The Effectiveness of Peer-Designed Agents in Agent-Based Simulations*

November 10, 2012

Abstract

The ability to reliably represent and replicate choices people make is crucial for building accurate models of day-to-day situations. The fact that people are inherently rationally- and computationally-bounded increases the difficulties in designing such simulations. This paper builds on the use of peer-designed agents (PDAs) – computer agents developed by people – to show their effectiveness in generating a variety of strategies and behaviors and in alleviating the simulation and behavior analysis of systems populated by human individuals with diverse strategies. The paper synthesizes the PDA-based simulation components and ideas that appear in recent PDAs literature into a cohesive simulation design, and reports a set of experiments aiming at validating the ability of PDA-based simulations to exhibit realistic behavior both in the individual agent and the system levels. The validation of individuals' ability to reliably capture their strategies into PDAs is quantitative, relying on four games from different domains. The domains vary in aspects such as game complexity and the environment dynamics. The applicability of the PDA-based approach in the system level is evaluated using a large scale experiment involving 34 PDAs, each designed by a different person. All in all, the set of strategies obtained by means of PDAs is substantially richer and more varied in comparison to the limited sets of strategies used in prior multi-agent simulation studies.

*Preliminary results of this research were published in the proceedings of AAMAS 2008.

1 Introduction

In recent years, agent technology has become the predominant approach for developing simulation systems [32, 34, 49]. The use of agents in simulations encompasses many advantages. First, it allows utilization of domain knowledge to achieve one's goals. Second, it allows simulation of high-level types of information-oriented behaviors, interaction with other individuals and persuasion in order to make separate decisions. In particular, Multi-Agent System (MAS) based simulations enable simulations of large-scale systems, due to the relatively low cost of cloning agents [46, 49]. This allows researchers to test their theories about social interactions, which might otherwise be unverifiable, in computational test beds.

While these agent capabilities substantially improve the modeling of the individuals they represent [19, 49], it is commonly believed that in order to achieve a good understanding of human behavior in large scale simulations, people must participate in the simulations. This is especially true due to the diverse behavior of people, which makes it difficult to capture behavior patterns in a monolithic model [2]. Moreover, as in many real-life situations, the agents simulating people in a given environment have only partial knowledge of the environment [37], making it more difficult to predict strategic behavior, which also varies greatly between different people. Problems also arise in scenarios where the agents are expected to exhibit human behavior, while the strategies embedded in them were derived from real data, collected under dissimilar settings.

Most agents-based simulations rely on agents designed by experts or the simulation designer herself. Naturally, the number of different agent behaviors embodied in the different agents when designed by a single person is small. The fact that the agents are designed by a person who is a "regular" user implies that the behaviors incorporated in the different agents represent that person's way of thinking or her beliefs regarding people's behaviors in this domain, rather than adequately representing the population in general. Furthermore, even if the different agents programmed do adequately represent specific individuals in the simulated population, the small set of behaviors does not necessarily capture the diversity and richness of behaviors exhibited by the different individuals of the simulated population.

This paper aims to solve the problem of generating a rich set of strategies that reliably capture the diversity in people's behavior in simulations by means of PDAs. Recent research has increasingly relied on peer-designed agents (PDAs) – computer agents developed by human subjects – as a representative of people (mostly as a means for improving the believability of agents in simulations) and

introduced the notion of PDA-based methods. PDAs have been used in various settings, from negotiation [26] to security domains [25]. Nevertheless, despite the wide use of PDAs in simulations, to the best of our knowledge an in-depth analysis of the effectiveness of PDAs in capturing the behaviors of the people they represent has not been carried out to date. Many of the works using PDAs simply assume the existence of such similarity, and even those that compare the performance of PDAs and people, base the comparison on the average (or collective) performance of the two groups [26]. Similarly, none of these works addressed or attempted to evaluate the question of how rich and varied are the strategies obtained through the use of PDAs. Furthermore, most PDA-related research focuses on the implementation results rather than on the methodological aspects of the approach.

This paper stems from the research on PDAs and deals with the question of how useful they are in simulating systems in which people are the key players, and what means are effective for incorporating PDA technology in such systems. In particular, it focuses on the person-PDA similarity and the diversity of strategies constructed. We present results from both single-player and multi-player games. The latter is, in many cases, impossible to solve using theoretical tools (e.g., finding an optimal strategy when dealing with cooperative environments, or an equilibrium when dealing with self-interested environments [6, 20]).

The paper makes an important leap forward in advancing PDA research by introducing a cohesive PDA-based method for the development and use of PDAs in simulation and in demonstrating that PDAs reliably capture the strategic behavior of their designers. The method summarizes and synthesizes common practices used in recent PDA literature into a comprehensive and unified approach that covers all aspects of the process. The concise and detailed description, as well as substantial experimental evidence of the ability of a PDA-based method to reliably capture people's strategies in PDAs, are, to the best of our knowledge, the first attempt to set the empirical basis for this notion.

This paper is organized as follows. The following section reviews relevant literature, mainly emphasizing the increased interest in agent-based simulations and PDAs in particular. It also presents the alternative approaches and the inconclusiveness concerning the ability of PDAs to reliably represent people in simulations. Section 3 details a PDA-based simulation design, focusing on various methodological aspects. The experimentation to evaluate the ability of capturing people's strategies using PDAs technology is described in detail in Section 4. Section 5 describes actual implementation of a large-scale simulation system in the parking domain, and discusses various aspects of that effort. An extensive

analysis of the ability of the PDA-based approach to produce a rich set of varied representative strategies and a comparison to prior work that used agent-based simulations is given in Section 6. Finally, we conclude and discuss directions for future research in Section 7.

2 Related Work

In recent years we have witnessed a substantial increase in the use of autonomous agents in medium-scale and large-scale simulation systems. This is mainly due to the advantage of representing people and complex objects by agents that can interact among themselves and scale [49]. Consequently, several multi-agent software platforms have been suggested for the simulation of complex adaptive systems, such as SWARM [32], JADE [5], Robocup-Rescue [41] and Repast Symphony [34]. Common to all these systems is the simplified development while standard compliance is ensured through a comprehensive set of system services and agents.

Nevertheless, while the focus of the simulation tools is on the infrastructure, the task of reliably capturing the individual agents' behaviors has remained under the responsibility of the simulation designer. Despite the various implementations based on the above tools, individuals' behaviors have usually been modeled using statistical data [42], pre-specifying an agent's roles using a set of parameters according to which the agents act [30], pre-defined events and reactions [33], defining a number of basic behaviors from which all of the agents' more complex behaviors are constructed [40, 43] or using a combination of rules and finite state machines to control an agent's behavior using a layered approach [45]. While this latter approach has yielded many interesting results, it is problematic in scenarios where the agents are expected to exhibit human behavior in situations different from those used to collect the real data from which the strategies were derived. Furthermore, it is very difficult to use historical data to capture the different variations exhibited in people's behavior due to the large amount of different behaviors and the difficulty in assessing and categorizing the data. In this case, using only a sample could cause over-fitting the strategy only to the sample that was given.

Participatory simulations have been intensively studied as a means of modeling human societies [16]. In these simulations there are two types of entities active in the same environment: human agents and computational agents, both sharing the same roles. When using participation, the simulation developers can actually "enter" the simulation and "experience" it from the inside. Thus, implementation problems can be tackled and situations not previously thought of can be identified

and the resulting simulations can be optimized. There are several other reasons for wanting to add participation to simulation, such as learning about its complexity or learning about collective and individual behaviors in the simulation. The system is validated as a whole, as opposed to single agent validation. Examples of such usage were proposed in [23] where multiagent-based participatory design methodologies were used to test socially embedded systems. The simulations in that work took place in virtual space and involved a large number of simulated users. However, some of the simulated users were replaced by avatars which were controlled by human subjects sitting in front of their personal computers during the simulation. The approach taken by participatory design further motivates the approach presented in this paper. It demonstrates that including humans in the simulation system and the simulation process can improve the simulation's realism. However, in participatory design methodologies people were active during the simulation itself, while in our approach agents are created prior to the simulation, cloning the players' strategy.

The use of PDAs in multi-agent system literature is quite extensive. For example, in Kasbah [12] PDAs that buy and sell were used to evaluate an electronic marketplace. In Colored Trails [21], PDAs were used for reasoning about players' personalities in uncertain environments. Other works, e.g., [26, 25, 11, 38] used PDAs to evaluate specific mechanisms in various domains such as security algorithms, automated negotiators. The concept of having people program strategies into agents has emerged from behavior economics, where the "strategy method" paradigm, according to which people state their action for every possible situation that may arise, is widely used [39, 36]. All the above works focused mainly on the performance of the system as a whole and did not attempt to evaluate the individual strategies that were obtained in terms of resemblance (to the strategy of the person who programmed the agent) and diversity of the total set obtained.

The main motivation for having people program complex agents' behaviors is the possibility that the resulting strategy will correspond to their own. This, however, is not straightforward. Evidence of discrepancies between actual and reported human behavior is a prevalent theme in research originating in other various domains, in particular in metacognition research [22]. Several examples of these discrepancies include over-reporting of political participation (roughly 25% of non-voters report of having voted immediately after an election [7]) or contrasting results between self-reported and performance-based levels of physical limitations (there is weak to moderate association between performance-based and self-reported measures in motor functioning [24]). While the above works provide evidence of differences between reported behavior and actual behavior,

they differ from our work in reference to two aspects. First, our work initially asserts that there are some inherent differences between the two and focuses on the attempt to measure the level of difference. Second, the behaviors reported in earlier research are generally simpler and do not apply to strategic decision-making of the type we investigate. The method presented and exhibited in this paper is unique in giving people the opportunity to create agents that represent their own behavior.

3 PDA-based Method

Our research is based on recent work that has shown the successful use of PDAs as a key mechanism for substituting people in simulations by capturing people's strategy. In this section, we elaborate on the design process of the PDAs and their incorporation in simulation settings. All PDAs, detailed design instructions, code skeleton and simulation settings described in this research can be downloaded[1] to allow replication of the results, as well as a better understanding for future implementations. The method presented here is a synthesis of various guidelines and ideas found in prior PDA-based simulation implementations [26, 21, 25, 27].

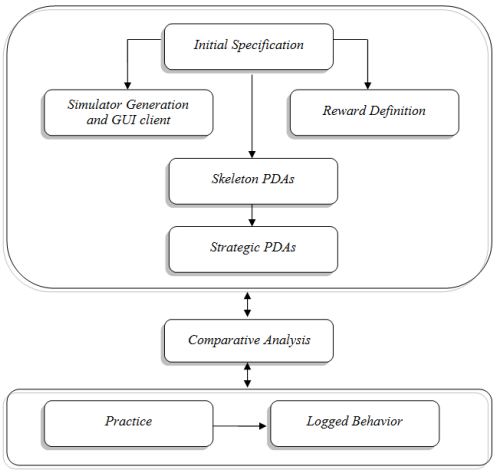
The key idea in PDA-based simulations is to have each PDA programmed by a different person. In general, the PDAs should be designed by a population that is as similar as possible to the people who are the subject of the simulation. We refer to these people as the *strategy designers*, as they embed their strategies in the PDAs. As we will demonstrate, the strategy designers need not be proficient in programming, as the translation of their strategy to a functional PDA can be done by a professional programmer.

The methodology is based on two layers as depicted in Figure 1a, which summarizes the method procedures. The first involves the development of the simulation itself and the PDAs, while the second is a validation and control layer, aimed at verifying that the PDAs and the system exhibit realistic behavior. Failure in the second layer requires repeating the agent development stage in the first layer.

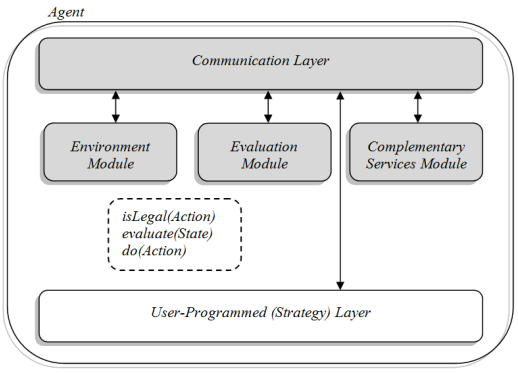
The first layer consists of the following phases: (a) initial specifications, (b) simulator generation, (c) skeleton PDAs, (d) reward definition and finally (e) generating strategic PDAs. We continue with a description of each such phase.

The ***initial specifications*** phase is the first phase of any PDA-based method. In this phase the key components of the system are defined, such as the set of play-

[1] `http://u.cs.biu.ac.il/~linraz/ParkingPDAs`



(a) Method procedures.



(b) A Skeleton PDA's Architecture. The modules filled with gray are fully coded and debugged when supplied to the strategy designers and the non-filled rectangle is the strategy layer which the strategy designer needs to program.

Figure 1: Procedures and infrastructure

ers, the possible set of states of each player, the actions each player can take, the observation of which each player is aware and the transition function or rules for each state and action. In this phase, a decision is made regarding the goal function of the agents to be developed. If the decision is to set the goal function externally, then at this stage the simulation designer needs to define the individual reward function. Instead of reward functions, which need to be maximized, in some environment the players are assigned cost functions which should be minimized.

The next phase is the ***simulator generation***. The subtle consideration at this phase with respect to a PDA-based method is the need to externalize the interfaces of the simulator in order to enable the incorporation of the PDAs later. The simulator is responsible for updating the world state at each time step according to the actions that the players take by activating the transition function, or by following a set of rules defined in the preceding phase. It also supplies the players with their partial observations of the current simulated world state and finally assigns a reward to each player.

The following phase is the key step in the design of the ***PDAs' core***. In this phase, a ***skeleton PDA*** is created. The skeleton PDA (see its architecture in Figure 1b) is equipped with all the functionalities needed to participate in a simulation run, such as communication with the server and observation of the environment.

It only lacks the strategy that determines which actions it will take.

If the agents' goal is pre-set by the system designer (e.g., in terms of individual utility function) then the process can continue to the next phase. Otherwise, e.g., if one of the simulation goals is to test social welfare, then user preferences or utilities might need to be elicited. These will be used to calculate social welfare when the simulation ends. The elicitation can be performed either by asking the strategy designer to express the PDAs' utility function directly or by using any elicitation method found in the literature (e.g., [9]).

The final phase is ***generating strategic PDAs***. In this step, people integrate their strategies into the core of the PDAs previously created, thereby producing a fully functional strategic PDA, or a *clone PDA*. Based on the choice made with respect to the reward definition, the instructions to the designer can either be (i) to program the strategy which maximizes a given reward function or (ii) to program the strategy which is as *similar* as possible to the one actually used in real life.

The second layer consists of (a) practicing and (b) logging. In the ***practicing*** phase, people can become familiarized with the simulated environment. This phase might be crucial in situations where the simulated environment is not common and straightforward. In the ***logging*** phase, participants are requested to play the game themselves (as avatars, using a designated GUI) and all actions are logged. This step enables the simulation designer to test the degree of reliability of the *clone PDAs* – i.e., whether PDAs can be trusted to behave similar to some person's behavior in a certain environment – that are produced in a later step. This degree can be evaluated based on a similarity measure, taking into account the behavior of people and the PDAs in the same world states, as detailed in Section 4. The closer the strategy of the clone PDA to the behavior of the person who designed it the better.

The whole process needs to be repeated for each variation of the initial setting that the simulation designer wants to test. In order to avoid bias and carryover effects, it is recommended that repeated experiments be conducted with different people.

4 Evaluating People's Ability to Capture their Strategy in PDAs

One of the main contributions of this paper is the provision of a quantitative validation of the ability of individuals to reliably capture their strategies using PDAs.

To this end, comprehensive experimentation was conducted, measuring the similarity of the behaviors displayed by people in four different games and hundreds of game sessions with the strategies embedded in their PDAs and actions made by them in identical game states.

To compare the behavior of people with that of the PDAs, we introduce the notion of *similarity*. We compare the actions the players choose to those of their PDAs at similar decision settings. The similarity measure is binary in the individual-decision level (receiving 1 if the same decision made in the exact same decision setting and otherwise 0). The level of similarity between an agent and a person is the average similarity over the entire set of similar decision situations the two encountered.

4.1 Description of the Games

The four games used for the experiments were: (a) Blackjack, (b) cost search, (c) repeated lottery, and (d) parking allocation game. These games differ both in their complexity and their strategic characteristics. They all can be described as games in which rule-based strategies can be applied, and were chosen as representative of most real-life environments, which share this property. The simulator server, GUI interface, GUI clients and skeleton PDAs for the games were generated and implemented in Java.

4.1.1 Blackjack

This game follows the rules of the classic Blackjack game[2], except that there are no stakes involved. The same deck of cards is used until all the cards are dealt. Then a new deck is shuffled. The players' aim is simply to win the game.

The game involves two players, the person playing and the dealer. The player can either *Hit* or *Stand*, while the reward function for both players assigns 1 for winning the game and 0 for losing it. At each time the player is aware of her own cards, the dealer's cards which are shown and all cards previously drawn. In this game, the player has incomplete information since she does not know the order of the cards still in the deck. Instead of defining a transition function, we employ the rules of Blackjack. The optimal strategy for maximizing the expected reward involves tracking the ratio of high cards to low cards and calculating the probabilities of the next cards to be drawn [44].

[2] `http://wizardofodds.com/blackjack`

The relative complexity of the game stems from the fact that each game state affects the next. While cards dealt cannot be re-dealt (until the deck is re-shuffled), it is still difficult for the average person to calculate the probabilities of the different cards being drawn based on the flow of the game.

4.1.2 Cost Search

In this game, the player is instructed to travel between stores in order to buy a commodity (e.g., a television). The player must personally visit the stores in order to observe the posted price of the product. All prices are drawn from a normal distribution function with a mean and standard deviation known to the person playing (the values used in our experiments were $\mu = 1000$ and $\sigma^2 = 200$). The player needs to decide when to stop the search, resulting in a purchase in the cheapest store visited until that point in time. The player must purchase the product before a known deadline and must also take into consideration the cost of the search itself, represented by the loss of income (e.g., the hourly salary that could have been earned working instead of searching). The goal of the player is to minimize the overall cost of the process (the sum of the product cost and the aggregated alternative cost along the search).

This game involves a single player and complete information. The set of actions the player can take is either *Continue* or *Stop*. The player's cost function (replacing the reward function) to be minimized is the amount of money paid for the commodity and the cost of the search itself up until the purchase time. If a purchase has not been made, the cost assigned is ∞. The transition function in this game dictates that if the player did not buy the product in time step $t-1$ and chose *Stop*, she must have bought the product. If the player did not buy the product in time step $t-1$ and chose *Continue*, she must have chosen to explore yet another store.

From the strategic point of view, the game is played under a time constraint rather than against an opponent. Solving the game theoretically in order to extract the optimal search strategy can be done using an instance of Pandora's problem [47], resulting in a stationary threshold below which the search should be terminated.

4.1.3 Repeated Lottery

In the Repeated Lottery game [18] the player is initially allotted a budget of $\$100$ from which she needs to decide on an amount to bet. The winning probability is

0.6, in which case the bet is doubled and returned to the user (i.e., the expected return is positive and equals 1.2). Each game is played for a maximum of 10 betting rounds, or until the player loses all her money. The player's goal is to maximize the amount of money she retains at the end of the game.

This game also involves a single player and complete information. The set of actions available to the player at each time step is continuous (the amount to bet) and depends on the amount of money earned so far. The player's reward function assigns her with the amount of money she has at the end of the game. Since the probabilities of winning and losing do not change throughout the game and the expected return of every bet is positive, the optimal betting strategy is to always bet the entire available budget.

4.1.4 Parking Allocation

In this game the player is instructed to park a car in a dynamic parking lot where cars are continually entering and exiting. The parking lot has 4 Entrance-Exit points as well as six rows of parking spaces, each with 70 aligned parking spaces, and one row with 35 aligned spaces. That is, a total of 455 available parking spaces. Traffic lanes have specific traffic direction(s). A front door located at one of the sides serves as a sole foot entrance/exit point. Foot traffic may proceed in either direction but both vehicle and foot traffic are restricted to the traffic lanes. The player is a single driver entering the parking lot, while other drivers are also looking for parking spaces. During all stages of the game, the player has limited visibility of other drivers and the available parking spaces in her vicinity. Before the game starts the player is asked to define her cost function for each parking space in terms of the importance of search time, distance from foot exit and distance from car exit. The goal of the player is to park in an available parking space while minimizing her individual cost function. Due to the complexity and richness of this domain an in-depth description and analysis is given in Section 5.

4.2 Experimental Setting

As in the PDA-based method, we allowed players to *practice* playing each game, after they had thoroughly reviewed the rules of the games, until they felt confident that they had understood them and had a well-formed strategy for the game.

Recall that the primary goal of the experiment was to evaluate the strategy designers' ability to capture their strategies in PDAs. Thus, we also employed a *logging phase*, preceding development of the PDAS' strategies. In the logging

phase, the players were instructed to play at least 20 runs in each game while all of their actions were logged into files. At this stage, the strategy designers were unaware of the fact that they would, later on, be instructed to program PDAs and that their actions had been logged.

We were also interested in evaluating which technique would generate the PDAs most similar in behavior to their designers. One possible technique was to instruct the designers to program the best strategy, that is, to maximize the reward function. The second option was to instruct them to program a strategy which is as similar as possible to the strategy that the designer believes she might actually use. Thus, the strategy designers were divided into two separate groups: a *Clone* group, in which participants were told that their reward will be fully correlated to the similarity between the way their agent acts in future games and the way that they had acted in the earlier games; and the *Optimize* group, in which participants were instructed to design a strategy layer that would minimize the agent's cost function (or maximize the expected benefit, depending on the game used) and were told that they would be evaluated based on the performance of their agents in the games.

In accordance with the method described earlier for designing strategic PDAs, both groups received agent-based clients on the same server on which they played earlier and were requested to develop their strategy layers accordingly. The log files produced in the earlier step were used in order to provide the agents with game states similar to those in which their designers made their own decisions, as part of the PDAs' evaluation.

Next we asked the participants of the *Clone* group to identify possible complementary tools or data that they believed could have improved the performance of their agents (in terms of similarity to their played strategy). Most respondents identified the logged behavior as the main data needed, so they were offered access to the log files collected previously and were instructed to update and resubmit their agents, which we will refer to as *logged cloning* PDAs. This process allowed us to evaluate how the implemented strategy might be affected by the availability of a certain knowledge base. The new agents received were tested and re-evaluated. In order to avoid log-based strategies (i.e., referring to specific world states observed in the log file), the code of all of the agents received in this phase was manually reviewed.

All in all, 41 senior computer science undergraduates participated in this experiment, generating 123 PDAs that were included in the analysis. The task was part of an undergraduate workshop curriculum: 27 students belonged to the *Clone* group, while the remaining 14 were assigned to the *Optimize* group, serving as a

control group.[3] A total of $3,009$ runs were played and logged during the experiment.

4.3 Similarity Performance

We can evaluate the similarity of the designers and their PDAs with the results of the experiment. Recall that the similarity function measures the similarity between a human player's behavior and her PDA's strategy. Since the strategy might use probabilistic functions, numerous runs must be conducted to gain statistical results. To this end, for each triplet of runs (i.e., state, observations and the human player's chosen action), the PDA (a) was put in a state identical to that of the human player, (b) was given the same observations as was the human player and (c) was required to decide on a single action. This was done 100 times for each choice of action taken by the human player. The percentage of actions taken by the PDA that were identical to the actions taken by the human player was considered to be a measure of the PDA's probability of taking the same action as the human player with respect to the given action choice. The average of these probabilities for a given run was used as the similarity measure. In the *Repeated Lottery* game, since the bet is a continuous variable, we decided that an action which would deviate within some $\alpha\%$ range[4] would be considered identical.

It should be noted that a 100% similarity between the behavior of the designers and their PDAs is not feasible mainly due to two reasons. First, people's strategies always involve some "noise", that is, decisions are made which do not follow the overall strategy, for numerous reasons (e.g., not thinking the move through, losing concentration, dealing with a situation never encountered before, etc.). Second, people do not always follow the same strategy every time they are presented with the same situation [15]. In the absence of a set method by which each participant plays thousands of games, some discrepancy is likely to occur.

Figure 2 presents the similarity obtained in the different games. Note that the figure presents the results of the average similarity obtained by all PDAs. As discussed later, the similarity measure obtains an even greater value if recalculated without some of the PDAs. Despite the small number of PDAs in this latter group, their inclusion has a substantial (negative) effect on the overall average.

Figure 2 shows that higher similarities were obtained for the *Clone* group in

[3] Since this research focuses on the *Clone* group, the participants were divided by a ratio of 1:2.

[4] We used a 20% range, which was set arbitrarily. Obviously a greater percentage would have resulted in greater similarity and vice versa.

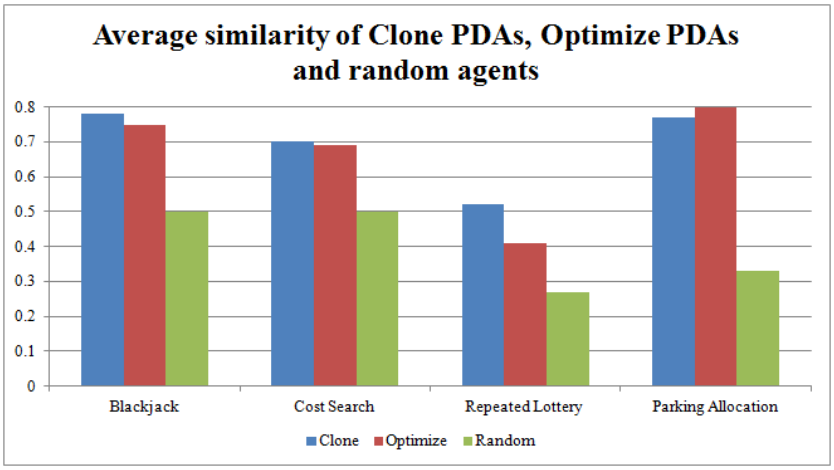


Figure 2: Average similarity of *Clone* PDAs, *Optimize* PDAs and random agents per game.

most games, although the difference was not found to be statistically significant for all four games when applying the *t-test*. The results demonstrate that people in real-life actually use what they believe to be the optimal strategy[5]. Furthermore, people's optimal strategy does not greatly rely on the computational and memory resources that are quite unlimited for computer agents. Therefore, in these types of games, people can be instructed to design the agent with what they perceive to be the optimal strategy, rather than their own observed strategy (which might be easier for them to implement). In addition, to evaluate the quality of the comparison, we also compared the similarity of people to that of a random agent in all four games. In all of the games we obtained significantly lower similarity for the random agents compared to the similarities between the people and their PDAs.

Reviewing the results and the strategies designed, we found that only a small portion of the population is responsible for most of the degradation in the average similarity. Figure 3 displays this phenomenon, detailing the percentage of participants who achieved a closeness score greater than the different thresholds given in the first column. From the figure, we can see how the "bad strategy producers" affect the results. For example, excluding the 15% participants that scored below a 50% closeness in the parking allocation game results in a new similarity value of 0.85 (in comparison to 0.77). Generally, the results show that a relatively large portion of the population, in all games, managed to achieve an average similarity

[5]While this may seem trivial, we would like to emphasize that there is evidence of discrepancy between what people describe as their strategy and what they actually do in real life [22].

Figure 3: The percentage and average similarity of participants and PDAs based on the given threshold of similarity.

higher than 0.6. The average similarity obtained by the PDAs with a closeness score of 50% or higher ranged between 0.77-0.82 (game-dependent).

As we explained in Section 4.2, people were permitted to re-submit their PDAs after gaining access to previous data. Of 27 participants, 7, 13, 16 and 7 decided to fine tune their PDA's strategy in the *Blackjack*, *Cost Search*, *Repeated Lottery* and *Parking Allocation* domains, respectively. Figure 4 presents a comparison of the average similarity of these participants (per game), comparing their initial and revised PDAs. The results demonstrate that by viewing the discrepancies between their own actions and their PDAs' actions, participants were able to improve the similarity of their PDAs' strategy to some extent in all four games.

In the *Repeated Lottery* and *Parking Allocation* domain, there was a significant improvement in the similarity of the revised PDAs (using the paired *t-test* with $p <$

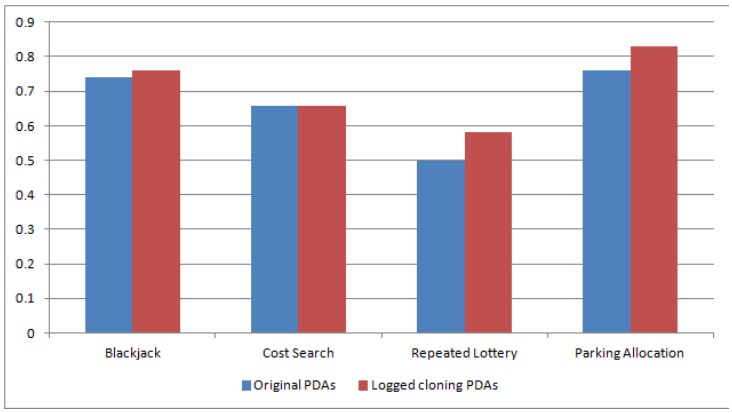


Figure 4: Average similarity obtained by the participants who attempted to improve their PDAs.

0.001 for both domains). For the other two games, no significant difference was found. The results of this experimentation suggest that, generally, a large portion of the population is capable of capturing their exhibited strategy with a reasonable level of precision. The results also support the fact that supplying designers with past logged behavior can improve the similarity between their exhibited behavior and their PDAs' strategy.

4.4 Capturing People's Strategies by Means of a Proxy

To bolster our confidence in the efficacy of the PDA-based approach, we continued with an additional experiment. In this experiment we were interested in verifying that even people without any programming background can design strategies which professional programmers can implement on their behalf, resulting in PDAs that exhibit similar behavior to that of their designer. In this case, the *Cost Search* game was used as the test-bed. There were 21 participants who took part in this experiment, ranging in age from 17.5 to over 50. Their occupations also varied and included a high school student, secretaries, a teacher, a nurse, engineers, a house-keeper, non-computer science students and an occupational therapist.

The same experiment methodology was followed in this experiment, except that the participants described their strategies to a professional programmer who embedded them into their PDAs' strategy layers. The average similarity obtained by participants who self-programmed their PDAs (27 agents) was 0.7, compared to 0.69 of those who had a professional programmer as a proxy (21 agents). The

results indicate almost the same average similarity as in the previous experiment.

These results further strengthen the claim that a PDA-based method enables the creation of PDAs whose behavior highly resembles their designers. The results also show that the clone PDAs need not be programmed by the designer; they can be conveyed by people lacking a computer science background to professional programmers, serving as their proxies.

5 Parking Space Search Domain

The complexity and richness of the parking space search domain motivated us to report in depth on the specific implementations, considerations and aspects of PDA-based design which emerged in this setting.

Recall the problem described in Section 4.1.4. It also has several interesting characteristics from the simulation system perspective. First, it is a common task, in many cases experienced on a daily basis. Second, the individual's decision-making is based on a searching strategy that involves spatial search along with various exploitations versus exploration considerations (e.g., opportunities that have already been identified and their estimated relevance and beliefs concerning future opportunities). Due to the dynamic settings of the domain, even though a person may have performed this task many times, each attempt is unique in terms of the available parking spaces, their location and the other cars that concurrently search for parking spaces.

This domain embeds many different strategies that are pursued by people (e.g., some people might strive to park in the first available space, others will prefer to look for a parking space closer to their destination) [48]. The complexity only increases as one's behavior might be affected by that of others, since such influence can cause frequent changes and unpredictable behavior. An optimal solution for this problem is unavailable due both to its complexity and the different cost functions representing different people with diverse concepts of optimality.

Parking lot simulation is useful when designing a new parking lot or reconstructing an existing one. In recent years, parking lot simulations have also become an efficient means for estimating the usefulness of new information technologies aiming to help drivers park more efficiently [3]. Previous work on this topic has placed little emphasis on extracting user parking strategies. For example, PARKSIM, which is the earliest parking space search simulation system, uses a network model to represent the driver's behavior [48]. In this system, all of the driving agents share the same cost function and use the same parking strat-

egy. Cassady and Kobza [10] developed a probabilistic approach to parking space search strategies. They used a specific example of parking space availability probabilities in order to analyze two predefined parking strategies. Common to most works in this domain is that the simulations they use generate only a few individual and unique strategies.

All of this only increases the need for reliable simulations that can reflect human behavior well. The PDA-based method allows the generation of a large collection of realistic parking strategies. Moreover, using the method it is easy to evaluate, as we show, the effect of new information technology on the different strategies. The information technology provided in the simulation is given as part of the *complementary services module*, as specified in the recommended architecture. We tested one information technology, denoted *info*, which we compared to a setting wherein no information technology was used, denoted *noInfo*. In *noInfo* only the general parking lot map and limited viewing frame was provided to players at every time step, while *info* repeatedly provided the players (i.e., drivers) with an *exact parking lot map at every time step*, showing both the free parking spaces and the exact location of all other agents. *info* uses GPS tools, which rely on sensors embedded throughout the parking lot. By measuring the improvement in the agents' performance when using the new technology, *info*, we were able to estimate its value.

5.1 Simulator Generation

Several sets of parameters were required for the simulator developed. The first set of parameters determines the parking lot structure (e.g., overall size, internal structure, the number and location of entrances and exit points, the alignment of different lanes and the permitted driving directions). All of these parameters are part of the real-time environment object and the environment module. The second set of parameters includes the rate of cars arriving and exiting the parking lot at each time step, used by the transition rules, as well as the players' view of their vicinity from any location (e.g., partial observations). These parameters are unknown to the agents.

The parking lot structure is represented as a map consisting of a matrix, where each point (x, y) stands for a fragment of a lane or a parking space, which is associated with a different walking distance to the front door. For every time step, each agent located at point (x, y) needs to choose an action to take from the possible action set A_i which includes: (i) move to a subsequent point (x', y'), (ii) park in a subsequent parking space (x', y') or (iii) stay put. Agents currently

not in the parking lot are considered to have chosen the fourth action, '*Null*'. The action chosen by the agent is reported to the simulation manager through the communication layer and executed only if legal (i.e., a movement according to the permitted driving directions to a place not blocked by another agent or a vacant parking space). At each time step the environment module supplies the agent with an updated partial observation of the area around it (the "*viewing frame*"). In addition, at each time step, the simulation manager extracts the viewing frame from the real-time environment object separately for each agent (this is done in the order by which the agents enter the parking lot). Following the strategic decision made by each agent, the simulation manager updates the current state in the real-time environment object.

New agents are instantiated by the simulation manager based on the arrival probability defined for each entry point as part of the state transition rules. The type of each instantiated agent (i.e., its specific implementation) is randomly selected from a pool of types, as defined by the transition rule. Upon instantiation, each agent operates according to its strategy in order to find a suitable parking space. Upon parking, the agent becomes inactive. The car represented by this agent will leave the parking lot based on the exiting probability defined for every exit point, in a manner similar to the way new agents are instantiated (i.e., each exit point has an exiting probability and the exiting car is randomly selected from all the parked ones). Since we are interested only in the parking search strategy, exiting cars do not drive through the lot. Instead, the parking space is simply considered vacant at some time point. In this manner we have tight control of the entrance and exit rates.

5.2 Definition of Cost Functions

The next step of the method requires generation of the skeleton PDAs. Our strategy designers' group consisted of 34 students (no one from this group participated in the experiment reported in the previous section), all of whom were senior computer science undergraduates. Each student received a skeleton PDA and had to implement her strategy for the case of receiving additional information and the case of receiving no information. The students were graded according to their agent's performance.

The strategy designers were requested to specify their cost functions. This was required as the goal of this experiment was to evaluate the emergent behavior from the social welfare perspective. After analyzing the different functions proposed, three parameters were extracted:

Walking distance. The distance a person has to walk from the parking space to the front door. Since all pedestrian traffic is restricted to the traffic lanes, a Manhattan Distance function was used.

Search time. The time a person spends searching for a parking space (measured as the number of time steps from the time the agent enters the parking lot until it parks).

Distance from exit. The distance a person has to drive from her parked car to the nearest exit point. Note that the Manhattan Distance function is irrelevant in this case since the parking lot's traffic directions must be taken into consideration.

These three parameters indeed resemble (with several changes) the measures reported by [10]. Assuming that both driving inside the parking lot from one point to its adjunct point and walking the same distance take the same time duration, we consider both distance and time to be measured by time alone.

Using these parameters, each player's goal was to minimize its cost, represented as:

$$Cost(WalkTime, SearchTime, ExitTime) =$$

$$C_1 \cdot WalkTime + C_2 \cdot SearchTime + C_3 \cdot ExitTime$$

where C_i is the weight given for each cost component ($\Sigma C_i = 1$). Each strategy designer defined her specific cost function, R^i, by determining C_1, C_2 and C_3 according to her own preferences. This function served as the core of the Evaluation Module.

5.3 Generating Strategic PDAs

As we described in Section 4.3, the assignment of minimizing the cost of *Parking* yielded better clone PDAs with higher similarity to the strategy depicting the designers' own behavior than the assignment of imitating it. Consequently, the entire group of strategy designers was required to minimize their agents' costs throughout this experiment. Each student was asked to implement the best strategy she could think of in order to minimize her agent's costs in the *noInfo* settings.

The strategy designers were supplied with appropriate tools for testing and debugging their PDAs. These tools were mainly intended to test the validity of the PDAs' strategy and were not aimed at supplying any feedback on their performance. The students were not allowed to exchange PDAs among themselves

for testing purposes and were instructed not to talk about their strategy with other strategy designers attending the workshop. At this stage each PDA was equipped with a specific parking strategy that was programmed by one of the strategy designers, according to her perception of what was considered a "good" parking space.

Finally, each strategy designer was introduced to a new (previously unknown) information technology, *info*, and was requested to revise her strategy accordingly, taking advantage of the new supplementary information. As we report in Section 5.5, user feedback reveals that once the basic agent strategy generation was completed, the transition to a new information technology was quite straightforward in terms of programming efforts.

5.4 Evaluation and Results

In order to evaluate the agents' performance when using the different information technologies, three initial occupancy percentages were used as well as three sets of Entry-Exit probabilities. The initial occupancy percentages used were 90%, 75% and 60%. Operating in a parking lot with a relatively high initial occupancy percentage is expected to result in greater difficulty for the agents to find a good parking space. The three sets of Entry-Exit probabilities simulated: (a) a scenario with similar entry and exit probabilities (i.e., a balanced parking lot), (b) a scenario where more cars entered the parking lot than left at a given time, and (c) a scenario where more cars left the parking lot than entered (on average) at every time step. The greater the exit probability (relative to the entrance probability), the easier it was for the agents to find an available parking space.

The idea of varying the initial parking lot occupancy percentage and the entry-exit rates of cars stems from the need to emulate different parts of the day or week. For example, when arriving at a shopping center's parking lot on a Sunday afternoon, we usually expect to encounter a high level of occupancy. Similarly, if it is a factory's parking lot, mornings are usually associated with a relatively high ratio between car arrival (entrance) and departure (exit) rates, and afternoons are usually characterized by a reverse pattern.

Combining each of the initial occupancy rates with each of the entrance/exit probability ratio sets results in nine different settings. In order to compare *noInfo* and *info*, the same cars must enter and exit the parking lot at the same time steps for every setting. Thus, 9 scenario files were constructed by the simulator (three for every transition rule).

For every setting, the same nine scenario files were used. Each simulation

	Improvement
avg	11.57%
90%	8.86%
75%	10.99%
60%	14.66%
en=ex	16.62%
en>ex	2.74%
en<ex	20.7%

(a) Improvement (i.e., lower costs) of *info* relative to *noInfo*, in percentages. The first row represents the average costs. The rest of the table presents a breakdown according to the different values used for the initial occupancy and the Entry-Exit rate.

	noInfo	*info*
Search Time	58.87	53.54
Walk Time	22.85	23.16
Overall Time	81.72	76.70

(b) Central Management Measurement Results

Table 1: *info* and *noInfo* results.

execution terminated after 3,500 agents entered and parked. At the end of each simulation run, the total cost of the parking process was averaged over all of the different agents (based on their specific cost functions) participating in the run.

The average costs received for each setting were *noInfo*=37.32 and *info*=35.24. The system evaluation was based on both absolute performance and the magnitude of the relative improvement obtained. A performance baseline was required in order to calculate the relative improvement. Obviously the baseline cannot be zero, since even for the most efficient set of agents there is still a lower theoretical limit for the system's performance[6] (e.g., whatever the theoretical allocation that maximizes the overall social welfare may be, players still need to park and walk to the main entrance, reflecting some cost). The lower bound used was the centralized allocation of parking spaces to agents when they were fully cooperative and obeyed the instructions received from the allocator. The performance (i.e., the average cost) calculated for the lower bound was 19.20.

Table 1a presents the improvement of *info* relative to *noInfo*, in percentages.

[6]That is, the average performance of all of the agents in all of the different scenarios for a specific information technology.

The first row represents the improvement in the average costs of all nine combinations of occupancy and Entry-Exit rates. The next three rows represent the improvement in the average costs where the initial occupancy was fixed and the Entry-Exit rates were varied. The last three rows represent the improvement in the average costs where the Entry-Exit rate was fixed and the initial occupancy was varied. As expected, the results support the fact that the transition from no information to full information results in great improvement (using the paired *t-test* with $p < 0.001$).

5.4.1 Evaluating Different Metrics

The results described above represent the PDAs' performance from an individual perspective. Nevertheless, in many cases, the simulation designer is interested in different measures. For example, a company's parking lot manager would try to minimize the amount of time workers spend in the parking lot, in order to maximize the amount of time they spend working. We now demonstrate an analysis of such a scenario.

The performance measure when the goal is to minimize the amount of time workers spend in the parking lot is defined as:

$$Cost_{mang} = \sum_{agent_i} WalkTime(agent_i) + SearchTime(agent_i)$$

assuming again that it takes the same time to walk and drive similar distances. As before, the goal is to minimize the cost. Table 1b presents the average costs of *noInfo* and *info* in terms of search time, walking time and overall time, using the same simulation results as above.

The results show that there was almost no change in the average walking distance with or without the additional information. This is because, as presented in detail in the utility function determination description, people base their strategies not solely on the walking time, but also on other parameters, such as search time, randomization and predefined route. These strategies affected the distribution of available parking spaces in the parking lot. Thus, even though the availability of information indeed reduced the average searching time, it did not reduce the average walking time to the exit.

5.5 Additional User Feedback

Throughout the entire experimentation, much emphasis was placed on obtaining feedback from participants through the use of questionnaires as well as interviews,

if required. Insights based on the analysis of this complementary user feedback are presented below. User feedback is an important aspect concerning the implementation, especially as the process of using and implementing PDAs highly relies on users.

First, we were interested in checking the generality of the designed strategies which resulted from the process, i.e., to what extent the strategies used depended on the specific structure of the parking lot supplied. We were also interested in investigating the ease of use of our skeleton PDAs and the general workload for people translating their real-life strategies into codes.

In order to evaluate the generality of their agents, the strategy designers were asked whether they would have changed their implementation if the parking lot used for the experiments was different. Strategy designers that answered positively were requested to detail parking lot characteristics that could have affected their strategy and how their strategy would have changed in these potentially new settings. The analysis of the answers reveals that 74% of the strategy designers considered their strategies to be independent of the parking lot structure. For those who considered their strategy to be parking lot specific, the most dominating factor was the ratio between the primary driving routes and the secondary driving routes, as well as the driving directions on these routes. Additional parking lot characteristics that were noted as factors that would result in strategy changes were the potential division of the parking lot into levels, different numbers and location of entrances and exits, different geographical distribution of parking spaces and the option to drive the car in reverse in a lane. None of the strategy designers indicated that their strategy would change as a function of other parameters, such as the entrance and exit rates. These results suggest many opportunities for using the developed set of agents in other diverse parking lot settings.

The last set of questions focused on the ease of use of the mechanisms supplied for programming the agents' strategies. Strategy designers were asked to estimate their overall effort in programming each of the different agent versions and the ease of programming on top of the skeleton PDA that was supplied to them. Almost all of the strategy designers emphasized that a considerable amount of the time invested in writing their agents was dedicated to designing the agent's strategy, while the actual programming process was short. While 80% of the participants stated that writing each agent required a moderate amount of time (a few hours), the rest estimated it would take up to three days. In addition, 72% of the strategy designers indicated that the development was intuitive, and that all the necessary functionalities for their strategies were initially supported. Furthermore, 21 strategy designers reported that if they would have to repeat this assignment,

the average time for completing it would be the same or would decrease. This feedback is important and reveals that researchers should invest enough time on creating a fully functional skeleton PDA, which in turn will allow the strategy designers to indeed focus most of their time on the strategy design than on other issues.

5.6 Validating the Realism of the Simulation

To validate the realism obtained with the parking allocation simulation, we carried out a complementary subjective study with four domain experts from the Rabin Medical Center in Petach Tikva, Israel, one of the biggest medical centers in Israel. The center has five parking lots, four of which are single story lots (as in our experiment) and the fifth is an underground two-story parking lot. We surveyed the deputy security officer of the center, two security employees whose job is to monitor the parking lots (using video surveillance, round the clock) and a police officer. The experts were presented with the GUI manager of the system and watched the behavior of the PDA-based simulation in eight random runs (each run included the entrance and parking of 200-300 cars). The experts were asked to evaluate the realism of the simulation. The feedback received from the experts suggests that the system successfully managed to simulate the dynamics of a realistic parking lot and that the behaviors exhibited by the different PDAs did in fact resemble those of people. Furthermore, the experts even pointed out that some of the observed strategies are ones that they might have adopted themselves in similar settings. As most subjective tests, the results do not supply a strong quantitative measure. However, along with the similarity test experiments, they are important complementary evidence of the ability of the PDA-based simulation to capture people's strategies.

6 PDAs as a Means of Enriching Strategies Set

The main premise of the PDA-based simulation, as discussed throughout the paper, is the ability to generate a wide variety of different strategic behaviors. This is of most importance whenever simulation with human-subjects is required, as in the real world different people act in different ways when put in the same situation. The ability to replace people's participation in the simulation with PDAs can thus be enabled only if the set of PDAs used can generate the same richness of strategic behaviors obtained with people. In this section we evaluate this aspect

of the PDA-based simulation. The evaluation is based on the strategic characteristics embedded in the PDAs that were developed in the experiments reported in this paper and a complementary analysis of individual performance. In the second part of this section, we compare the number and nature of strategies obtained with those of different strategies used in prior work.

6.1 Variety of Strategies

In order to analyze the full variety of behaviors embedded in our PDAs, we manually reviewed the code and documentation that were handed in with each developed agent. This review process revealed various ideas and implementations. Some examples of the different strategies used for each game are:

- *Blackjack*: (1) take some predetermined cards' sum thresholds into consideration, as well as the dealer's card and aces, (2) use probabilities and randomization when deciding whether to draw another card.
- *Cost Search*: (1) do not visit more than a pre-determined number of shops when calculating the total price, (2) visit an exact, pre-determined number of shops, without calculating the total price, (3) do not visit more than a pre-determined number of shops when calculating the total price, using also some randomization factor.
- *Repeated Lottery*: (1) never bet on the entire amount when using pre-defined betting sums, (2) use randomization only to decide whether or not to bet, but never bet the entire amount, (3) randomly choose an amount to bet from a predefined set of sums.
- *Parking Allocation*: (1) park if the space is near the pedestrian exit; as time goes by the agent randomly chooses between moving directions and changes course if the moving direction selected is blocked, (2) assign heuristic values to possible parking spaces, and proceed in the minimum cost direction, finally park in some position only if the cost assigned to the current position is smaller than two times the minimum cost assigned to any of the positions, (3) park in the first vacant space found, choose the driving direction randomly, maintain the courses already taken in order to change course if used three times.

The examples reflect various domain-specific and general elements that are used as part of the PDAs' strategies. Based on the review of all agents' codes and

documentation, we generated the following list of strategy characteristics used by any of the agents in each game:

- *Blackjack*: (1) use some predetermined cards' sum threshold, (2) use some predetermined cards' sum intervals, (3) take the dealer's cards into consideration, (4) consider an ace in a different manner than other cards, (5) take into consideration the number of cards in the player's hands, (6) act according to some pre-defined probabilities, and (7) make some random decisions.

- *Cost Search*: (1) consider the number of visited stores, (2) consider the product's price, (3) make some random decisions, (4) take the total price (search time and price) into consideration, and (5) do not calculate the cheapest price.

- *Repeated Lottery*: (1) predefined bets - absolute or relative to current amount, (2) make some random decisions, (3) never bet the total current budget, (4) bet according to some pre-defined probabilities, (5) rely on results of former bets, (6) calculate the first bet in a different manner than the other bets, (7) consider a number of intervals of a potential amount, and (8) bet very small bets in some cases.

- *Parking Allocation*: (1) park close to a pedestrian exit, (2) park in the first vacant space found, (3) park farther away from the pedestrian exit as time goes by, (4) make some random decisions, (5) park in the best located vacant space, (6) follow a predefined route, (7) change route if waiting too long, (8) use different strategies in different cases, and (9) willing to wait in place.

Table 2 gives the percentage of agents in which each characteristic was found, for each of the games. As can be observed from the table many of the agents combined several of these characteristic resulting in a rich set of strategies.

Additional evidence for the richness and diversity of the strategies produced with the PDA approach is given in Figure 5. The figure depicts the individual average performance achieved by each agent in the different games. The performance measure is game-dependent and is derived from the agent's goal in the game: For the cost search game the measure used is the overall cost (that the player attempts to minimize) which is the sum of the product price and the costs accumulated along the way. For the Blackjack game the measure used is the percentage of games in which the agent won (rather than the dealer). For the repeated lottery game the measure used is the amount of money the agent ended up with. Finally,

Strategy	% Agents
Cost Search	
Predefined number of visited stores	78%
Consider price	70%
Use randomization	22%
Consider total price	15%
Do not calculate cheapest price	7%
Repeated Lottery	
Predefined bets	89%
Use randomization	52%
Never bet 100%	48%
Use probabilities	33%
Consider previous rounds	33%
First bet(s) are exceptional	18%
Multiple intervals	18%
Very small bets in some of the cases	11%

Strategy	% Agents
Blackjack	
Single threshold	66%
Multiple intervals	33%
Consider dealer's cards	30%
Consider ace separately	15%
Consider number of cards in hand	4%
Use probabilities	4%
Use randomization	4%
Parking Allocation	
Park close to exit	52%
Park in the first vacant space found	44%
Willing to park further away as time goes by	33%
Use randomization	26%
If a better vacant space is observed go there	22%
Predefined route	15%
If standing in place too long – change courses	15%
Multiple strategies	11%
Willing to wait in place	7%

Table 2: Percentage of agents that implemented the different strategies in each game.

for the parking simulation, the measure used is the average parking search, despite the fact that the agents did not necessarily attempt to minimize that parameter. The reason for this latter choice is that in parking the agents used a self-defined utility function thus any agent's average individual cross-games utility is not comparable with the utility of others. The average search time, on the other hand, is an objective comparable measure and any difference in the average value obtained for this measure is an indication of the differences in the strategies used.

The agents used to generate the graphs in Figure 5 are the 27 "clone" agents that were developed for each game, except for the parking game where we had a total of 34 agents from the simulation that was developed. The figure depicts sub-

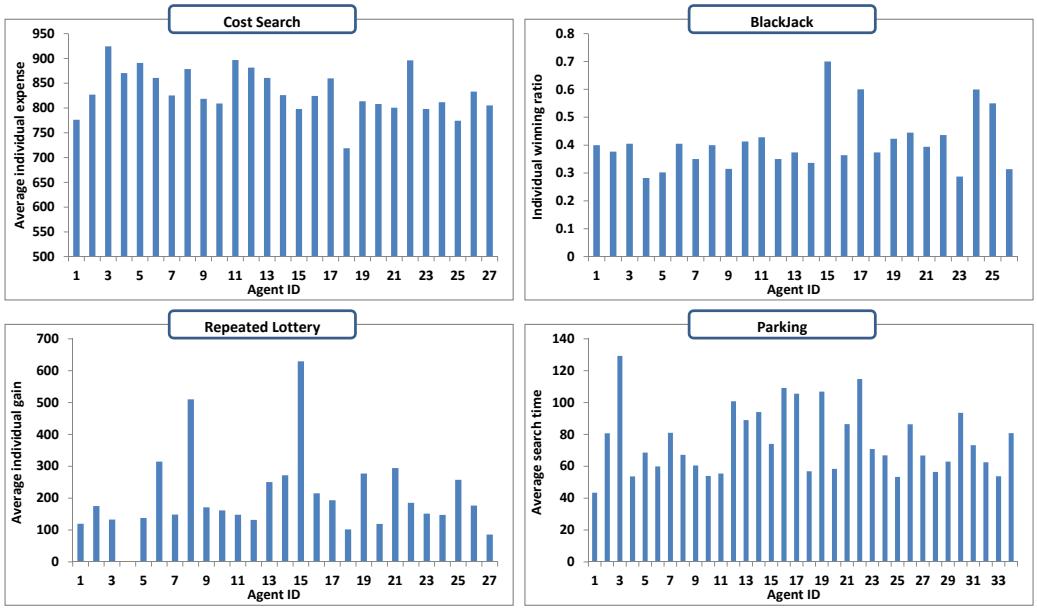


Figure 5: Individual agents' performances in: (a) the cost search game (measured as the average agent's individual expense); (b) the blackjack game (measured as the percentage of games the agent won); (c) the repeated lottery game (measured as the amount of money the agent ended up with); and (d) the parking simulation (based on the parking search time). Results reflect a high diversity in the agents' strategies.

stantial variance in the average performance obtained by different agents within each game, pointing to differences in the population as a whole. The fact that very few agents managed to achieve a similar performance within each game suggests that even in cases where different agents' strategies used the same ideas in their design (as demonstrated using the above classification of strategies), the actual strategy implementation was substantially different.

6.2 Comparison with Other Simulated Systems

A multi-agent simulation where agents' strategies are instantiated from a set of dozens of different strategies is a substantial advancement in comparison to prior literature. Table 3 illustrates the number and origin of the strategies implemented

in various systems found in the literature.[7] As observed from the table, the listed simulation systems use 1-6 different strategic behaviors. These sets of strategies are limited in comparison to those used in PDA-based simulations. Furthermore, in most of the examples given in the table, prior literature, rather than the simulated real world, is the origin of the implemented behaviors.

In PDA-based simulations, the number of implemented strategies depends on the number of strategy developers used. Thus, if more types of behavior are needed, all the simulation-designer needs to do is use an extended number of strategy developers. This approach is also beneficial in new research areas, where prior literature does not exist.

7 Discussion and Future Work

This paper details a coherent method for the development and use of PDAs for multi-agent system-based simulations with people. While participatory simulations [16, 23] enable programmed agents and human players to interact in a simulation, they still rely heavily on humans participating in each simulation. The PDA-based approach can disconnect the people from the actual runs and hence are more flexible for running large-scale simulations. We also detail two extensive experiments which empirically support two key aspects of this method, essentially (a) people's ability to reliably capture their strategy in PDAs and (b) the actual possibility of managing such a distributed development by so many different people.

The diversity of people, the uncertainty in their actions and the influence of people on others pose a great challenge in developing a set of behaviors with enough variety and realism that can reliably represent the richness of behaviors exhibited in real-life environments. Indeed, settings with PDAs have been proposed in recent years. Nonetheless, they have not been scaled to large simulations that can truly mimic a real-life domain. The method presented in this paper suggests the use of people familiar with the task at hand, as the designers of the PDAs. Naturally, this requires having access to individuals from the simulated population in order to develop the PDAs and that they will have some basic programming skills. We have shown that in cases where the individuals lack sufficient programming skills, a proficient programmer can embed their strategies into the PDAs. It is noteworthy that there are also situations in which using PDAs might be prone

[7] While the table cannot contain every simulation implementation that is reported in literature, it reliably represents the typical pattern.

Simulated system	Strategy origin	# of Strategy classes per experiment	# of Strategy classes overall
Combined electricity and gas markets [17]	Prior literature	3 models	3 models
Football teams [14]	Prior literature	2 optional behaviors per each 2 styles of play	2 optional behaviors per each 2 styles of play
Stressful crowd situations [35]	Prior literature	1, a behavior engine	1
A retail approach as part of the service industry [28]		2, proactive and reactive	2
Non-pharmaceutical in curtailing impacts of influenza [29]		1	1
Evacuation under different road network structures [13]		1	1
Crime prevention [8]	Prior literature	1	6
PARKSIM [48]		1, a network model	1
Cassady and Kobza's system [10]	Probabilistic approach	2	2
Simulation of disaster Response [31]	Prior literature, using an artificial neural network	Increasing number of roles and corresponding agents	4 different roles
Trading in high-frequency markets [1]	Pre-defined, based on zero intelligence and trading strategy	2	2
Traffic and mobility simulations [4]	Prior literature (e.g., Dijkstras shortest path)	1	1

Table 3: Strategies and their origin as they appear in other multi-agent literature.

to inaccuracy, especially in situations which people are not likely to encounter often (e.g., high stress situations, evacuation scenarios). Still, for a wide range an appropriate set of strategy designers can be easily recruited and the method is of much benefit.

We show that the use of PDAs enriches the simulation by facilitating the generation of a large set of different strategies, each relying on a diverse set of strategic characteristics. Moreover, it enables the simulation of behaviors more similar to human behaviors. This is in contrast to traditional agent-based simulation literature where very few diversified strategies are used, with hardly any validation as of the correlation of these strategies to the individual behavior reflected by real subjects.

In future research we hope to develop a tool which can identify the small portion of the population that causes discrepancy. The availability of such a tool could be very useful when using the presented method. Nonetheless, until such a tool is developed, the process of logging the designers' behavior strategy and comparing it to the clone PDAs' behavior can be followed. We believe that investing the time and design needed so that a small group of strategy designers can apply this process prior to a large-scale experiment is worthwhile, as it produces reliable clone PDAs and ensures far more credibility of the final simulation results.

These are the first steps on a long journey towards understanding people's perception of their strategic behavior and what means should be applied to facilitate the elicitation of their strategies. To the best of our knowledge, this is the only large-scale attempt of this type made so far to measure people's ability to elicit their strategic behavior. Future research should focus on testing additional/alternative methods for enhancing the ability of participants to reliably capture their strategies.

Acknowledgement

We thank Reuma Magori-Cohen for her support in the performance of the statistical tests. We also thank the Rabin Medical Center, Beilinson Campus, and its security department for their assistance with the realism verification.

References

[1] M. Aloud and E. Tsang. Modelling the trading behaviour in high-frequency markets. In *Computer Science and Electronic Engineering Conference*, pages 7–12, 2011.

[2] R. C. Arkin. Designing autonomous agents. *Robotics and Autonomous Systems*, 6(1-2):105–122, 1990.

[3] Y. Asakura and M. Kashiwadani. Effects of parking availability information on system performance:a simulation model approach. In *Vehicle Navigation and Information Systems Conference*, pages 251–254, 1994.

[4] M. Balmer, N. Cetin, and K. N. adn B. Raney. Towards truly agent-based traffic and mobility simulations. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 60–67, 2004.

[5] F. Bellifemine, A. Poggi, and G. Rimassa. Jade: a fipa2000 compliant agent development environment. In *Proceedings of the Fifth International conference on Autonomous agents*, pages 216–217, 2001.

[6] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of markov decision processes. *Mathematics of Operations Research*, 27(4):819–840, 2002.

[7] M. Bertrand and S. Mullainathan. Do people mean what they say? implications for subjective survey data. *American Economic Review*, 91(2):67–72, 2001.

[8] T. Bosse and C. Gerritsen. Comparing crime prevention strategies by agent-based simulation. In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 491–496, 2009.

[9] C. Boutilier, R. Patrascua, P. Poupart, and D. Schuurmans. Constraint-based optimization and utility elicitation using the minimax decision criterion. *Artificial Intelligence*, 170(8):686–713, 2006.

[10] C. R. Cassady and J. E. Kobza. A probablistic approach to evaluate strategies for selecting a parking space. *Transportation Science*, 32(1):30–42, 1998.

[11] M. Chalamish, D. Sarne, and S. Kraus. Programming agents as a means of capturing self-strategy. *Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 1161–1168, 2008.

[12] A. Chavez and P. Maes. Kasbah: An agent marketplace for buying and selling goods. In *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology*, pages 75–90, 1996.

[13] X. Chen and F. B. Zhan. Agent-based modeling and simulation of urban evacuation: Relative effectiveness of simultaneous and staged evacuation strategies. *Journal of the Operational Research Society*, 59:25–33, 2008.

[14] S. Dobson and J. Goddard. Strategic behaviour and risk taking in football. Working Paper 0805, University of Crete, 2008.

[15] F. T. Dolbear and L. B. Lave. Inconsistent behavior in lottery choice experiments. *Behavioral Science*, 12(1):14–23, 1967.

[16] A. Drogoul, D. Vanbergue, and T. Meurisse. Multi-agent based simulation: Where are the agents? In *Proceedings of the Third International Workshop on Multi-agent based simulation*, pages 1–15, 2002.

[17] K. Egli. Analysis of strategic behaviour in combined electricity and gas markets using agent-based computational economics. Master's thesis, Swiss Federal Institute of Technology Zurich, Switzerland, 2007.

[18] J. Eichberger, W. Güth, and W. Müller. Attitudes towards risk: An experiment. *Metroeconomica*, 54(1):89–124, 2003.

[19] K. Fischer, B. Chaib-draa, J. P. Müller, M. Pischeland, and C. Gerber. A simulation approach based on negotiation and cooperation between agents: A case study. *IEEE Transactions on Systems, Man, and Cybernetics*, 29(4):531–545, 1999.

[20] M. K. Ghosh, D. McDonald, and S. Sinha. Zero-sum stochastic games with partial information. *Journal of Optimization Theory and Applications*, 121(1):99–118, 2004.

[21] B. Grosz, S. Kraus, S. Talman, B. Stossel, and M. Havlin. The influence of social dependencies on decision-making: Initial investigations with a new game. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 780–787, 2004.

[22] C. Harries, J. S. Evans, and I. Dennis. Measuring doctors' self-insight into their treatment decisions. *Applied Cognitive Psychology*, 14:455–477, 2000.

[23] T. Ishida, Y. Nakajima, Y. Murakami, and H. Nakanishi. Augmented experiment: Participatory design with multiagent simulation. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, pages 1341–1346, 2007.

[24] G. Kempen, M. van Heuvelen, R. van den Brink, A. Kooijman, M. Klein, P. Houx, and J. Ormel. Factors affecting contrasting results between self-reported and performance-based levels of physical limitations. *Age and Ageing*, 25(6):458–464, 1996.

[25] R. Lin, N. Agmon, S. Kraus, S. Barrett, and P. Stone. Comparing agents' success against people in security domains. In *AAAI Conference on Artificial Intelligence*, pages 809–814, 2011.

[26] R. Lin, S. Kraus, Y. Oshrat, and Y. K. Gal. Facilitating the evaluation of automated negotiators using peer designed agents. In *AAAI Conference on Artificial Intelligence*, pages 817–822, 2010.

[27] R. Lin, Y. Oshrat, and S. Kraus. Automated agents that proficiently negotiate with people: Can we keep people out of the evaluation loop? In T. Ito, M. Zhang, V. Robu, S. Fatima, and T. Matsuo, editors, *New Trends in Agent-based Complex Automated Negotiations*, volume 383 of *Studies in Computational Intelligence*, pages 57–80. Springer-Verlag, 2012.

[28] M. A. Majid, P.-O. Siebers, and U. Aickelin. Modelling reactive and proactive behaviour in simulation. In *Operational Research Society Simulation Workshop*, pages 23–31, 2010.

[29] L. Mao. Agent-based simulation for weekend-extension strategies to mitigate influenza outbreaks. *BMC Public Health*, 11(522):1–10, 2011.

[30] D. Massaguer, V. Balasubramanian, S. Mehrotra, and N. Venkatasubramanian. Multi-agent simulation of disaster response. In *First International*

Workshop on Agent Technology for Disaster Management, pages 124–130, 2006.

[31] D. Massaguer, V. Balasubramanian, S. Mehrotra, and N. Venkatasubramanian. Multi-agent simulation of disaster response. In *First International Workshop on Agent Technology for Disaster Management*, 2006.

[32] N. Minar, R. Burkhart, C. Langton, and M. Askenazi. The swarm simulation system: A toolkit for building multi-agent simulations. Working Papers 96-06-042, Santa Fe Institute, 1996.

[33] S. R. Musse and D. Thalmann. Hierarchical model for real time simulation of virtual human crowds. *IEEE Transactions on Visualization and Computer Graphics*, 7(2):152–164, 2001.

[34] M. North, E. Tatara, N. Collier, and J. Ozik. Visual agent-based model development with repast simphony. In *Procededings of the Agent 2007 Conference on Complex Interaction and Social Emergence*, pages 173–192, 2007.

[35] M. Nygren. Simulation of human behaviour in stressful crowd situations. Master's thesis, School of Computer Science and Engineering, Royal Institute of Technology, Sweden, 2007.

[36] T. Offerman, J. Potters, and H. Verbon. Cooperation in an overlapping generations experiment. *Games and Economic Behavior*, 36(2):264–275, 2001.

[37] S. Ribaric and T. Hrkac. Object-oriented simulator of multi-agent system for temporally rich domains. In *Proceedings of the 25th International Conference on Information Technology Interfaces*, pages 397–402, 2003.

[38] A. Rosenfeld and S. Kraus. Modeling agents based on aspiration adaptation theory. *Autonomous Agents and Multi-Agent Systems*, 24(2):221–254, 2012.

[39] R. Selten, K. Abbink, J. Buchta, and A. Sadrieh. How to play (3 x 3)-games.: A strategy method experiment. *Games and Economic Behavior*, 45(1):19–37, 2003.

[40] W. Shao and D. Terzopoulos. Autonomous pedestrians. *Graphical Models*, 69(5–6):246–274, 2007.

[41] S. Tadokoro, H. Kitano, T. Takahashi, I. Noda, H. Matsubara, A. Shinjoh, T. Koto, I. Takeuchi, H. Takahashi, F. Matsuno, M. Hatayama, J. Nobe, and S. Shimada. The robocup-rescue project: A robotic approach to the disaster mitigation problem. In *IEEE International Conference on Robotics and Automation*, volume 4, 2000.

[42] T. Takahashi, S. Tadokoro, M. Ohta, and N. Ito. Agent based approach in disaster rescue simulation - from test-bed of multiagent system to practical application. In *RoboCup 2001: Robot Soccer World Cup V*, pages 102–111, 2002.

[43] D. Terzopoulos, X. Tu, and R. Grzeszczuk. Artificial fishes: Autonomous locomotion, perception, behavior, and learning in a simulated physical world. *Artificial Life*, 1(4):327–351, 1994.

[44] E. O. Thorp. *Beat the Dealer: A Winning Strategy for the Game of Twenty-One*. Random House Publishing, 1962.

[45] B. Ulicny and D. Thalmann. Towards interactive real-time crowd behavior simulation. *Computer Graphics Forum*, 21(4):767–775, 2002.

[46] O. Vaněk, M. Jakob, O. Hrstka, and M. Pěchouček. Using multi-agent simulation to improve the security of maritime transit. In *Proceedings of 12th International Workshop on Multi-Agent-Based Simulation*, 2011.

[47] M. L. Weitzman. Optimal search for the best alternative. *Econometrica*, 47(3):641–54, 1979.

[48] W. Young and W. Yue. A study of the preformance of two parking-lot layouts. *Traffic, Engineering and Control*, 33:434–439, 1992.

[49] Y. Zhang, K. Biggers, L. He, S. Reddy, D. Sepulvado, J. Yen, and T. Ioerger. A distributed intelligent agent architecture for simulating aggregate-level behavior and interactions on the battlefield. In *The Fifth World Multiconference on Systemics, Cybernetics and Informatics*, pages 58–63, 2001.