Can Agent Development Affect Developer's Strategy?

Avshalom Elmalech
Department of Computer Science
Bar-Ilan University, Israel
elmalea@cs.biu.ac.il

David Sarne
Department of Computer Science
Bar-Ilan University, Israel
sarned@cs.biu.ac.il

Noa Agmon
Department of Computer Science
Bar-Ilan University, Israel
agmon@cs.biu.ac.il

Abstract

Peer Designed Agents (PDAs), computer agents developed by non-experts, is an emerging technology, widely advocated in recent literature for the purpose of replacing people in simulations and investigating human behavior. Its main premise is that strategies programmed into these agents reliably reflect, to some extent, the behavior used by their programmers in real life. In this paper we show that PDA development has an important side effect that has not been addressed to date — the process that merely attempts to capture one's strategy is also likely to affect the developer's strategy. The phenomenon is demonstrated experimentally, using several performance measures. This result has many implications concerning the appropriate design of PDA-based simulations, and the validity of using PDAs for studying individual decision making. Furthermore, we obtain that PDA development actually improved the developer's strategy according to all performance measures. Therefore, PDA development can be suggested as a means for improving people's problem solving skills.

Introduction

Peer-designed agent (PDA) technology has been gaining much interest in recent years, mostly due to its potential of reducing much of the complexities and overheads of using people in laboratory experiments (Lin et al. 2010). Unlike expert-designed agents, PDAs are developed by non-domain experts, where the goal is to exhibit human-like rather than optimal behavior. As such, PDA technology has been increasingly used in recent years to replace people in system evaluation (Rosenfeld and Kraus 2012) in various domain such as negotiation (Lin et al. 2010), costly information gathering (Elmalech and Sarne 2012), security systems (Lin et al. 2011), parking allocation (Chalamish, Sarne, and Lin 2012) and training (Lin et al. 2013). Another common use of PDAs is in studying individual decision making (Grosz et al. 2004). The main premise in all these works is that the developed PDAs adequately represent the strategy of their developers.

The effectiveness of using PDAs as human behavior generators depends primarily on the similarity between the behaviors exhibited by PDAs and their developers. Nevertheless, despite the great interest in this technology, the applicability of PDA technology was evaluated, to date, mostly through measuring the similarity between the behaviors exhibited by PDAs and their developers, either at the macro level, i.e., comparing the collective or "average" behavior (Lin et al. 2010; Azaria et al. 2014), or at the micro level, i.e., comparing individual behaviors in similar decision situations (Chalamish, Sarne, and Lin 2012; Elmalech and Sarne 2012). No prior research, to the best of our knowledge, has attempted to investigate whether developers' strategy undergoes some kind of transformation along the process. This aspect is, however, of great importance, since if indeed the process of developing a PDA has some effect on developers' strategies, then much caution should be taken when using this technology. In particular, one needs to keep in mind that the change in the developers strategies precludes the consideration of the resulting set of strategies as a reliable representative sample of the general population's strategies. Therefore, even if PDAs reliably represent their developers, the results obtained by using them apply to a population which is somehow different than the original one, due to the strategy transformation this group has undergone when developing the PDAs.

In this paper we attempt to test whether indeed the development of a PDA changes one's strategy using the classic "Doors game" (Shin and Ariely 2004). For this purpose, we present the experimental design and report the results of an experiment comparing people's strategies before and after developing PDAs, and the strategies used by the PDAs they developed.

The analysis of the results suggests that indeed people's strategies change during the development of a PDA. Furthermore, we show that the change happens while developing the PDA rather than after, and that the change is favorable. This latter finding is based both on an increase in the average score achieved, as well as in several additional measures demonstrating the effectiveness of the strategy in games played by the participants after, compared to prior to, the development of the PDAs. This result has an important implication concerning the possible use of PDA technology as a means of improving people's problem solving skills.

Developing a PDA requires several skills. In addition to the actual programming, the developer needs to be able to express her strategy in a programmable manner. In order to reason about the contribution of the expressive part to the change in strategy, we report the results of a complementary experiment. This latter experiment follows the same methodology, however instead of requesting that people develop a PDA prior to playing the game they were requested to express their strategy in free text. The results of this experiment rule out any effect of the expressive part over people's strategies, according to all measures used. Therefore, the effect of PDAs programming over people's strategy is attributed to the PDA development process as a whole.

The Doors Game

The doors game was initially introduced by (Shin and Ariely 2004), and is a variant of the famous exploration versus exploitation problem. In the basic version of the game, a player is faced with three doors (alternatives), each associated with a different distribution of payoffs. The payoff distribution of each door is a-priori unknown to the player. The player first clicks (i.e., chooses) a door to begin with, 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 payoff 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. The player's goal is to maximize gains given a limited budget of clicks. Once the player has used all of her allowed clicks, the game terminates and she is paid the sum of her door-click payoffs. The single click that the player needs to sacrifice to switch doors is in fact a switching cost. This setting can be trivially mapped to the Multi-Armed Bandit problem (Auer et al. 1995).

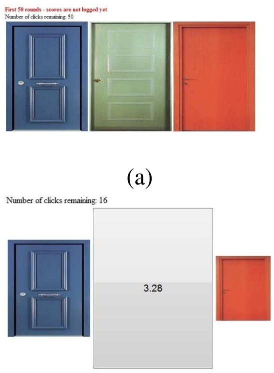
With the above game, human subjects have been found to be sufficiently efficient in the sense that they choose to engage in "door exploration" in the first few clicks and then stick with the door associated with the best expected yield (Shin and Ariely 2004). Nevertheless, for a specific variant of this game it has been found that people's strategies are highly inefficient (Shin and Ariely 2004). This specific game variant is identical to the original game, except that unless clicked at current round, the door size is continuously reduced, until it eventually vanishes. If the door is clicked before vanishing, it returns to its original size. For a setting where doors are reduced in size by $\frac{1}{15}$ of their original width, it has been found that players tend to switch from door to door, in an effort to keep their options open. This results in a substantial performance degradation (in terms of the rewards accumulated) compared to sticking with the best yielding door. Therefore, for our "doors game" experiment, we used this latter variant to test the extent to which people's inherent tendency of keeping all options viable (even when the cost of doing so is greater than the potential benefit) is affected by PDA development.

The reason for choosing the doors game for demonstrating the effect is due to the simplicity of the game. The game does not require advanced computational capabilities which people are lacking, hence we expect people to develop a PDA which its strategy will not be different than their own. Also, due to the simplicity of the game, it is easy for participants to understand the rules of the game and to come up with a legitimate strategy – rather than a random strategy which is observed in games where it is difficult to understand the rules of the game.

Experimental Design

In this section we describe the experimental design applied in our experiments, and specify the measures used. We implemented the doors game in a way that it could be played either using an interactive GUI client or through the use of a PDA. For the PDA development task, we followed the common practice from prior work (Sarne et al. 2011; Lin et al. 2010), i.e., we provided a skeleton of a functional PDA that lacked only its strategy layer. Strategy developers thus had to develop only the strategy component, using a rich API that was supported by the agent. Participants recruited for the experiments were all senior undergraduate or graduate computer science students, and were requested to develop a PDA that will play doors game on their behalf. Each participant received thorough instructions on the game rules, her goal in the game and the compensation method, which essentially was linear in her score in the game. This was followed by taking part in several practice games. Participants had to practice until stating that they understood the game rules and they had a good sense of what their game strategy was like. At this point participants were randomly divided into two groups. Participants of the first group (31 students) were requested to play a single instance of the game after the training stage. Participants of the second group (48 students) were requested to develop a PDA, and immediately after were requested to play an interactive instance of the game. While developing PDAs, the participants did not have the infrastructure to test how well it performs. Results were analyzed based on different measures as described below. In addition to analyzing the behavior of participants in the game played, we also measured the performance of the PDAs developed when used with the same doors game setting.

Our experiment with the doors game followed a specific experimental design reported in (Shin and Ariely 2004), where the game includes two phases, each with a "budget" of 50 clicks. In the first phase (the exploration phase), the participants did 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 an optimal 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. Following the original experimental design, we used the three different distribution functions specified in (Shin and Ariely 2004), all with a mean payoff of 6. The payoff distribution of the first door was highly concentrated around the mean (normal distribution with a variance of 2.25); the second door also had values around the mean but the values were much more diffused (normal distribution with a vari-



(a)

(b)

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

ance of 9); and the payoff distribution of the third door was highly skewed toward high numbers (chi square distribution with 8 degrees of freedom). The minimal and maximal values of the three distributions were -4 and 19, respectively.

In the GUI version of the game the system presented the doors to the students, with each door size changing according to the clicks made. Figure 1a is a screen-shot of a game where the user has not yet clicked on any door. Figure 1b is a screen-shot taken in the middle of the game, where the user has picked the middle door and the other two doors have begun to shrink. The subjects were told that their reward will be proportional with the total amount of payoffs they receive in the game (i.e., those accumulated over the last 50 rounds).

Following (Shin and Ariely 2004) we used three measures to evaluate the effectiveness of the strategies used. The first measure, denoted "outcome performance", is the average score of participants in the game (in the last 50 rounds). Since the three doors in the game had the same expected payoff of 6, the average outcome performance of the optimal strategy is 6. The second measure, denoted "number of switches", is the average number of time players switched doors in the last 50 rounds. As explained above, an effective strategy in this game should result in an insignificant number of switches at this stage, since all "explorations" should take place during the initial 50 rounds of the game. A high value of this measure leads to poor results in the game. The third measure, denoted "elimination point", is the average turn at which participants stopped switching between doors in the second stage of the game (last 50 rounds). For reasons similar to those given for the "number of switches" measure, an effective strategy is typically characterized with a low value for the "elimination point" measure (the closer to zero, the better).

Results and Analysis

In this section we report the results of comparing the performance of PDA developers without and post development of a PDA. Statistical significance was tested using the student's t-test (two-sample assuming unequal variances). The results are primarily reported as the group's average since the game is of a probabilistic nature and there is only one result for each participant.

Figure 2 depicts the average outcome performance of the group of students who played the game without developing a PDA (denoted "no-PDA"), the group of PDAs themselves (denoted "PDAs") and the group of students who played the game after developing PDAs (denoted "post-PDA"). As demonstrated in the figure, there is a substantial difference between the average performance of the no-PDA and the post-PDA groups. The difference is statistically significant ($p - value < 0.001$), indicating that indeed different strategies were used. In particular, it is apparent that the outcome performance measure of the post-PDA group substantially improved. Since the outcome performance when playing this game is bounded by 6 (as explained earlier), the inefficiency improvement between no-PDA and the post-PDA is 64%.[1] The PDAs score according to the outcome performance measure is similar to the performance of the post-PDA group (statistically indifferent), suggesting that the change in the PDAs developers' strategies occurred while developing the PDAs and not after. The difference between PDAs and the no-PDA group is statistically significant ($p{-}value < 0.003$), indicating that the PDAs use strategies different than those of the no-PDA group, hence they cannot be used as a reliable representation of the latter in this specific domain.

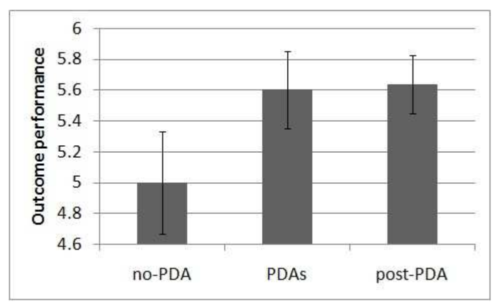


Figure 2: Comparison of the outcome performance.

Figures 3 and 4 depict the difference in the average number of switches and the average elimination point between the three groups (no-PDA, PDAs and post-PDA), respectively. The results are consistent with those found for the outcome performance measure: the difference between the average performance of the no-PDA and the post-PDA groups is substantial and statistically significant ($p{-}value < 0.001$ for both), indicating that indeed different strategies were used. In both cases the differences suggest an improvement in the measure in the post-PDA group compared to the no-PDA group. The inefficiency improvement between no-PDA and the post-PDAs is 69% for the average number of switches measure, and 60% for the average elimination point measure (where the theoretical bound was considered

[1]The inefficiency improvement measures the decrease (in percentages) in the difference between the average result achieved and the theoretical bound (6 in the case of outcome performance), as the difference between the two represents the strategy's inefficiency.

to be 0 for both measures). The PDAs score according to the two measures was similar to the performance of the post-PDA group (no statistical difference), and different from the performance of the no-PDA group ($p - value < 0.001$ for both), supporting, once again, the conclusions that the change in the PDA developers' strategies occurred prior to completing the PDA development and not after, and that the PDAs do not reliably represent the no-PDA group in this domain.

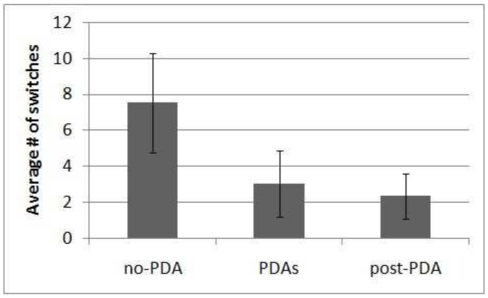


Figure 3: Comparison of the average number of switches.

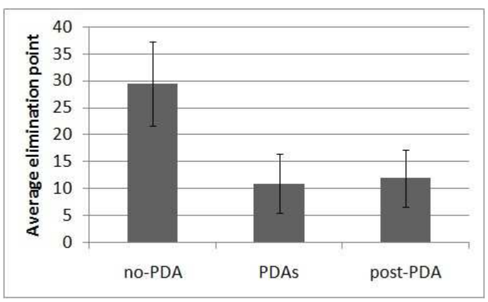


Figure 4: Comparison of the average elimination point.

A drill down analysis of the individual performance measure value sheds some light on the nature of the change and its trend in people's strategies due to the development of a PDA in our experiments. Figure 5 depicts the distribution of the number of switches recorded for the different participants in the no-PDA and post-PDA populations. The grouping was done according to "optimal" strategy (0 switches), "close to optimal" (1-3 switches) and rather "inefficient" strategies (4-9, 10-19, and 20 and above). As can be observed in the figure, the process of PDA development resulted in a substantial shift in strategy according to this classification: 43% of the "inefficient' strategies changed to "optimal" and "close to optimal" (where the majority changed to "optimal"). This indicates that the strategy development does not have an equal effect on all PDA developers. While some of them kept their inefficient strategy, those that ended up revising their strategy shifted to a mostly efficient one. Overall, while in the no-PDA population 32% of the subjects used an optimal strategy, in the post-PDA group 65% of the subjects were found to use that strategy. Similar qualitative results were found using a drill-down analysis of the elimination point measure.[2]

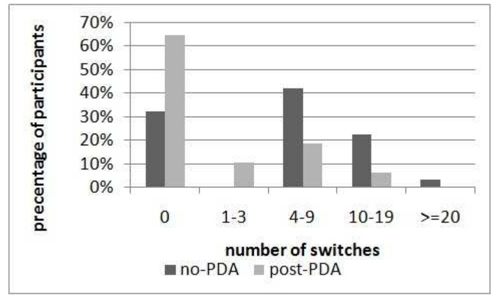


Figure 5: Drill-down comparison between the number of switches without and post development of PDAs.

Strategy Description

In this section we report the results of a complementary experiment which we conducted to identify the reason why the process of developing a PDA affects its developer's strategy. More specifically, to examine if the phenomenon is due to the descriptive nature of developing a PDA. For this experiment, however, since participants were not required to exhibit any programming skills, but merely to describe their strategy in a way that can be later programmed, we relied on participants recruited from Amazon Mechanical Turk (AMT) (AMT 2010)[3] and had them play the doors game. Overall, 100 people participated in this experiment, whereby 50 of them were asked to describe (i.e., express) their strategy prior to playing the game and the rest were asked merely to play the game. Both groups received detailed instructions and practiced the game prior to expressing their strategy or playing. The layout of the experiment and the experiment design used were similar to those reported in the foregoing section. Figure 6 depicts the performance of the two groups according to the three measures defined for the doors game. As depicted in the figure, the strategy expressing activity had no influence whatsoever on performance in all three measures (all differences are statistically insignificant). These results may indicate that the change in behavior reported in the previous section is not due to the descriptive nature of developing PDAs, but rather due to other characteristics of PDA development. [4]

Here, again we provide a drill-down analysis of the distribution of the number of switches in the individual level (Figure 7). As illustrated in Figure 7, the change in the number of strategies that can be considered "optimal", due to strategy expression, is marginal, whereas most change is observed between the different segments of the "inefficient"

[2]This kind of analysis for the outcome performance measure is futile, since this measure, when taken individually, highly depends on chance.

[3]For a comparison between AMT and other recruitment methods see (Paolacci, Chandler, and Ipeirotis 2010).

[4]We note that the differences observed could be because of the differences in populations (programmers vs. mostly non-programmers).

groups. The change pattern in the number of switches due to the strategy description process is fundamentally different than the one associated with PDA development reported in the foregoing section. While in the ”strategy description experiment” there is no shift in the strategies of the subjects, in the ”PDA experiment” the strategies of the subjects shifted from the “inefficient” group to the “optimal” group, supporting the conclusions drawn from Figure 6. Similar qualitative results were found using a drill-down analysis of the elimination point measure.

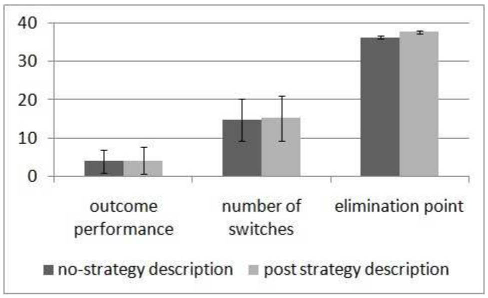


Figure 6: Comparison of the results of people without and post describing their strategy over the three measurements.

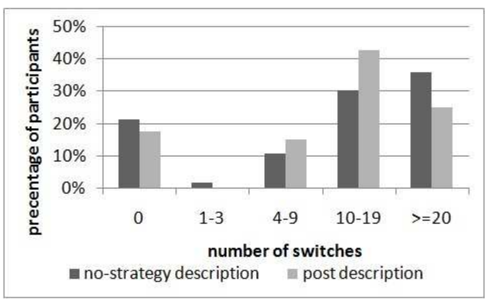


Figure 7: Drill-down comparison between the number of switches without and post strategy description.

Related Work

The use of agents in general for human behavior simulation is quite extensive in the AI literature (Zhang et al. 2001; Bosse et al. 2011; Chalamish, Sarne, and Lin 2012; Lin et al. 2010). Within this rich literature, two primary methodologies for simulating human behavior with agents can be identified: experts designed agents (EDAs) and agents developed by non-experts (PDAs).

EDAs typically are agents whose strategies are developed by the simulation designers. Over the years simulation designers have used various methods for setting human-like behavior in the agents they developed (EDAs). These include, among other methods, statistical-data based modeling (Takahashi et al. 2002), pre-specification of agents' roles using a set of parameters according to which the agents act (Massaguer et al. 2006), using pre-defined events and reactions (Musse and Thalmann 2001), defining a number of basic behaviors from which all of the agents' more complex behaviors are constructed (Shao and Terzopoulos 2007; Terzopoulos, Tu, and Grzeszczuk 1994) or using a combination of rules and finite state machines to control an agent's behavior using a layered approach (Ulicny and Thalmann 2002). The main advantages of using EDAs for simulation purposes (compared to recruiting people) are their capabilities to interact among themselves and scale. The main difficulty of this method is that the simulation designer and even domain experts are quite limited in the number of different individual behaviors they can generate and the time it takes them to develop them.

The success of designing efficient EDAs, i.e., ones that reliably simulate human behavior, is controversial. For example, it has been shown (Gode and Sunder 1993) that there is a resemblance between the transaction price path of agent traders designed using bounded rational theories and the transaction price path of human subjects in a double auction environment. However, other research claimed that these results do not hold once the value of one of the market parameters slightly changes (Van Boening and Wilcox 1996; Brewer et al. 2002).

PDAs technology has been widely used in recent years. For example, in Kasbah (Chavez and Maes 1996) PDAs that buy and sell were used for evaluating an electronic marketplace. In Colored Trails (Grosz et al. 2004), PDAs were used for reasoning about players' personalities in uncertain environments. Other works, e.g., (Lin et al. 2010; 2011; Chalamish, Sarne, and Lin 2012; Rosenfeld and Kraus 2012) used PDAs for evaluating specific mechanisms in various domains such as evaluating security algorithms and evaluating automated negotiators.

In some respects, the idea of using people to program agents as a means for achieving a reliable set of strategies for specific decision situations was inspired by the "strategy method" paradigm from behavioral economics (Selten, Mitzkewitz, and Uhlich 1997). In the strategy method people state their action for every possible situation that may arise in their interaction. The main difference between the strategy method and PDAs technology is that in the first participants need to *describe* their choices for each possible state, whereas with PDAs the requirement is to express a *cohesive formulation* of their strategy (Chalamish, Sarne, and Lin 2012; Rosenfeld and Kraus 2012; Cheng et al. 2011). This entails various implications related to the time it takes to capture one's strategy (an advantage for the PDAs in cases where the possible number of system's states is large and an advantage for the strategy method when the game is extremely simple, e.g., in the ultimatum game), the ability to understand one's strategy (an advantage for PDAs, as their code can be analyzed afterwards) and the ability to use the strategy when the setting slightly changes (impossible with the strategy method).

The main motivation for using PDAs in simulations is the premise that PDAs reliably represent their designers' strategies. This, however, is not straightforward. Evidence of discrepancies between actual and reported human behavior is a prevalent theme in research originating in various domains, in particular in metacognition research (Harries, Evans, and Dennis 2000). Examples of such discrepancies include over-reporting of political participation (Bertrand and Mullainathan 2001) and contrasting results between self-reported and performance-based levels of physical limitations (Kempen et al. 1996). Yet, much of the PDA literature tends to assume that people can successfully (to some extent) capture their real-life strategy in a given domain when programming an agent (Cheng et al. 2011; Rosenfeld and Kraus 2012). Even in cases where some discrepancy between PDAs and people's behavior is reported, the average performance is reported to be similar, suggesting that PDAs can replace people in mechanism evaluation (Lin et al. 2010; 2011). None of the above literature deals with the question of whether the PDA developing process itself affects the developer's strategy, which is the focus of this paper.

Finally, we note that work in psychology, computer science and education presented evidence that computer programming can be a powerful tool for improving thinking and for developing good problem-solving skills (Feurzeig, Papert, and Lawler 2011; Nickerson 1982; Jeffries et al. 1981; Clements and Gullo 1984). In addition, the programming process can be used for teaching students fundamental concepts in mathematics and logic. The main difference between these related works and the work presented in this paper is that while the first focus on the general effect of programming over the cognitive skills of the programmer in general, our work focuses on whether the process of developing an agent for a *specific* problem changes the developers strategy for solving that specific problem. In addition, in prior work the question of when the change in strategy occurs was not addressed.

Conclusions

Based on the results reported and their analysis, we conclude that indeed the development of a PDA affects and reshapes one's strategy in the domain used for our experiments. Since the effect reported in this work was demonstrated over one domain we do not claim that this effect occurs in all domains. The aim of this work was to demonstrate the existence of the effect rather than its magnitude as a function of the decision problem's characteristics. This important aspect of PDA technology, which has not been investigated to date, has many important implications. In particular, system designers (e.g., simulation designers) interested in using PDAs for generating human behaviors need to reveal the extent to which PDA development indeed changes individuals' strategies in their simulated domains. Based on the extent of the change found, they will need to tradeoff the loss incurred by the fact that the strategies embedded in the PDAs are not necessarily the ones that would have been used by their developers if they had not been requested to develop the PDAs, and the many benefits of using PDAs (such as reduced overhead, flexibility in the number of settings that can be tested and the ability to perform any experiment in a timely manner). This main result also contributes to PDA literature in the sense that it provides a possible explanation for discrepancies observed between the behaviors of PDAs and their developers.

PDAs' performance was found to be significantly different from the performance of those that played our games without developing a PDA, and insignificantly different from those playing after developing a PDA. This suggests that the change in developers' strategies occurs while working on their PDA and before completing it — by the time the PDA is complete, it is already equipped with the revised strategy. When requesting that participants express their strategy, rather than actually develop a PDA, no change in behavior was observed. This suggests that the effect of the development of PDAs goes beyond the need to express one's strategy, whereas the design and programming themselves account for the change.

One may argue that the reason for the change in the PDA developer's strategy is due to trial-and-error which occurs during a standard process of designing PDAs. This, however, is not the case. In our game, when programming the strategy the participants did not have the infrastructure to test how well it performs. Therefore the effect is not due to trial-and-error.

We observed substantial inefficiencies in the strategies used by the no-PDA population. Interestingly, the inefficiencies characterizing strategies of individuals in the doors games are primarily rooted in people's tendency to keep all their options available (Shin and Ariely 2004), which is a psychological effect. Through the development of PDAs a large portion of the population managed to overcome these inefficiencies. This suggests that PDA development can be used as a means of improving one's strategy in scenarios where the source of inefficiency is due to a psychological effect. In fact, we see much room for future research aiming at developing tools and methods for identifying domains and conditions where PDA development is strategy-improving. Furthermore, the benefit of PDA development should not be limited only to those who are capable of programming. Within this scope we suggest repeating our experiments with the general audience, using semi-programming tools (e.g., Scratch (Resnick et al. 2009)) and a bit more structured methods for expressing one's strategy to see if any of these benefits can be useful for the general population.

We note that despite the effect that the process of developing a PDA has on its developer, this technology is extremely useful in settings where such effect is tolerable. The use of PDAs in such cases can save substantial resources and facilitate the evaluation of numerous potential configurations, in a relatively short time, without having to recruit people over and over again for expensive experiments. In particular, this technology is useful for simulating and researching large-scale systems due to the relatively low cost of cloning agents.

Finally, we suggest further research which will be aimed at better explaining which aspect of PDA development is responsible for the change in its developer's strategy. While our complementary experiment ruled out the expressive aspect of the process as an explaining factor, we do believe that further research will shed some light on this interesting question.

Acknowledgement

We would like to thank Amos Azaria for his help. This work was partially supported by ISF grant 1083/13.

References

AMT. 2010. Amazon mechanical turk. *www.mturk.com*.

Auer, P.; Cesa-Bianchi, N.; Freund, Y.; and Schapire, R. E. 1995. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Foundations of Computer Science*, 322–331.

Azaria, A.; Richardson, A.; Elmalech, A.; and Rosenfeld, A. 2014. Automated agents' behavior in the trust-revenge game in comparison to other cultures. In *AAMAS*.

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

Bosse, T.; Gerritsen, C.; Hoogendoorn, M.; Jaffry, S.; and Treur, J. 2011. Agent-based vs. population-based simulation of displacement of crime: A comparative study. *WIAS* 9(2):147–160.

Brewer, P.; Huang, M.; Nelson, B.; and Plott, C. 2002. On the behavioral foundations of the law of supply and demand: Human convergence and robot randomness. *Experimental economics* 5(3):179–208.

Chalamish, M.; Sarne, D.; and Lin, R. 2012. Enhancing parking simulations using peer-designed agents. *ITS* 14(1):1–7.

Chavez, A., and Maes, P. 1996. Kasbah: An agent marketplace for buying and selling goods. In *Proc. of Intelligent Agents and Multi-Agent Technology*, 8–12.

Cheng, K.; Zuckerman, I.; Nau, D.; and Golbeck, J. 2011. The life game: Cognitive strategies for repeated stochastic games. In *Proc. of social computing (socialcom)*, 95–102.

Clements, D. H., and Gullo, D. F. 1984. Effects of computer programming on young children's cognition. *Journal of Educational Psychology* 76(6):1051.

Elmalech, A., and Sarne, D. 2012. Evaluating the applicability of peer-designed agents in mechanisms evaluation. In *Proc. of IAT*, 374–381.

Feurzeig, W.; Papert, S. A.; and Lawler, B. 2011. Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments* 19(5):487–501.

Gode, D., and Sunder, S. 1993. Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. *Journal of political economy* 101:119–137.

Grosz, B.; Kraus, S.; Talman, S.; Stossel, B.; and Havlin, M. 2004. The influence of social dependencies on decision-making: Initial investigations with a new game. In *Proc. of AAMAS*, 782–789.

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

Jeffries, R.; Turner, A. A.; Polson, P. G.; and Atwood, M. E. 1981. The processes involved in designing software. *Cognitive skills and their acquisition* 255–283.

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

Lin, R.; Kraus, S.; Oshrat, Y.; and Gal, Y. 2010. Facilitating the evaluation of automated negotiators using peer designed agents. In *Proc. of AAAI*, 817–822.

Lin, R.; Kraus, S.; Agmon, N.; Barrett, S.; and Stone, P. 2011. Comparing agents' success against people in security domains. In *Proc. of AAAI*.

Lin, R.; Gal, Y.; Kraus, S.; and Mazliah, Y. 2013. Training with automated agents improves people's behavior in negotiation and coordination tasks. *DSS* 60:1–9.

Massaguer, D.; Balasubramanian, V.; Mehrotra, S.; and Venkatasubramanian, N. 2006. Multi-agent simulation of disaster response. In *Proc. of Agent Technology for Disaster Management*, 124–130.

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

Nickerson, R. 1982. Computer programming as a vehicle for teaching thinking skills. *Thinking: The Journal of Philosophy for Children* 4(3/4):42–48.

Paolacci, G.; Chandler, J.; and Ipeirotis, P. 2010. Running experiments on amazon mechanical turk. *Judgment and Decision Making* 5(5):411–419.

Resnick, M.; Maloney, J.; Rosenbaum, E.; Silver, J.; Silverman, B.; et al. 2009. Scratch: programming for all. *Communications of the ACM* 52(11):60–67.

Rosenfeld, A., and Kraus, S. 2012. Modeling agents based on aspiration adaptation theory. *AAMAS* 24(2):221–254.

Sarne, D.; Elmalech, A.; Grosz, B. J.; and Geva, M. 2011. Less is more: restructuring decisions to improve agent search. In *Proc. of AAMAS*, 431–438.

Selten, R.; Mitzkewitz, M.; and Uhlich, G. 1997. Duopoly strategies programmed by experienced players. *Econometrica* 65(3):517–555.

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

Shin, J., and Ariely, D. 2004. Keeping doors open: The effect of unavailability on incentives to keep options viable. *Management Science* 50(5):575–586.

Takahashi, T.; Tadokoro, S.; Ohta, M.; and Ito, N. 2002. Agent based approach in disaster rescue simulation - from test-bed of multiagent system to practical application. In *Proc. of RoboCup*, 102–111.

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

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

Van Boening, M., and Wilcox, N. 1996. Avoidable cost: Ride a double auction roller coaster. *The American Economic Review* 86:461–477.

Zhang, Y.; Biggers, K.; He, L.; Reddy, S.; Sepulvado, D.; Yen, J.; and Ioerger, T. 2001. A distributed intelligent agent architecture for simulating aggregate-level behavior and interactions on the battlefield. In *Proc. of SCI*, 58–63.