Noname manuscript No.
(will be inserted by the editor)

Agent Development as a Strategy Shaper

Avshalom Elmalech · David Sarne · Noa Agmon

Received: date / Accepted: date

Abstract This paper studies to what extent agent development changes one's own strategy. While this question has many general implications, it is of special interest to the study of Peer Designed Agents (PDAs), which are computer agents developed by non-experts. This latter emerging technology has been 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. 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. This result has many implications concerning the appropriate design of PDA-based simulations as well as the validity of using PDAs for studying individual decision-making. The phenomenon is demonstrated experimentally, using two very different application domains and several performance measures. Our findings suggest that the effect over one's strategy arise both in situations where it is potentially possible for people to reason about the optimal strategy (in which case PDA development will enhance the use of an optimal strategy) and in those where calculating the optimal strategy is computationally challenging (in which case PDA development will push people to use more effective strategies, on average). Since, in our experiments, PDA de-

· Avshalom Elmalech
Department of Computer Science
Bar-Ilan University, Israel.
E-mail: elmalea@cs.biu.ac.il
· David Sarne
Department of Computer Science
Bar-Ilan University, Israel.
E-mail: sarned@cs.biu.ac.il
· Noa Agmon
Department of Computer Science
Bar-Ilan University, Israel.
E-mail: agmon@cs.biu.ac.il

velopment actually improved the developer's strategy, PDA development can be suggested as a means for improving people's problem solving skills. Finally, we show that the improvement achieved in people's strategies through agent development is not attributed to the expressive aspect of agent development per-se but rather there is a crucial additional gain in the process of designing and programming ones strategy into an agent.

Keywords PDAs · decision-making · simulation design

1 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 overhead of using people in laboratory experiments [31]. Unlike expert-designed agents, PDAs are developed by non-domain experts, where the goal is to exhibit human-like rather than optimal behavior in a given domain. As such, PDA technology has been suggested in recent years to replace people in system evaluation [38] in various domains such as negotiation [31], costly information gathering [20], security systems [30] and parking allocation [16]. Another common use of PDAs is in studying individual decision-making [26]. 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 [31,7], or at the micro level, i.e., comparing individual behaviors in similar decision situations [16,20]. No prior research, to the best of our knowledge, has attempted to investigate whether a developer's 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 while developing the PDAs.

In this paper we attempt to test whether the development of a PDA does in fact change one's strategy, and if so then to what extent and under what conditions. For this purpose we use an experimental infrastructure that relies on two games from very different application domains. The first is the classic "doors game" [41] and the second is the "penetration detection game" [3].

The fundamental difference between the two, other than belonging to different application domains, is that in the first the optimal solution to the problem is "reachable" to people (i.e., can potentially be calculated) whereas in the second it is computationally infeasible to people. Other differences include repetitiveness (the doors game is a multi-stage recurring game, whereas the penetration detection game is a one-shot game) and the nature of the reward (average/overall payoff in the "doors game" in comparison to a binary payoff in the penetration detection game).

In the following two sections we describe the "doors game", which was our primary testbed, and the general experimental design which was used also for the penetration detection game. The analysis of the results, as given in Section 4, suggests that people's strategies indeed 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.

Sections 5 and 6 provide a set of additional experiments aiming to reason about the origins of the change observed in people's strategies due to PDA programming. In particular, we attempt to find out if it is the expressive nature of PDA programming or the fact that it enables people to think their strategy through that leads to the change. 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 that follows the same methodology as above, however instead of requesting that people develop a PDA prior to playing the "doors game", they were requested to express their strategy in free text. The results of this experiment, that were obtained both with a population of programmers and a population of non-programmers, 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. Furthermore, the comparison of users' strategy descriptions and the way they actually played ruled out the potential hypothesis that the lack of improvement in case of strategy description is due to the failure of people to follow their description throughout the game, rather people's suboptimal strategies.

Section 7 present our experiments and results with the penetration detection game. The results complement and align with those achieved with the "doors game" and their analysis leads to the same conclusions, i.e., people's strategies change during the development of a PDA. The unique nature of the game suggests that agent development affects people's strategy even in domains where the optimal solution cannot be calculated by people, and only an approximation of the optimal solution can be calculated by them. A thorough review of related work is given in Section 8. Finally, we conclude with a discussion and directions for future work.

2 The Doors Game

The "doors game" was initially introduced by Shin and Ariely [41], 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 (for a schematic illustration of the game, see Figure 1). 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 [6].

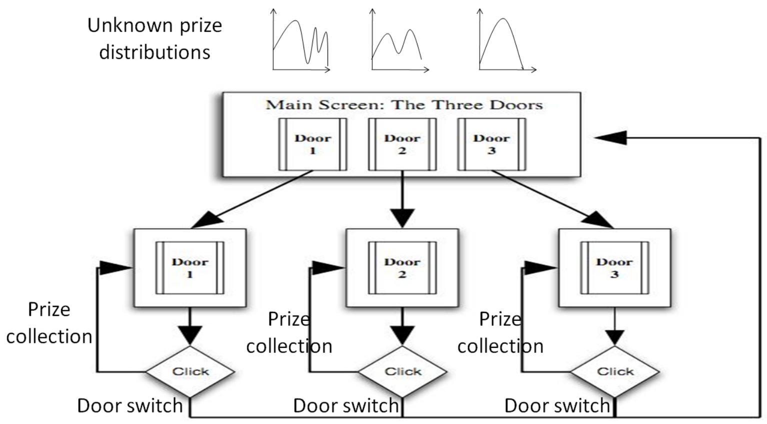

Fig. 1 An illustration of the "doors game".

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 [41]. Nevertheless, for a specific variant of this game it has been found that people's strategies are highly inefficient [41]. This specific game variant is identical to the original game, except that unless it is clicked in the 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 experiments, 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.

Overall, the "doors game" variant described above offers many advantages when used as an infrastructure in our experiments for testing whether one's strategy is affected by PDA development. The game is quite simple, does not require advanced computational capabilities which people lack, hence we expect people to develop a PDA whose 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 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.

3 Experimental Design

In this section we describe the experimental design applied in our experiments, and specify the measures used. We implemented the "doors game" such 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 [23,31], 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 and graduate computer science students, all taking the same course, and were requested to develop a PDA that would play the "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 until they had a good sense of what their game strategy was like. At this point participants were randomly divided into two groups. Participants in the first group (31 students) were requested to play a single instance of the game after the training stage. Participants in the second group (48 students) were requested to develop a PDA, and upon completing their PDA were requested to play an interactive instance of the game. We emphasize that the experimental design is inherently constrained by the need to allow people some substantial time to thoroughly think about their strategy (either playing the game or for PDA development), hence fully monitoring them along the process is impractical. In that sense, one measure of precaution taken was instructing participants not to discuss their strategies with others throughout that stage. To enforce this requirement, a three-tier enforcement system was used: First, we made the students aware of our intention to check the PDAs' code with MOSS, which is an automated system that checks similarity in code [1]. Second, we asked the course TA, who was subscribed to both the formal and informal forums of this course, to tap any conversation related to the experiment. Third, we asked three members of our research group who were enrolled in this course (and naturally did not take part in the experiment) to report to us in case they found out that any of their classmates shared information related to this experiment. 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 experiments with the "doors game" followed a specific experimental design reported in [41], 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 on which to keep clicking. This phase is long enough to enable a 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 actually 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 [41], 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 variance of 9); and the payoff distribution of the third door was highly skewed toward high numbers (chi square distribution with 8 degrees of freedom, shifted by 2 to the left in order to enable the same average as the other two distributions). The minimal and maximal values of the three distributions were -2 and 19, respectively. Figure 2 illustrates the three distributions used in our experiments.

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 3 is a screen-shot of a game where the user has not yet clicked on any door. Figure 4 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 already begun to shrink. The participants were told that their reward will be proportional to the total amount of payoffs they receive in the game (i.e., those accumulated over the 50 rounds of the second phase of the experiment).

Following [41] 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

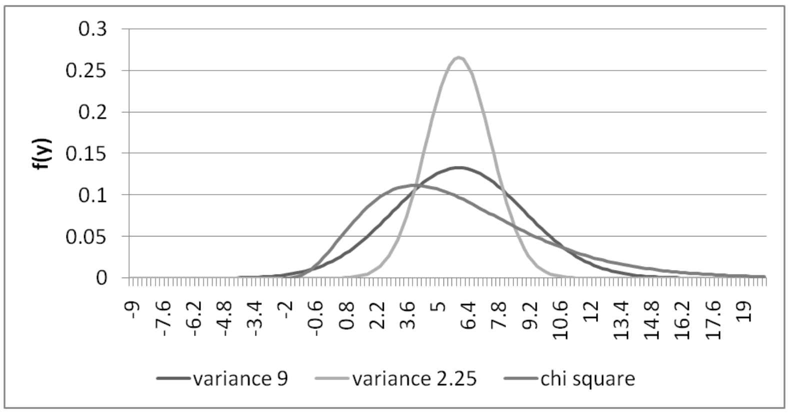

Fig. 2 An illustration of the three distributions used in the "doors game" experiments.

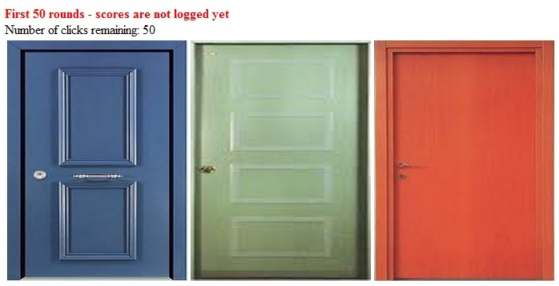
First 50 rounds - scores are not logged yet
Number of clicks remaining: 50

Fig. 3 A screen-shot of the "doors game" user interface before the user clicks on a door.

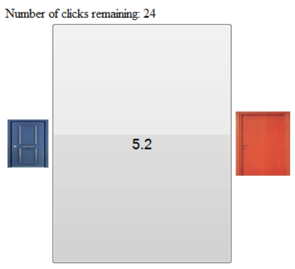
Number of clicks remaining: 24

Fig. 4 A screen-shot of the "doors game" user interface in the middle of the game after the user clicked on the middle door.

performance of the optimal strategy is 6. The second measure, denoted "number of switches", is the average number of times 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).

4 Results and Analysis

In this section we report the results from comparing the performance of PDA developers with no PDA 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 5 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 was 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.

Figures 6 and 7 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

[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.

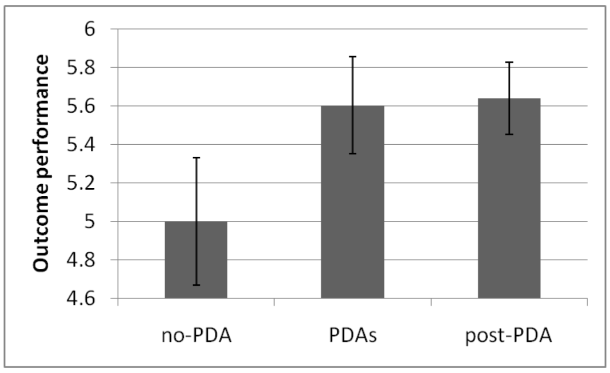

Fig. 5 Comparison of the outcome performance.

(where the theoretical bound was considered 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). Once again, this supports 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.

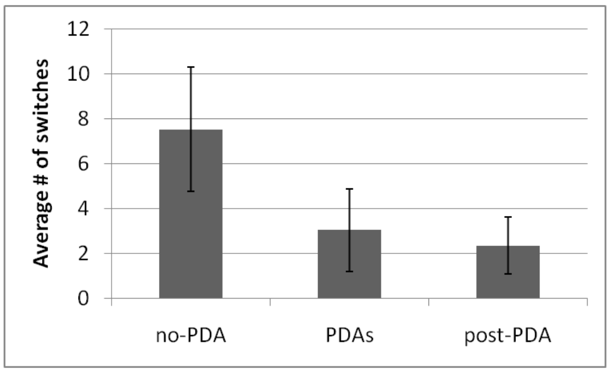

Fig. 6 Comparison of the average number of switches.

An in depth 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 8 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

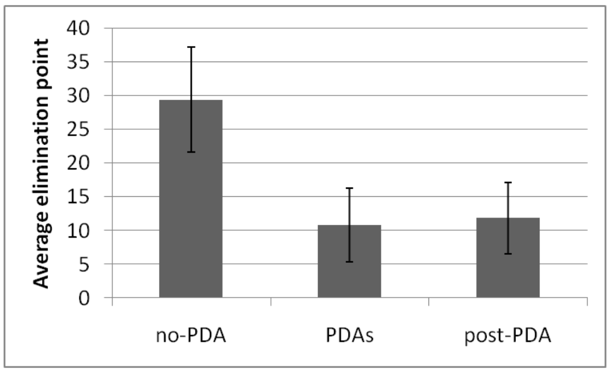

Fig. 7 Comparison of the average elimination point.

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 an in depth analysis of the elimination point measure.[2]

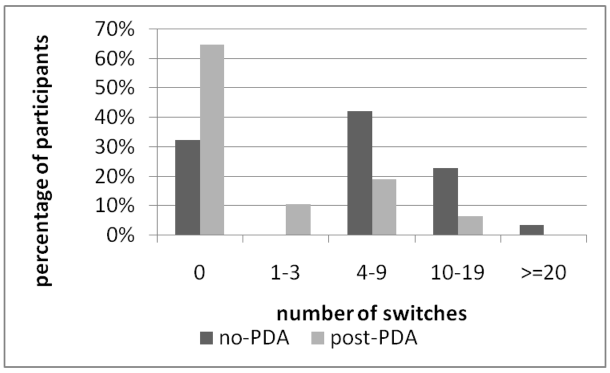

Fig. 8 In depth comparison between the number of switches without PDA and post development of PDAs.

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

5 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 we recruited a new group of 32 computer science students and had them play the "doors game". In the first stage the students received a detailed explanation of the game and were asked to write down their strategy. The students were requested to send us their strategy seven days after, and then, after sending, to play the GUI version of the game[3]. The layout of the experiment and the experiment design used were similar to those reported in the previous section (including the enforcement mechanism for insuring that students will not share their strategy with classmates). In particular, we used the same enforcement system described in Section 3 in order to make sure that participants are not discussing their strategies among themselves (except of course for using the MOSS system which is irrelevant to strategy description). Figure 9 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. Here again we provide an in depth analysis of the distribution of the number of switches at the individual level (Figure 10). As illustrated in Figure 10, most of the change is within the number of strategies that can be considered "optimal", and the change between the different segments of the "inefficient" part of the graph is marginal. 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 former 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 9. Similar qualitative results were found using an in depth analysis of the elimination point measure.

The absence of any performance improvement when users play the game after describing their strategies is surprising. The explanation for this, as advocated in the paper, is probably because writing down one's strategy is not sufficient enough to generate a change in the way this person solves a problem. However, the deeper process of designing an agent (which presumably involves a different kind of thinking) may influence the way a person solves a problem. In order to thoroughly understand this phenomenon, we reviewed the log file of users strategy descriptions. For each user we validated that the strategy de-

[3] The idea was to equalize conditions, so that people would have the chance to thoroughly think through their stated strategy, much like how PDA development takes some time.

scribed is correlated with the way she played the game (we note that looking at ones strategy description and comparing it with the set of actions reflected in the game played is a bit problematic, but this is the best that can be done in order to check if users played according to their specified strategy description). Most of the users (75%) played according to their strategy description. Furthermore, only 20% of the users overall described an optimal strategy for the game (and all of them played accordingly). We conclude that the reason for the lack of improvement in the case of strategy description is not due to the failure of people to follow their description throughout the game, rather people's suboptimal strategies.

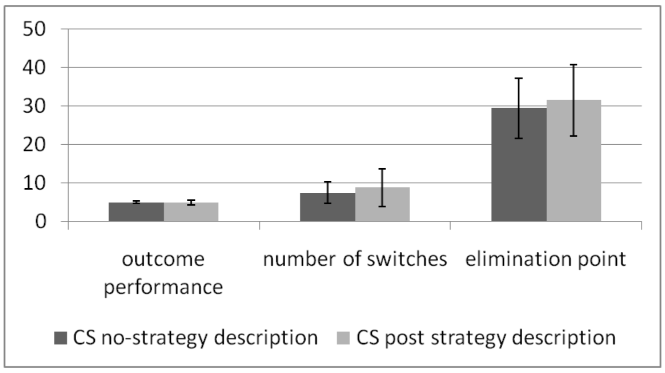

Fig. 9 Comparison of the results of CS students without and post strategy description, over the three measurements.

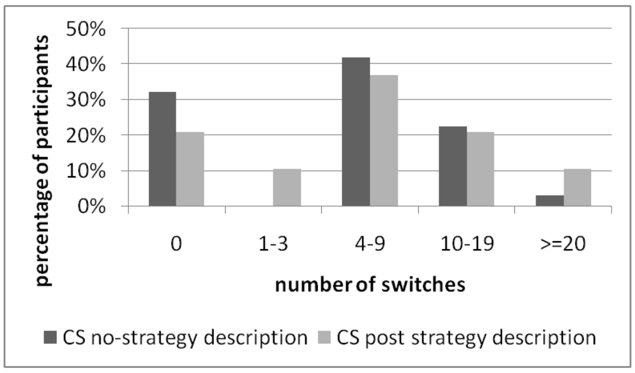

Fig. 10 An in depth comparison between the number of switches without and post strategy description.

6 Strategy Description by the General Population

In the previous section we compared the strategies of CS students playing the game without describing their strategies with the strategies of CS students playing the game after describing their strategies. The conclusion was that the effect of PDA design on the designers' strategies is not due to the descriptive aspect of programming and that strategy description by itself does not change the strategies of *CS students*. In this section we test again this aspect, this time however with the general population, i.e., individuals without programming skills. The idea is that it is possible that programmers, unlike the general population, do not pay much attention to strategy description and do not get to thoroughly think about a strategy in a structured manner until they actually need to program it. People who are not familiar with programming, on the other hand, may put a greater emphasis on strategy description and potentially benefit from the process. If this is indeed the case, then the implications, in the form of using strategy description as a tool for improving people's decision-making, certainly warrant the additional experimentation.

For this purpose we recruited participants from Amazon Mechanical Turk (AMT) [5][4] and had them play the "doors game". While the use of participants recruited through AMT as a means for representing the behavior of the general population is quite extensive and common in literature of our kind [16, 38, 23, 31], there is no actual guarantee that the recruited population is a perfect representation of the general population. Still, it provides a rich sample of the population, with participants of different countries, educational background and of a wide age range, which is difficult to recruit otherwise unless one has no time and budget constraints. Our AMT participants were within the 20 to 65 age range, 54% of them were men, and the population greatly varied in their education (high school education, college students, higher education degrees). 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 experimental design used were similar to those reported in the former section. The only difference is that in the AMT experiments we asked the participants about their programming skills and we filtered out those who reported having some. The payment to the participants was not correlated with their background knowledge in programming – both types of participant received money for taking part in the experiment. The only difference between the groups was that we did not use the data of the participants that reported having programming skills. Therefore participants did not have any incentive to lie about their programming skills (or lack of). Figure 11 depicts the performance of the "no strategy description" and "post strategy description" groups according to the three measures defined for the "doors game". As depicted in the figure, the strategy expressing activity had

[4] For a comparison between AMT and other recruitment methods see [36].

no influence whatsoever on performance in any of the measures (all differences are statistically insignificant). These results may indicate that there is no difference in the behavior of the general population and CS students after describing their strategy. Furthermore, PDA-design can be suggested for strategy improvement while strategy description cannot.

Here, again we provide an in depth analysis of the distribution of the number of switches at the individual level (Figure 12). As illustrated in Figure 12, while the general population generally performs worse than CS students in this game, the pattern of transition between strategies after strategy description is quite similar to the one described in the former section when experimenting with CS students. Similar qualitative results were found using an in depth analysis of the elimination point measure.

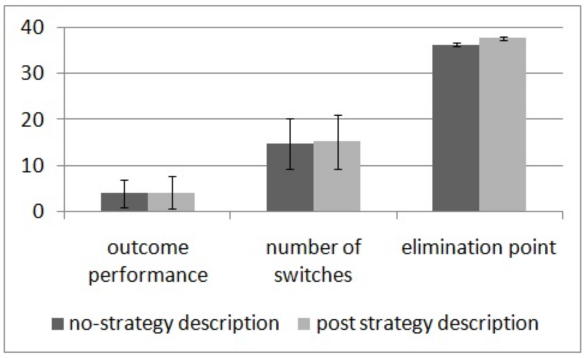

Fig. 11 Comparison of the results of AMT participants without strategy description and post strategy description over the three measurements.

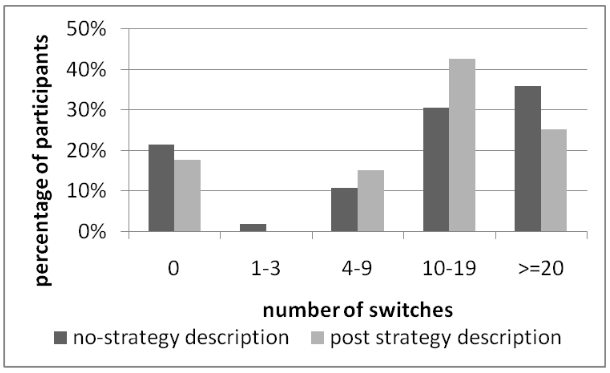

Fig. 12 An in depth comparison between the number of switches without strategy description and post strategy description of AMT participants.

In order to confirm our results that the improvement of peoples' strategy after designing PDAs is not due to a regular process of re-thinking the strategy, we conducted another experiment. For this experiment we recruited 50 participants from AMT and had them play the "doors game" twice. Before each game the participants were asked to write down their strategy (for the second time; participants were allowed to see their first strategy description and could use it as is if they had no change in strategy). This design ensures that participants can re-think their strategy after they try it out. The performance difference between those playing their strategy once and those who got to describe it twice was found to be not statistically significant, in all three measures. In fact, only one person changed his strategy to the optimal one after describing his strategy for the second time, indicating that the affect of designing PDAs on the designers' strategies is not due to "re-thinking" the strategies, but due to other factors that are rooted in the way people think when they engage in programming.

7 The Penetration Detection Domain

In order to strengthen and generalize the results presented in the previous sections, we conducted another set of experiments with a different game from the security domain called "the penetration detection" (PenDet). The PenDet game considers the problem of patrolling a perimeter of a closed area by a team of robots, as described in much detail by [3]. In this problem, a team of k robots is required to repeatedly travel along a cyclic path of N segments in order to detect penetration attempts that are controlled by an adversary. The k given robots are spread uniformly (in time) along the perimeter with distance d between every two consecutive robots, and maintain this uniform inter-robot distance along the execution. At each time step all robots either continue straight with probability p, or turn around with probability $1-p$ (and if so, they stay in place for one time unit). Therefore, the probability p characterizes the patrolling strategy. It is assumed that the adversary—after choosing a penetration segment—remains in its position for t time units (known as *penetration time*), during which it might be detected by a robot that passes through its chosen penetration segment. In the version of the game we used, the user plays the adversary's role and acts against simulated robots, which execute a specific random-based patrolling strategy. The adversary, played by a human player/PDA, becomes acquainted to the initial robots location and observes their movement. Then, she is asked to choose a segment through which she recommends to penetrate. The optimal strategy for the adversary is to choose the segment associated with the *lowest* probability of penetration detection. The optimal strategy for the patrolling robots is the one associated with the *highest* probability of penetration detection. This strategy can be efficiently computed when the robots face a full knowledge adversary [2], random knowledge adversary, zero-knowledge adversary [4] or an adversary that may analytically estimate the weakest segment of the patrol (through which it would like to penetrate) to some extent [3]. However, it is unclear when to use which strategy, as the exact adversarial model (level of knowledge) is unknown.

The reason for choosing this game is because the goal of the game can easily be explained. It is essentially a one-shot game thus relatively quick to play and easy to construct a strategy for. At the same time the strategies people use highly vary because of the probabilistic nature of the game and the relatively large number of possible penetration segments.

Out of 12 PenDet game variants described in [30], in four variants prior research reports a tendency of people to attempt penetrating through the segment placed exactly in-between two robots. This behavior reflects people's tendency towards middle points rather than attempting to follow an actual ppd calculation. The four variants, denoted 2-Neighbor, 3-Neighbor, MaxiMin, MidAvg, differ primarily in the patrolling strategy used by the robots:

2-Neighbor The robots assume that they are facing an adversary that does not know the weakest segment, but estimates the area of the weakest segment (the neighborhood - two close segments). The strategy, therefore, sets a value for p such that the expected ppd of the two "weakest" neighboring segments (in terms of ppd) is maximized.

3-Neighbor Similar to 2-Neighbor, but with a neighborhood of three segments (corresponds to larger uncertainty of the adversary with respect to the weakest segment of the patrol).

MaxiMin The robots assume they are facing an adversary that knows exactly the patrol algorithm and the current location of the robots. The robots' strategy is the one that maximizes the minimal ppd along the perimeter.

MidAvg This is a heuristic strategy, that uses the average of the p value of the deterministic algorithm ($p = 1$, proven optimal against a random adversary in [4]) and the p value calculated by the MaxiMin algorithm (optimal against a full knowledge adversary).

All the above strategies are random-based, and follow the patrolling framework described in [2,3]. For the four variants we adopted we used the exact settings that were used in [30], i.e., setting t to 6, k to 3, and the number of segments between the robots (d) is 8. Under the setting described above p value for 2-Neighbor equals 0.7604, for 3-Neighbor 0.9095, for MaxiMin 0.7037, and for MidAvg 0.8518. Our goal is thus to show that people strategies for the PenDet game has changed primarily in the sense that after designing PDAs people's tendency to penetrate through middle points decreases and instead they choose to penetrate through segments associated with a lower ppd compared to the middle segment's ppd.

We recruited 36 human subjects, all senior computer science undergraduate students and we used a *within subject* design. In both the first and third stages of the experiments, human subjects played the role of the adversary in a web-based version (see Figure 13) of the four different game variants described above. Subjects viewed the patrolling behavior of the robots in the games they were playing, after which they had to click on a segment through which they wish to penetrate. In the second stage of the experiment subjects were asked to design a PDA for the adversary role that will play the game on their behalf. Before playing 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.

The measure used to evaluate the performance of the patrol strategy, and consequently the performance of the adversary playing against this strategy, is the expected probability of penetration detection (ppd): higher expected values of ppd means better performance of the strategy. Equivalently, from the adversary's perspective (the human subjects and the PDAs), lower expected values of ppd are better.

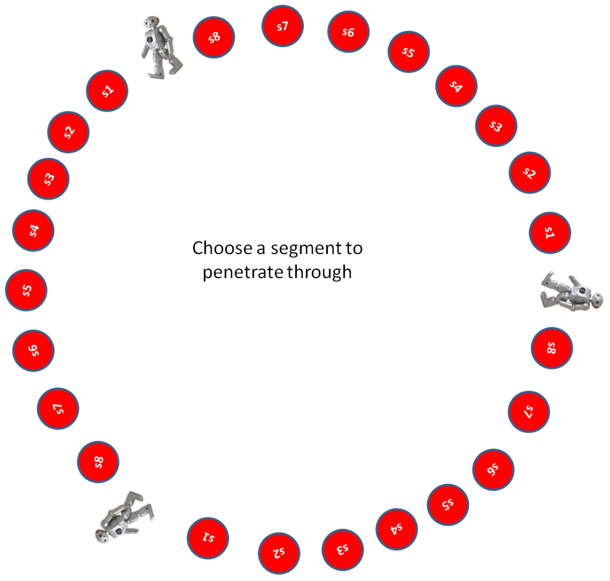

Fig. 13 The penetration detection web-based game. There are 3 robots and 8 segments (s1..s8) between each pair of patrolling robots. The participant needs to choose one of the segments (red circles) to penetrate through.

Figure 14 describes the expected ppd values achieved in the four game variants of the PenDet game for the two stages (pre-PDA and post-PDA). The difference in performance between the pre-PDA and the post-PDA groups is statistically significant ($p - value < 0.03$), indicating that indeed different strategies have been used. In particular, we observe that the expected ppd values of the post-PDA stage has dropped considerably, supporting the hypothesis that the process of PDA-development improves one's strategy.[5]

[5] Note that better performance of the adversary (the human players) infers worse performance of the patrolling strategy.

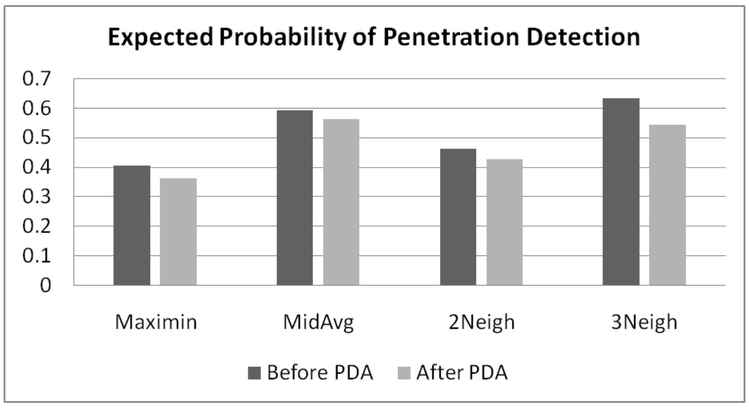

Fig. 14 The probability of penetration detection before and after developing the PDA, for $d = 8, t = 6$.

In order to understand *why* the subjects' performance have improved, i.e., why the expected probability of successful penetrations was higher after designing PDAs, we have compared between the penetration segments chosen by the subjects in both stages of the game. We demonstrate these choices for two instances: when the simulated robots patrol according to the MaxiMin strategy (Figure 15), and when they patrol using the 3-Neighbor strategy (Figure 16). The figures present the percentage of penetration attempts per segment before and after PDA development. The segments having the lowest expected ppd based on the patrolling strategy for MaxiMin are segments s_5, s_7 with ppd $= 0.24$ (in both segments), and for 3-Neighbor segment s_7 with ppd $= 0.138$. Before designing PDAs, the subjects tended to choose the mid segments ($s_3 - s_5$): 65% for MaxiMin, and 43% for 3-Neighbor. After developing PDAs, the penetration segments chosen by the subjects shifted to segments $s_5 - s_8$: 61% for both MaxiMin and 3-Neighbor (in comparison to 38% and 27% for segments $s_3 - s_5$ for MaxiMin and 3-Neighbor, respectively). It seems that the initial instinct of people is to choose the middle point for penetration, and after the process of developing a PDA this instinct changes with an understanding of finding the weakest segment (i.e., the one that is associated with the lowest probability of being detected).

8 Related Work

The use of agents in general for human behavior simulation is quite extensive in the AI literature [47,11,16,31]. 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).

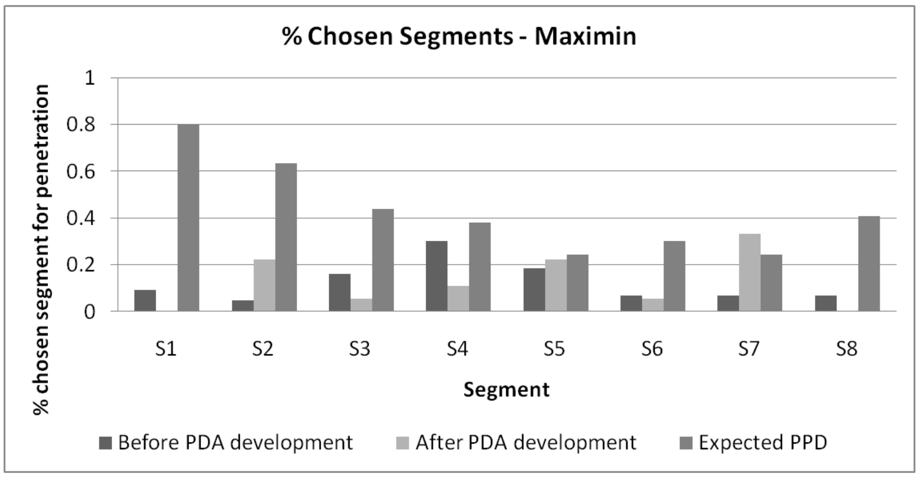

Fig. 15 Choice of penetration segments by the subjects before and after developing PDAs.

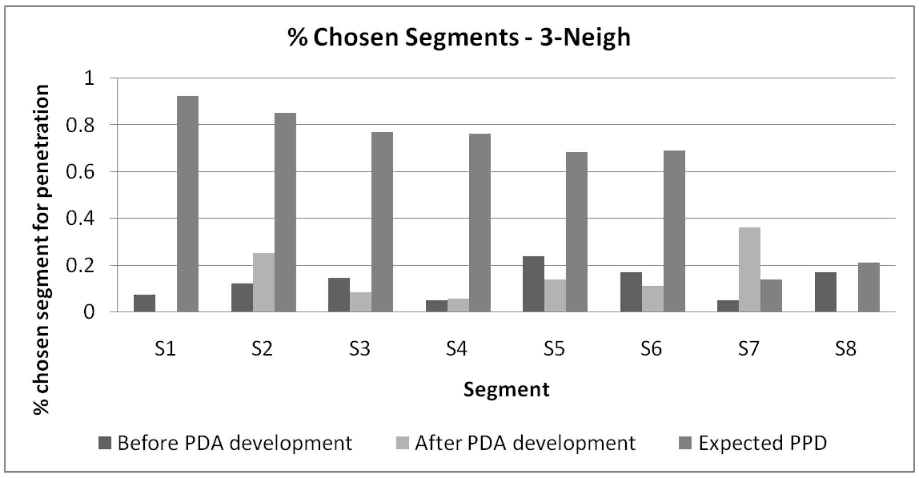

Fig. 16 Choice of penetration segments by the subjects before and after developing PDAs.

These include, among other methods, statistical-data based modeling [43], pre-specification of agents' roles using a set of parameters according to which the agents act [32], using pre-defined events and reactions [34], defining a number of basic behaviors from which all of the agents' more complex behaviors are constructed [40, 44] or using a combination of rules and finite state machines to control an agent's behavior using a layered approach [45]. 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 [25] 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 [46,13].

PDAs technology has been widely used in recent years. For example, in Kasbah [17] PDAs that buy and sell were used for evaluating an electronic marketplace. In Colored Trails [26], PDAs were used for reasoning about players' personalities in uncertain environments. Other works, e.g., [31,30,16,38] 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 [39]. In the strategy method people state their action for every possible situation that may arise in their interaction. The question of whether or not the strategy method can ensure real-life behaviors has been extensively researched in the last decade. Several authors reported no substantial differences between the choices made using the strategy method and those made by people in experiments, e.g., in the sequential dictator game [15], in measuring people's willingness to pay for insurance [9,10] and even in sequential exploration problems [42]. Others, however, reported significantly different results with the strategy method compared to ones obtained by people playing the same game, e.g., with trust game variations [14], trust-punishment games [12] and social good allocation games [33]. One of the explanations given for the difference between the two methods is the tendency of people to be influenced by emotions, where in strategy method people are in a "cold" state and are less emotionally arouse [14]. 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 [16,38,18]. 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 [27]. Examples of such discrepancies include over-reporting of political participation [8] and contrasting results between self-reported and performance-based levels of physical limitations [29]. 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 [18,38]. 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 [31, 30]. 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 [24,35,28, 19]. 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.

9 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 domains used for our experiments. While making general claims indeed requires further experimentation, possibly with additional, more complex and more varied domains, the fact that the same effect was reflected in both games we experimented with suggests that there is a large class of settings where the phenomena is likely to occur. The two games that were used highly vary in their characteristics (e.g., their one-shot vs. repetitive nature and the binary vs. accumulated payoff) and in particular in the ability of people to reason about the optimal strategy—while in the "doors game" the optimal strategy can be calculated by people, in the penetration detection game people can typically come up with near-optimal solutions. Furthermore, even the roots of people's inability to reach the optimal solution are different in the two games. In the first it is due to a psychological bias and in the second it is due to computational difficulties.

Overall, the aim of this work is 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 games, when programming the strategy the participants did not have the infrastructure to test how well it performs. Therefore the effect is most likely 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 game" are primarily rooted in people's tendency to keep all their options available [41], which is a psychological effect. In the penetration detection game the inefficiencies characterizing strategies of individuals are primarily rooted in people's attraction towards middle points. 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 [37]) 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.

In this work there was one optimal strategy. We believe that in other domains where there may be multiple optimal strategies or the notion of optimality may be tied to user preference, the same phenomena recur. However, in order to investigate the behavior of people in these domains a utility elicitation mechanism is required. Therefore, we purposely avoid these kinds of domains in the work presented here since it is difficult and sometimes impossible to measure improvement of utility when the utility function is not well defined.

All in all, we believe 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. In addition it will be interesting to investigate the effect of PDA development over time, i.e., experimenting if the same set of people would deliver similar performance when played the same game after a reasonable period of time of not having played the game.

Acknowledgements We would like to thank Amos Azaria for helping us to collect the data for the no-PDA experiment. A preliminary version of this paper was presented at HAIDM14, and appears in the proceedings of AAAI 2014 [21] and AAMAS 2014 [22].

References

1. MOSS a system for detecting software plagiarism. `https://theory.stanford.edu/~aiken/moss/`. Accessed: 2015-05-06
2. Agmon, N., Kraus, S., Kaminka, G.: Multi-robot adversarial patrolling: Facing a full-knowledge opponent. JAIR **42**, 887–916 (2011)
3. Agmon, N., Kraus, S., Kaminka, G., Sadov, V.: Adversarial uncertainty in multi-robot patrol. In: Proc. of IJCAI (2009)
4. Agmon, N., Sadov, V., Kaminka, G., Kraus, S.: The impact of adversarial knowledge on adversarial planning in perimeter patrol. In: Proc. of AAMAS, pp. 55–62 (2008)
5. AMT: Amazon mechanical turk. www.mturk.com (2010)
6. Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.E.: Gambling in a rigged casino: The adversarial multi-armed bandit problem. In: Foundations of Computer Science, pp. 322–331 (1995)
7. Azaria, A., Richardson, A., Elmalech, A., Rosenfeld, A.: Automated agents' behavior in the trust-revenge game in comparison to other cultures. In: AAMAS (2014)
8. Bertrand, M., Mullainathan, S.: Do people mean what they say? implications for subjective survey data. American Economic Review **91**(2), 67–72 (2001)
9. Bosch, A., Silvestre, J.: Does risk aversion or attraction depend on income? an experiment. Economics Letters **65**(3), 265–273 (1999)
10. Bosch, A., Silvestre, J.: Risk aversion and embedding bias. Materia (s) **1**, 01–2006 (2006)
11. Bosse, T., Gerritsen, C., Hoogendoorn, M., Jaffry, S., Treur, J.: Agent-based vs. population-based simulation of displacement of crime: A comparative study. WIAS **9**(2), 147–160 (2011)
12. Brandts, J., Charness, G.: Truth or consequences: An experiment. Management Science **49**(1), 116–130 (2003)
13. Brewer, P., Huang, M., Nelson, B., Plott, C.: On the behavioral foundations of the law of supply and demand: Human convergence and robot randomness. Experimental economics **5**(3), 179–208 (2002)

14. Casari, M., Cason, T.: The strategy method lowers measured trustworthy behavior. Economics Letters **103**(3), 157–159 (2009)
15. Cason, T., Mui, V.: Social influence in the sequential dictator game. Journal of Mathematical Psychology **42**(2), 248–265 (1998)
16. Chalamish, M., Sarne, D., Lin, R.: Enhancing parking simulations using peer-designed agents. ITS **14**(1), 1–7 (2012)
17. Chavez, A., Maes, P.: Kasbah: An agent marketplace for buying and selling goods. In: Proc. of Intelligent Agents and Multi-Agent Technology, pp. 8–12 (1996)
18. Cheng, K., Zuckerman, I., Nau, D., Golbeck, J.: The life game: Cognitive strategies for repeated stochastic games. In: Proc. of social computing (socialcom), pp. 95–102 (2011)
19. Clements, D.H., Gullo, D.F.: Effects of computer programming on young children's cognition. Journal of Educational Psychology **76**(6), 1051 (1984)
20. Elmalech, A., Sarne, D.: Evaluating the applicability of peer-designed agents in mechanisms evaluation. In: Proc. of IAT, pp. 374–381 (2012)
21. Elmalech, A., Sarne, D., Agmon, N.: Can agent development affect developers strategy? In: Proc. of AAAI, pp. 923–929 (2014)
22. Elmalech, A., Sarne, D., Agmon, N.: Peer designed agents: Just reflect or also affect? In: Proc. of AAMAS, pp. 1429–1430 (2014)
23. Elmalech, A., Sarne, D., Grosz, B.J.: Problem restructuring for better decision making in recurring decision situations. Autonomous Agents and Multi-Agent Systems **29**(1), 1–39 (2015)
24. Feurzeig, W., Papert, S.A., Lawler, B.: Programming-languages as a conceptual framework for teaching mathematics. Interactive Learning Environments **19**(5), 487–501 (2011)
25. Gode, D., Sunder, S.: Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. Journal of political economy **101**, 119–137 (1993)
26. Grosz, B., Kraus, S., Talman, S., Stossel, B., Havlin, M.: The influence of social dependencies on decision-making: Initial investigations with a new game. In: Proc. of AAMAS, pp. 782–789 (2004)
27. Harries, C., Evans, J., Dennis, I.: Measuring doctors' self-insight into their treatment decisions. Applied Cognitive Psychology **14**, 455–477 (2000)
28. Jeffries, R., Turner, A.A., Polson, P.G., Atwood, M.E.: The processes involved in designing software. Cognitive skills and their acquisition pp. 255–283 (1981)
29. Kempen, G., Van Heuvelen, M., Van den Brink, R., Kooijman, A., Klein, M., Houx, P., Ormel, J.: Factors affecting contrasting results between self-reported and performance-based levels of physical limitations. Age and Ageing **25**(6), 458–464 (1996)
30. Lin, R., Kraus, S., Agmon, N., Barrett, S., Stone, P.: Comparing agents' success against people in security domains. In: Proc. of AAAI (2011)
31. Lin, R., Kraus, S., Oshrat, Y., Gal, Y.: Facilitating the evaluation of automated negotiators using peer designed agents. In: Proc. of AAAI, pp. 817–822 (2010)
32. Massaguer, D., Balasubramanian, V., Mehrotra, S., Venkatasubramanian, N.: Multi-agent simulation of disaster response. In: Proc. of Agent Technology for Disaster Management, pp. 124–130 (2006)
33. Murphy, R., Rapoport, A., Parco, J.: Credible signaling in real-time trust dilemmas. University of Arizona, mimeo (2007)
34. Musse, S., Thalmann, D.: Hierarchical model for real time simulation of virtual human crowds. Transactions on Visualization and Computer Graphics **7**(2), 152–164 (2001)
35. Nickerson, R.: Computer programming as a vehicle for teaching thinking skills. Thinking: The Journal of Philosophy for Children **4**(3/4), 42–48 (1982)
36. Paolacci, G., Chandler, J., Ipeirotis, P.: Running experiments on amazon mechanical turk. Judgment and Decision Making **5**(5), 411–419 (2010)
37. Resnick, M., Maloney, J., Rosenbaum, E., Silver, J., Silverman, B., et al.: Scratch: programming for all. Communications of the ACM **52**(11), 60–67 (2009)
38. Rosenfeld, A., Kraus, S.: Modeling agents based on aspiration adaptation theory. AAMAS **24**(2), 221–254 (2012)
39. Selten, R., Mitzkewitz, M., Uhlich, G.: Duopoly strategies programmed by experienced players. Econometrica **65**(3), 517–555 (1997)

40. Shao, W., Terzopoulos, D.: Autonomous pedestrians. Graphical Models **69**(5–6), 246–274 (2007)
41. Shin, J., Ariely, D.: Keeping doors open: The effect of unavailability on incentives to keep options viable. Management Science **50**(5), 575–586 (2004)
42. Sonnemans, J.: Decisions and strategies in a sequential search experiment. Journal of Economic Psychology **21**(1), 91–102 (2000)
43. Takahashi, T., Tadokoro, S., Ohta, M., Ito, N.: Agent based approach in disaster rescue simulation - from test-bed of multiagent system to practical application. In: Proc. of RoboCup, pp. 102–111 (2002)
44. Terzopoulos, D., Tu, X., Grzeszczuk, R.: Artificial fishes: Autonomous locomotion, perception, behavior, and learning in a simulated physical world. Artificial Life **1**(4), 327–351 (1994)
45. Ulicny, B., Thalmann, D.: Towards interactive real-time crowd behavior simulation. Computer Graphics Forum **21**(4), 767–775 (2002)
46. Van Boening, M., Wilcox, N.: Avoidable cost: Ride a double auction roller coaster. The American Economic Review **86**, 461–477 (1996)
47. Zhang, Y., Biggers, K., He, L., Reddy, S., Sepulvado, D., Yen, J., Ioerger, T.: A distributed intelligent agent architecture for simulating aggregate-level behavior and interactions on the battlefield. In: Proc. of SCI, pp. 58–63 (2001)