  **for** each problem $s$ in $Pml$ **do**
3:      **if** guaranteed scenario was selected ($c = g$) **then**
4:          **for** each $outcome$ in $OP$ **do**
5:              increase $outcome$ by guaranteed value $(s_x)$
6:      **else**
7:          $Temp \leftarrow OP$
8:          clear $OP$
9:          **for** each $outcome$ in $OP$ **do**
10:             $OP[outcome] \leftarrow Temp[outcome] \cdot (1 - s_p)$ [1]
11:             $OP[outcome + s_y] \leftarrow Temp[outcome] \cdot s_p$
12: return $OP$
```

Once we obtain the probabilities $\boldsymbol{p} = p^1, p^2 \ldots p^n$ ($\sum_{i=1}^{n} p^i = 1$) and outcomes $\boldsymbol{y} = y^1, y^2 \ldots y^n$ ($y^1 < \ldots < y^n$), based upon the Cumulative Prospect Theory [21], we assess the value for the user by:

$$u(\boldsymbol{p}, \boldsymbol{y}) = \sum_{i=1}^{n} d\big(\sum_{j \geq i} p^j\big) y^i - d\big(\sum_{j > i} p^j\big) y^i \tag{5}$$

We now explain the intuition behind this method. Note that when $i = 1$, $\sum_{j \geq 1} p^j = 1$ and since $d(1) = 1$, then $d\big(\sum_{j \geq 1} p^j\big) y^1$ is simply y^1. This illustrates the fact that the user is guaranteed an outcome of at least y^1 (since y^1 is the lowest possible outcome). The second position's contribution ($i = 2$) illustrates the fact that the user has a probability of $1 - p^1$ to obtain at least y^2. The second term in the sum omits

[1] In 10 and 11, if $OP[outcome]$ or $OP[outcome + s_y]$ already have a value, increment that value by $Temp[outcome] \cdot (1 - s_p)$ or $Temp[outcome] \cdot s_p$ accordingly.

multiple counting of the same outcome. We now show that Equation 5 generalizes a single prospect. Given a single prospect, there are two possible outcomes $\boldsymbol{p} = 0, s_y$, and the probabilities are $\boldsymbol{y} = 1 - s_p, s_p$. Assigning p and y in Equation 5 we obtain: $u(\boldsymbol{p}, \boldsymbol{y}) = d(1) \cdot 0 - d(s_p) \cdot 0 + d(s_p) \cdot s_y - 0 \cdot s_y$ which is identical to Equation 4.

In order to be more efficient, in practice we do not sum the probabilities in every iteration but start with 1 and subtract the current probability in every iteration.

4.3 Advice Provision

Given a Multiple Prospect Selection Problem, each combination of choices yields different vectors $\boldsymbol{p}$ and $\boldsymbol{y}$ and therefore using the model above, each combination of choices yields a different user value ($u(\boldsymbol{p}, \boldsymbol{y})$). SAMP searches for a combination of choices which maximizes this value. This search can be invoked using any search method (such as genetic algorithm or interior-point).

4.4 Fully Rational Agent

Since human full rationality assumption is broadly used and is a very common assumption, we compare the performance of SAMP to the performance of a *rational agent* which assumes human full rationality (i.e. assumes that people want to maximize their expected monetary value). Performance is measured in terms of human satisfaction. In all selection problems the rational agent advises the user to choose the uncertain outcome if $s_p \cdot s_y \geq s_x$, and the guaranteed outcome otherwise. The rational agent's advice does not depend on the number of selection problems k. A user following the rational agent's advice, will maximize his expected outcome.

5 Experimental Design

We ran our experiments using Amazon's Mechanical Turk (AMT) [1]. AMT has become an important tool for running experiments and was established as a viable method for data collection [12].

We recruited 52 participants for SAMP's learning phase and 202 participants for evaluating both SAMP and the fully rational agent. Two subjects were removed from the evaluation process due to invoking a response in less than 15 seconds, which was considered unreasonably fast, and we suspected that they just wanted to move on to their next task. All other subjects took at least 25 seconds to submit their preferences, with an average of 94.7 seconds.

In the evaluation phase, each subject received a single advice, either from SAMP or from the fully rational agent. In the learning phase the subjects did not receive any advice. Each subject participated only once. 52% of the subjects were males and 48% were females. Subjects' ages ranged from 18 to 75, with a mean of 32.2 and median of 29. All subjects were residents of the USA.

We set $k = 5$, i.e. the subjects had to make their choice regarding five prospect selection problems. The guaranteed outcome was drawn uniformly between 2 and 10 cents, and the probability to win the uncertain outcome (s_p) was drawn uniformly from

$\{0.05, 0.1, 0.15, 0.2, ..., 0.95\}$. Choosing the uncertain outcome yield upto 35% more on the expected utility value of the guaranteed outcome.

Once the subjects selected their preferences, we invoked a lottery on each of the prospect selection problems and paid them according to their preferences. I.e. if a subject chose the guaranteed outcome for a certain selection problem he was paid accordingly, and if he chose the uncertain outcome, the system randomly generated a number r in $[0, 1]$ and paid the subject s_y if $r \leq s_p$. In order to encourage narrow bracketing in the learning phase, the subjects were told to answer each selection problem as if it were standalone, and instead of having the system invoke all the selection problems, the system randomly selected a single selection problem and invoked only that one. The subjects were fully aware of this process. In the evaluation phase, however, the system obviously invoked all five prospect selection problems.

In the evaluation phase, in addition to the five selection problems, the subjects were also presented with the agent's advice. Prior to receiving the actual advice, the subjects were told that the advice is provided by a third party agent which is trying to help them. The subjects were shown the consequences of following the advice using a pie chart, which indicated the actual probability of winning each possible outcome. A pie chart is a common method of showing how a full resource is split up. Therefore, in our experiments, we use the pie chart to show how the certainty is split to smaller probabilities. This method enables easy comprehension and provides all information in a single chart.

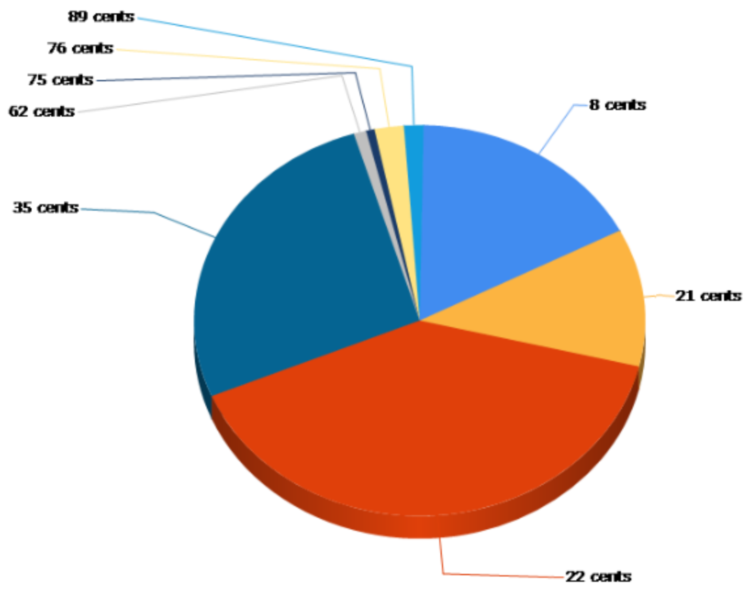


Fig. 1. A visualization of the agent's advice

Figure 1 presents an example for a visualization of the agent's advice. In this example, if the user follows the exact advice given by the agent, he is guaranteed a win of at least 8 cents, however, he might win upto 89 cents. The interpretation of the pie chart was explained to the subjects, and their comments indicated that they clearly understood this interpretation. After submitting their choices and told the final results, the subjects were asked the following three questions:

1. Are you happy with the final result?
2. Are you happy with the decisions you made?
3. How good was the advice given to you by the system?

The subjects answered these questions on a 1 to 5 scale, where 1 is associated with ”not at all” in the first two questions, and ”very bad advice” in the third question, 3 is associated with ”o.k.”, and 5 is associated with ”very happy” in the first two questions, and ”very good advice” in the third question. All subjects were also asked to give comments if they had any.

5.1 SAMP construction

Recall that in SAMP construction, after the data is collected a logistic regression classifier is built using this data. This classifier is built in order to find the decision weights, therefore the classifier is trained on all the data. However, in order to evaluate the performance of the classifier we also ran a 10 fold cross validation on the data. We found that the accuracy of the classifier was 67.2% (F-Measure 0.67). These results are satisfying, as we are considering a general human model and people differ in their risk attitudes. Nonetheless, this classifier allows us to build a good enough model for people's decision weights. Using other classifiers yield similar results (e.g. SVM yields accuracy of 67% and neural networks yield 65.6%), however, we use logistic regression so we can extract the model as described in section 4.1.

After extracting the parameters we were able to calculate the decision weights for any given probability. Several decision weights learned by SAMP for some selected probabilities can be found in Table 1. For example the decision weight associated with 0.25 is 0.20. This indicates that people should be indifferent between a prospect that offers a 25% chance to win 10 cents and a guaranteed outcome of 2 cents. Note that the decision weights are lower than the actual probabilities in all cases; this indicates that, in our environment, people were risk averse regardless of the probability.

Probability	Decision Weight	Probability	Decision Weight
0.10	0.07	0.60	0.48
0.25	0.20	0.75	0.64
0.40	0.32	0.90	0.85

Table 1. Decision weights for selected probabilities

Using these decision weights SAMP builds a general human model as described in section 4.2. Since we used a small k, we had a small search space and therefore

SAMP used an exhaustive search when searching for the best combination to be advised to the users. Figure 1 presents an example of a SAMP's advice, in a scenario where the guaranteed outcome among all selection problems sums up to 21 cents. Following the advice seems preferable to choosing the guaranteed outcome, as the probability of gaining less than the guaranteed outcome by following the advice is less than 14% and the user is very likely to gain more than the guaranteed outcome, with a slight chance of gaining much more. The expected utility from following SAMP's advice is 25.7 cents, however, SAMP evaluated the human value of its advice in 23.4 cents (greater than any other combination of choices).

5.2 Results

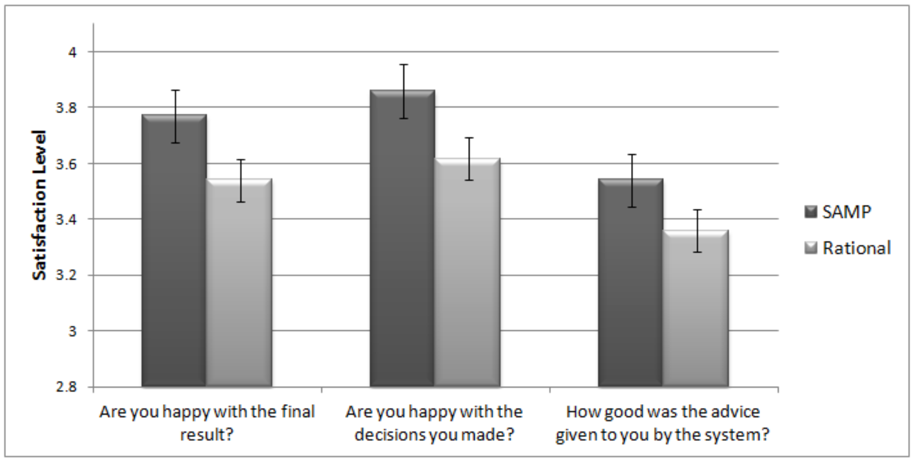


Fig. 2. Comparison between user satisfaction levels among subjects who received SAMP's advice and those who received the fully rational agent's advice

Figure 2 presents the final user satisfaction levels for users who received SAMP's advice and users who received the fully rational advice. The higher the satisfaction level the better the result. As can be seen in the figure, in all three questions SAMP outperformed the rational agent. ANOVA test on all three parameters (together) indicates that these results differ significantly $(p < 0.05)$. In the first two questions results differ significantly $(p < 0.05)$ also using the student's t-test between the two groups, however, the results obtained from the third question did not reach significance level by the student's t-test (although they were very close to achieving it, with $p = 0.07$).

We also compare the average fraction of advice followers, i.e. the fraction of prospect selection problems where each user followed the advice for that selection problem. The advice given by SAMP for each of the selection problems was followed 76.5% of the time, while the advice given by the rational agent was followed only 70.8% of the time. Since more users followed the advice given by SAMP, on average, users who received

SAMP's advice resulted with a non-statistically significant higher outcome (34.5) than those who received the rational advice (32.2). However, we attach minor importance to this achievement since SAMP is not destined to increase the users' expected outcome but to increase their satisfaction.

To sum up the results, the subjects' satisfaction level in all parameters was greater when they received SAMP's advice than when they received the fully rational agent's advice. This indicates that the assumptions used by SAMP are more accurate than assuming full rationality. However, the rational agent also performed well, since its advice was followed in a large majority of selection problems. The rational agent also reached an average score of 3.36 when asked how good it was (recall that 3 is associated with "o.k."). Therefore, the rational agent may be good enough for a cold start (when no training data is available).

Many subjects appreciated the pie chart display as we received comments that praised it. One example is: "The pie chart made it much easier for me to make my decision."

6 Discussion and Future Work

Research has shown that people act differently when the stakes are higher, and tend to be more risk-averse [8]. Nonetheless, many users commented that they would act differently if the stakes were higher. Since people tend to reject favorable prospects when the stakes are higher, narrow bracketing will cause them to reject many more favorable prospects than they should. We therefore expect that, when the stakes are higher, our agent will have a larger impact on the user satisfaction. However, in our settings raising the stakes would not allow us to actually pay the subjects and the prospect selection problems would have needed to be hypothetical (for a comparison between hypothetical and real scenarios see [13]). Another option would be to retain the current stakes but conduct the research in a different culture where the average monthly expenditure of participants is much lower. Using a different culture instead of actually raising the stakes is a common method used in psychological experiments [8,4].

In many occasions the topics involved may be dependent, where a certain result in one topic may increase or decrease the probability for a different result in a second topic. For example, if there is heavy traffic on one road, this may be a sign that there will also be heavy traffic on a second road. In such cases SAMP may easily be generalized using Bayesian inference. However, since humans encounter great difficulty in calculating Bayes' rule themselves (unless properly trained [18]), therefore, SAMP may be even more of an assist in such cases.

It remains questionable, what will happen as the number of prospect selection problems increases (k). In such a case, the pie chart may become very complex as the number of outcome options increases exponentially. One option is to group up similar outcomes, but still it is unclear how to do so.

In current work, the experiments were based on a single trial. We intend to test a scenario where the user doesn't see all of the selection problems at once, but must make his selections sequentially. This scenario is quite common in real life where one usually faces sequential decisions. In such a scenario, the user (and therefore the agent) has only partial data on future selection problems, making it harder to advise the user as to which option to choose for each selection problem. The user on the other hand, might come to appreciate the value of the advice better over time and increase his trust in it. The agent may also gain trust by informing the user what he may have won had he followed the advice.

In future work we also intend to build a personalized agent that will provide different advice depending on the user with whom it is interacting. We will either need to find a good method to cluster the training data into different clusters and have SAMP provide advice depending on the user's cluster, or completely personalize the agent using parameters that could be different for every user. When interacting with a user, the agent will first collect data on the specific user and then provide advice which is best suited for him. One way to collect data is by asking the user to first make his decisions as if each selection problem is a standalone and then the agent will advise the combination which is best for that specific user. We expect this method to significantly improve the agent's performance in terms of user satisfaction. However, this method would require a longer procedure for data collection, and the users participating in the learning phase would be required to answer an additional questionnaire. A personalized agent will also require learning data from more subjects.

7 Conclusions

When people need to make decisions regarding several topics they tend to view each topic as if it were on its own and when deciding which action to take they ignore all other topics. This tendency was shown to be harmful, and people would perform better if they could make a more intelligent selection based on all topics together. Helping people in such cases isn't simple since people do not necessarily want to maximize their expected monetary value.

In this paper we present an agent which advises people in an environment which includes multiple prospect selection problems, where in each selection problem the user must choose between a guaranteed outcome and an uncertain outcome. This agent collects data on humans in the desirable environment and, based on the data it builds a general human model using prospect theory concepts. Using this model, the agent searches for a combination of selections which is most favorable for humans and recommends it to the user. We present the resulted combination using a pie chart which visualizes the probability of each possible outcome. The advice composed by our agent significantly outperforms fully rational advice (i.e. advice which maximizes expected utility) in terms of user satisfaction.

Reaching such an achievement using a general human model is conspicuous since people differ from each other, and many times advice which might be good for one user may not be as good for another. In this work we have proven the concept of using decision weights, showed how to learn them from data and generalize their use to a combination of multiple prospect selection problems.

8 Acknowledgment

This work was supported in part by ERC grant $\#267523$, the Google Inter-university center for Electronic Markets and Auctions, MURI grant number $W911NF-08-1-0144$ and ARO grants $W911NF0910206$ and $W911NF1110344$.

References

1. Amazon. Mechanical Turk services. http://www.mturk.com/, 2012.
2. A. Azaria, Z. Rabinovich, S. Kraus, C. V. Goldman, and Y. Gal. Strategic advice provision in repeated human-agent interactions. In *AAAI*, 2012.
3. Nicholas Barberis, Ming Huang, and Richard H. Thaler. Individual preferences, monetary gambles, and stock market participation: A case for narrow framing. *American Economic Review*, 96(4):1069–1090, 2006.
4. L. A. Cameron. Raising the stakes in the ultimatum game: Experimental evidence from Indonesia. *Economic Inquiry*, 37(1):47–59, 1999.
5. M. Friedman and L. J. Savage. The expected-utility hypothesis and the measurability of utility. *The Journal of Political Economy*, 60(6):463–490, 1952.
6. Uri Gneezy and Jan Potters. An experiment on risk taking and evaluation periods. *Quarterly Journal of Economics*, pages 631–645, 1997.
7. Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, January 2004.
8. Steven J Kachelmeier and Mohamed Shehata. Examining risk preferences under high monetary incentives: Experimental evidence from the people's republic of china. *American Economic Review*, 82(5):1120–41, December 1992.
9. Daniel Kahneman. *Thinking, fast and slow*. Allen Lane, London, 2011.
10. Daniel Kahneman and Amos Tversky. Prospect theory: An analysis of decision under risk. *Econometrica*, 47(2):263–291, 1979.
11. Ester Moher and Derek J. Koehler. Bracketing effects on risk tolerance: Generalizability and underlying mechanisms. *Judgment and Decision Making*, 5(5):339–346, August 2010.
12. G. Paolacci, J. Chandler, and P. G. Ipeirotis. Running experiments on Amazon Mechanical Turk. *Judgment and Decision Making*, 5(5), 2010.
13. Daniel Read. Monetary incentives, what are they good for? *Journal of Economic Methodology*, 12(2):265–276, 2005.
14. Daniel Read, George Loewenstein, and Matthew Rabin. Choice bracketing. *Journal of Risk and Uncertainty*, 19(1):171–197, December 1999.
15. Donald A. Redelmeier and Amos Tversky. On the framing of multiple prospects. *Psychological Science*, 3(3):191–193, May 1992.
16. F. Ricci, L. Rokach, B. Shapira, and P.B. Kantor, editors. *Recommender Systems Handbook*. Springer, 2011.
17. David Sarne, Avshalom Elmalech, Barbara J. Grosz, and Moti Geva. Less is more: restructuring decisions to improve agent search. In *Proc. of AAMAS*, pages 431–438, 2011.
18. J. Shanteau, M. Grier, J. Johnson, and E. Berner. Teaching decision-making skills to student nurses. *Teaching decision making to adolescents*, pages 185–206, 1991.
19. Thorsten Thies and Frank Weber. Optimal reduced-set vectors for support vector machines with a quadratic kernel. *Neural Comput.*, 16(9):1769–1777, September 2004.
20. A. Tversky and D. Kahneman. The framing of decisions and the psychology of choice. *Science*, 211(4481):453–458, January 1981.
21. Amos Tversky and Daniel Kahneman. Advances in prospect theory: Cumulative representation of uncertainty. *Journal of Risk and Uncertainty*, 5(4):297–323, October 1992.

Player Adaptive Agents in Location-Based Mobile Games

Adib Parkar, Spencer Frazier, Chao Huang, Yu-Han Chang and Rajiv Maheswaran

University of Southern California
4676 Admiralty Way
Marina Del Rey, CA 90292

Introduction

With GPS enabled mobile phones becoming ubiquitous, there is a tremendous interest being generated in location-based games for such platforms. The convenience of being able to play the game at any place, coupled with the raw processing power now available in handheld devices, provides an opportunity for relatively complex games that are not limited to a traditional gaming console. Our ultimate goal is to design a mobile game agent that adapts to a player's play style and generates targets based on the player's location that attempts to increase player movement. As a first step towards reaching this goal, we present a game for iOS devices that we used to attempt to model player behavior by running a two week experiment.

Related Work

We are attempting to look at how location-based games can be incorporated with intelligent agents that can model player behavior and can learn patterns using these models so that it may manipulate the game world to try and modify how the game is being played. This 'modification' of the play style may range from getting the player to move more or less, or to increase or decrease player engagement, or to entice a player towards a particular goal or target. Such a game would not only have to be well designed but also robust in data collection and computation.

Applications of such mobile game agents are now becoming quite popular. Researchers are exploring the use of mobile games to promote healthy living [1], through active playing [6] and passive encouragement [4]. In our game, we currently use experimental data gathered using a control group to come up with a game model to get players to increase movement and play activity. Using this new model, we hope to create an agent within the game itself that can learn a personalized model for the player currently playing the game so as to optimize that player's activity. In our specific case, we are looking to maximize player movement while engaged in the game, and our agent design is focused on this aspect. There is no doubt this is a hard problem. How can we design an agent that uses the game player's own habits to manipulate in-game entities so as to not only increase player movement, but also ensure the player remains engaged in playing and enjoying the game. Personalization in games has been studied using procedural content generation [5], genetic algorithms [6], Bayesian knowledge tracing in educational games [3], and effective computation that models emotions [2].

Platform and Technology

The games map-based interface was coded using Objective-C, XCode, Google Maps, and the iOS MapKit framework. The battle interface was coded using a combination of iOS UIKit frameworks and the Cocos2D iOS animation framework. The networking protocol for reporting and recording user actions, user locations and user states based on game content utilizes the Objective-C ASIHTTP request framework to deliver asynchronous updates to a MySQL database containing disparate values for all players separated by userID and deviceID. Graphics for the game were drawn by the researchers.

Game Description

The game ZOMBIESC is a location-based game in which players attempt to eradicate zombies that spawn around them by engaging in battles with the zombies. Zombies that spawn around the player may be of various types and difficulty, with 'higher level' (i.e. increasingly difficult) zombies being harder to kill but providing larger rewards. Zombies spawn (appear) on a world map around the player's current location, and a player must move towards a zombie group in order to fight it. When in battle mode, a player attempts to defeat all the zombies in that group. If the player feels that the battle is not going well, he or she has the option to flee without gaining any points. Thus a successful strategy involves not only engaging in as many battles as possible, but also in wisely choosing which zombie groups to fight. Fighting zombies provided players with 'cash' and 'tickets'. Cash could be used to purchase better ammunition, health, and armor upgrades, while the tickets were a means of keeping score.

Over the course of the fifteen days of the experiment, we tried different clustering techniques for the zombie spawn locations in an attempt to see if we could get players to move more while playing the game. Using this data we hoped to design the game itself as an intelligent agent that attempts to learn a player's behavior while playing the game and modifies the spawn location of various zombie groups to try and entice the player to change his or her location for larger rewards. Our data is based on the play patterns of 39 people who took part in the experiment.

The Game Agent

The core data component which is used to derive a representation for a state of the game is the player's GPS data. The latitude and longitude, in addition to the player's heading and the GPS accuracy, are used together to model a play style for that player. We define the following play styles - actively playing, passively playing, and not playing. Active play is achieved when a player has multiple location changes while also engaging in battles with groups of zombies. A location change is defined such that the player has had three significant location updates within a span of thirty seconds. A location update is defined in such a manner to account for unwanted GPS noise that may occur in areas of poor GPS reception. If a player does not qualify as having made three significant location updates, but is still actively entering battle mode, we consider the player to be

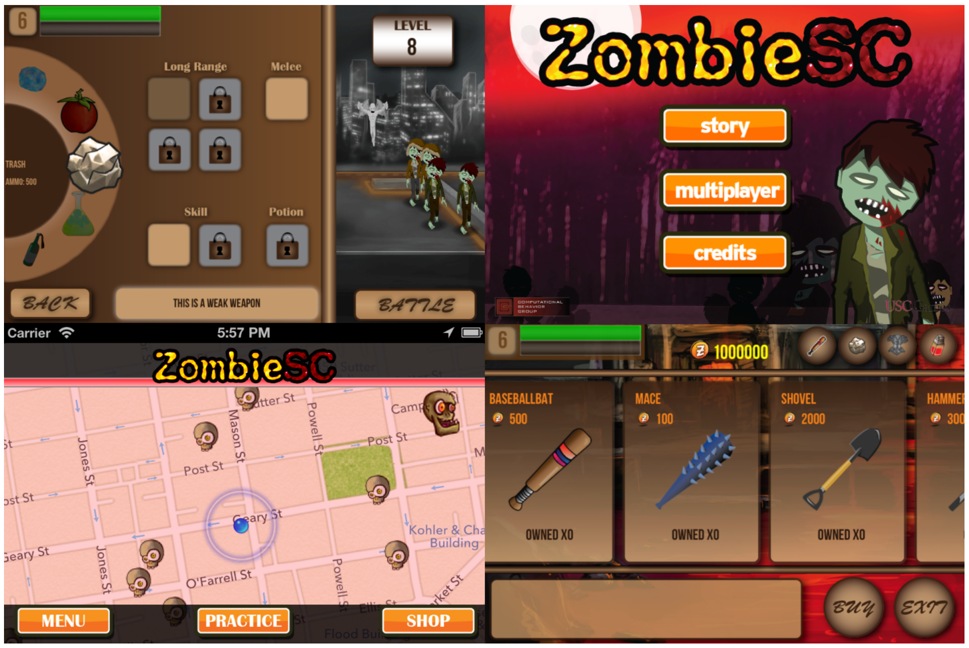


Fig. 1. Screenshots from the game.

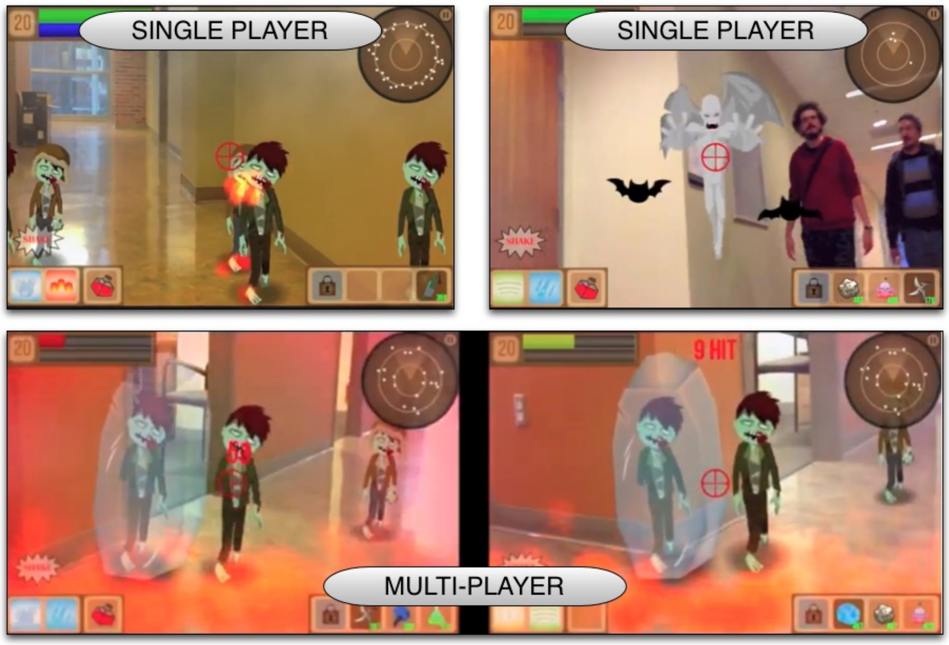


Fig. 2. Battling zombies.

’passively playing’, or playing but not moving. The game agent tries to learn a technique to get players out of the passively playing group and into the actively playing group. Players that do not interact with the game world, or do not play the game at all are considered to be not playing. It is possible for players to be moving and not playing the game, that is the players provide location updates but do not engage in battles, but this is also considered as ’not playing’ the game.

When a player first starts playing the game, zombies are spawned randomly around the player in a 500m radius. The game logs all play data, including the location of the spawned zombies, the action performed by the player (location update - i.e. movement, small battle, medium battle, or large battle), the heading of the player, the GPS accuracy, and the player’s statistics (score and wins). For the first week of the experiment (September , zombies were always spawned randomly around the player’s location, however for the second week, the zombie spawn locations were clustered around the players differently in the hope that it would get the players to move more to gain a higher score. In both cases, zombies were spawned within the 500m radius. Our aim was to see if the new positioning algorithm for the zombies increased player movement and activity.

Positioning of the zombies was thus completely under the game agent’s control. In the current design, the game agent did not have any knowledge about any real world points of interest or importance other than the player’s latitude and longitude. Thus, it is possible for the game agent to cause zombies to spawn in locations that may not have been physically accessible to the player. However, since zombies would appear in multiple locations, the player would always have some options to proceed towards.

Experiment and Results

To get people to play the game, we held a raffle with the winners being the players with the highest scores at the end of both weeks of the experiment. We had 39 people play the game. In the first week, the game spawned zombies completely at random every 30 seconds around the player’s current location. For the second week, the zombie spawning algorithm was modified to cluster zombies further away from a player while actively playing the game, perhaps forcing the player to change his or heading to encounter more zombies. The zombie clusters were mainly tight clusters with a radius of about 200m, and there were three clusters within the 500m radius around the player. The zombies spawn locations were affected by the player’s heading and the type of zombie the player normally targeted. Variables to affect the ratio of small, medium and large zombies spawned were available for manipulation. However, even in this case, zombies were re-spawned every 30 seconds.

We obtained a good amount of player data across the two weeks of the experiment from the 39 participants. The game tracked player locations as well as player actions, which helped divide players into the three categories mentioned before - actively playing, passively playing, and not playing. One of our goals was to increase player activity in the second week, something that we achieved to a certain extent as can be seen in Figure 3. Clearly, on average, players were playing more each day in week two when compared to week one on a day to day basis. Using the data, we were also able to ’re-

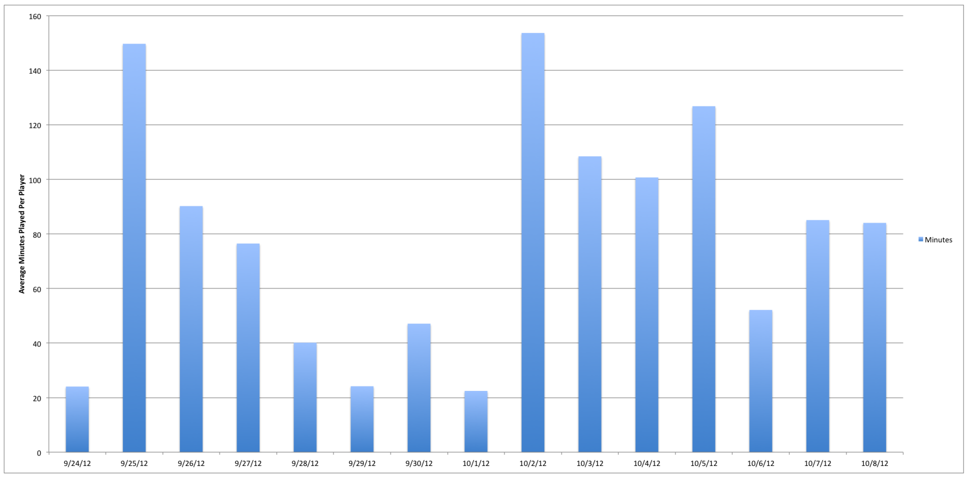


Fig. 3. Average number of minutes played per day across the two weeks.

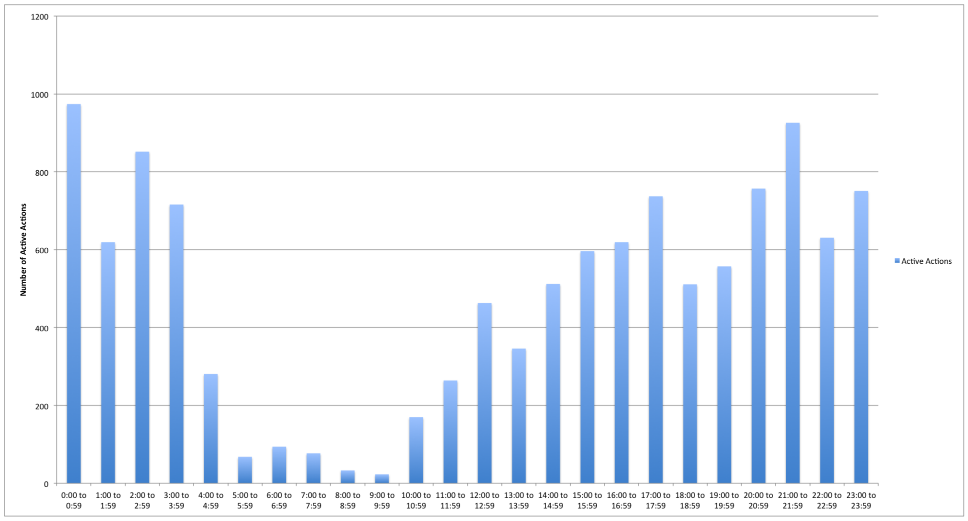


Fig. 4. Play activity distribution over different hours of the day.

play' the game by mapping player and zombie locations on a map. This visualization of the data was required to see if we managed to achieve our second goal of week two, which was to make the players move more while actively playing. However, we were unable to entice players to move any more in week two as compared to week one. A reason for this appears to be the fact that zombies would re-spawn at different locations every thirty seconds, which allowed players to simply wait until zombies spawned closer to them before engaging in battles for points. This was the preferred strategy for most players, rather than actually moving towards zombies when they spawned further away.

Further Work

Since our attempted adaptive algorithm was unsuccessful in increasing player movement, one of the first tweaks we plan to implement is to remove the option of being able to wait for close zombie re-spawns due to the thirty second refresh. This is just a step towards figuring out how to optimally cluster zombies, since the algorithm itself is not completely adaptable to a specific player. Our final goal is to create an in-game agent that learns using the player's habits on how to optimally cluster the targets so as to maximize some form of player activity.

Acknowledgements

This material is based upon work supported by the AFOSR Award No. FA9550-10-1-0569 and ARO MURI award No. W911NF-11-1-0332.

References

1. F. Buttussi and L. Chittaro. Smarter phones for healthier lifestyles: An adaptive fitness game. In *Pervasive Computing*, 2010.
2. E. Hudlicka. Affective computing for game design. In *4th Intl. North American Conference on Intelligent Games and Simulation*, 2011.
3. A. Molnar. Educamovil: Mobile educational games made easy. In *World Conference on Educational Multimedia, Hypermedia and Telecommunications*, 2011.
4. J. Pollak, G. Gay, S. Byrne, E. Wagner, D. Retelny, and L. Humphreys. Time to eat - using mobile games to promote healthy eating. *IEEE Pervasive Computing*, 9:21–27, 2010.
5. N. Shaker. Towards automatic personalized content generation for platform games. In *Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2010.
6. M. Verma. Personalised gaming: a motivation and overview of literature. In *Proceedings of The 8th Australasian Conference on Interactive Entertainment: Playing the System*, 2005.