Software ENgineering

Why agents for automated negotiations should be adaptive

D.D.B. van Bragt, J.A. La Poutré

Report SEN-R0208 April 30, 2002

ISSN 1386-369X

Why Agents for Automated Negotiations Should be Adaptive

D.D.B. van Bragt J.A. La Poutré[†]
David.van.Bragt@cwi.nl Han.La.Poutre@cwi.nl
[†] *Also with the School of Technology Management, Eindhoven University of Technology*
De Lismortel 2, 5600 MB Eindhoven, The Netherlands

CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

ABSTRACT
We show that adaptive agents on the Internet can learn to exploit bidding agents who use a (limited) number of fixed strategies. These learning agents can be generated by adapting a special kind of finite automata with evolutionary algorithms (EAs). Our approach is especially powerful if the adaptive agent participates in frequently occurring micro-transactions, where there is sufficient opportunity for the agent to learn online from past negotiations. More in general, results presented in this paper provide a solid basis for the further development of adaptive agents for Internet applications.

2000 Mathematics Subject Classification: 68T05, 68T20, 91A05, 91A10, 91A18, 91A20, 91A22, 91A26
1998 ACM Computing Classification System: G.1.6, I.2.6, I.2.8
Keywords and Phrases: e-commerce, adaptive agents, automated negotiations, evolutionary algorithms, finite automata
Note: Work carried out under theme SEN4 "Evolutionary Systems and Applied Algorithmics". An earlier version of this paper has been presented at the 7th International Conference of the Society for Computational Economics on Computing in Economics and Finance (CEF'2001) (New Haven, CT, USA, June 28-29, 2001). An abstract of this paper has also been published in the Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001) (San Francisco, CA, USA, July 7-11, 2001, Morgan Kaufmann Publishers, p. 1093). This research has been performed within the framework of the project "Autonomous Systems of Trade Agents in E-Commerce", which is funded by the Telematics Institute in the Netherlands.

1. Introduction

The rapid growth of a global electronic market place, together with the establishment of standard negotiation protocols, currently leads to the development of multi-agent architectures in which artificial agents can negotiate on behalf of their users [4, 8]. Most of today's (prototype) systems for automated negotiations, like Kasbah or Tête-à-Tête, use simple and static negotiation rules. We show, however, that such "fixed" bidding agents can be exploited by more sophisticated "adaptive" agents. These adaptive agents are able to learn strategies which perform (almost) optimally against a variety of fixed opponents. Furthermore, they are able to adapt their strategies online to deal with changing opponents and open environments.

Our results thus imply that fixed bargaining strategies should not be used for bounded environments (like specific business-to-business markets) with only a limited number of participants and agent types. Instead, we recommend to design and use adaptive agents. For open environments, like the Internet, the usage of an (almost) fixed strategy offered by some provider is not recommended for the same reason.

We implement an adaptive agent as a collection of bargaining strategies (represented as finite automata) which is optimized by an evolutionary algorithm (EA) [11, 2]. EAs transfer the principles of natural evolution, first discovered by Darwin, to a computational setting. These algorithms have been used in the past, with considerable success, to solve difficult optimization problems [11, 2]. More recently, EAs have also been used as an innovative and powerful way to develop adaptive agents who are able to operate successfully in multi-agent systems [12, 5, 1, 9, 15, 14, 6].[1]

An adaptive agent (using an EA) is able to improve its performance by (i) experimenting with novel strategies ("mutation"), (ii) combining ("crossing-over") previously-used strategies, and (iii) removing inferior strategies ("selection"). Such an evolutionary approach is especially powerful if the agent participates in frequently occurring micro-transactions (i.e., transactions with a small value). In this case, the agent has sufficient opportunity to experiment with different strategies and learn online from past negotiations.

We focus on so-called *multi-issue* negotiations in this paper. In multi-issue negotiations not only one aspect, namely the price of a product, is important, but other aspects are taken into account as well (for instance the quality of the product, the delivery time, etc.). These multi-issue negotiations give the two parties the opportunity to reach more satisfactory deals compared to negotiations which only concern the price of the product. The complexity of multi-issue negotiations increases rapidly, however, when the number of issues becomes large. This explains the need for powerful search techniques (like EAs) to generate effective bargaining strategies.

The remainder of this paper is organized as follows. The class of bargaining problems that we consider is briefly discussed in Section 2. Section 3 then explains how *adaptive* agents can be generated for this class of problems using evolutionary techniques. The bargaining strategies used by the adaptive agent consist of finite automata. Section 4 discusses the automaton design that we use. In Section 5, we describe the opponents of the adaptive agent in the computational experiments. These experiments are presented in Section 6. In this section the adaptive agents compete against different opponents in various (static and dynamic) environments. Section 7 concludes.

2. The multi-issue bargaining problem

We consider the well-known alternating-offers bargaining protocol [13]. When the game starts, one of the two parties is designated (at random) to make the first offer. At time $t = 0$, this player (denoted as "player 1") makes an offer. The other player (denoted as "player 2") then accepts or rejects the initial offer. If the initial offer is rejected, player 2 makes a counter offer in the next round (at $t = 1$). This alternating process of making proposals then continues until an offer is accepted or until the bargaining deadline is reached (at $t = n$). If no agreement has been reached before the deadline (that is, for $t < n$) both players receive nothing. We set n equal to 10 in this paper.

Players are allowed to bargain over multiple issues simultaneously (as we mentioned in the Introduction). Formally, a multi-issue offer can be denoted as a vector $\vec{o}$. The i-th component of this vector, denoted as o_i, specifies the share of issue no. i that the proposing player receives if his offer is accepted. The index i ranges from 0 to $m - 1$, where m is the total number of issues. We assume (without loss of generality) that the total bargaining surplus available per issue is equal to unity.

Each player independently evaluates the utility (or "payoff") of the received offers. We assume that the players use an additive utility function. The utility function $u(\vec{o})$ can be written in this case as $u(\vec{o}) = \sum_{i=0}^{m-1} w_i o_i$, where $\vec{w}$ is an m-dimensional vector of "weights" for all issues. The weight vector is normalized (i.e., $\sum_{i=0}^{m-1} w_i = 1$).

3. Designing adaptive agents

We implement an adaptive agent as a collection of strategies which is optimized by an evolutionary algorithm (EA) [11, 2]. EAs transfer the principles of natural evolution, first discovered by Darwin, to a computational setting. These algorithms have been used in the past, with considerable success, to solve difficult optimization problems. Examples include problems with huge search spaces, multiple local optima, discontinuities, and noise [11, 2].

As in natural ecosystems, EAs typically evolve a population of individuals. Here, each individual is a bargaining strategy of the adaptive agent. These bargaining strategies consist of a special kind

[1]An excellent overview of (earlier) research on learning in multi-agent systems is given in [16, Ch. 6].

of finite automata (see Section 4). Like in nature, the survival of each bargaining strategy depends on its fitness (the "survival of the fittest" concept). We use the mean utility obtained by a strategy (against the different opponents) as its fitness measure (see Section 5).

An outline of the EA, which optimizes the strategies of the adaptive agent, is given in (the lower part of) Fig. 1. A technical description of our EA implementation is given in Appendix 2. The interested

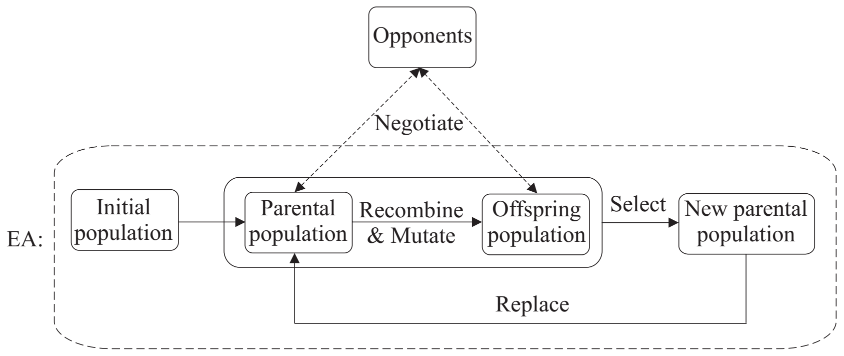

Figure 1: Iteration loop of the evolutionary algorithm (EA). This algorithm updates the population of strategies used by the adaptive agent.

reader is referred to this appendix for further details.

The EA starts with a randomly initialized "parental" population of bargaining strategies. The fitness of these strategies is then determined by letting them negotiate with a group of opponents (see Section 5). The mean utility obtained in these negotiations (averaged over all opponents) is used as the fitness value of the evolutionary strategy. Subsequently, "offspring" strategies are created (see Fig. 1). The offspring strategies are generated by selecting two strategies in the parental population (at random, with replacement). Two offspring strategies are then created from the two parents. These offspring strategies are generated by mutation, which (randomly) changes parts of the strategies, and recombination, which combines ("crosses-over") the two parental strategies. This process is repeated until the offspring population is filled. The fitness of the new offspring is again determined by negotiation with the pool of opponents. In the final stage (see Fig. 1), the fittest strategies from the parental and offspring populations are selected as the new "parents" for the next iteration. This final step completes one iteration (or "generation") of the EA. When this iterative process is repeated by the adaptive agent, highly-fit strategies will evolve in the course of time (see Section 6).

4. Bargaining automata

We represent the agent's bargaining strategies by a special kind of finite automata. Our automaton design follows, and further extends, previous (theoretical) work by Binmore et al. [3]. Their finite automata are adaptations of Moore automata [7]. A Moore automaton can be represented by a four-tuple (S, s_{init}, h, f), where $S = \{s_0, s_1, ..., s_{q-1}\}$ is a finite set of q states; $s_{init} \in S$ is the automaton's initial state; $h(s) : S \to A$ is an output function, associating an action $h(s)$ from a set of actions A with every state $s \in S$; and $f(s, a)$ is a transition function identifying the state to which the automaton shifts after receiving action $a \in A$ as an input in state $s \in S$. Note that the set of possible outputs associated with a state (i.e., the set A) is simply equal to the set of all possible (multi-dimensional) offers (i.e., $[0, 1]^m$). We use automata with 8 states in the computational experiments (i.e., $q = 8$).

Binmore et al. argue that a player should be able to behave differently when selected to be the initial proposer or the initial responder (see [3, pp. 263-264]). We therefore distinguish between an initial state for the initial proposer, $s_{init,1}$, and an initial state for the initial responder, $s_{init,2}$. This allows the automata to condition their behavior on their role in the game.

To avoid a continuation of the bargaining process ad infinitum, Binmore et al. also assume that each automaton contains at least one "acceptance" state. A transition to this state signals that an offer has been accepted and terminates the game. We use a slightly different approach. Instead of adding an acceptance state, we associate an acceptance threshold value τ_{acc} with each state. When an automaton receives an offer $\vec{o}$ from the opponent, it first makes a transition (as specified by the transition function $f(s, \vec{o})$). The automaton then compares the utility $u(\vec{o})$ of the proposal with the acceptance threshold value τ_{acc} which is attached to the state where it has just arrived. If $u(\vec{o}) \geq \tau_{acc}$, the proposal is accepted, otherwise a counter offer is made. Computational experiments show that this acceptance model yields superior results compared to a model with acceptance states.

Finally, we specify the transition function $f(s, \vec{o})$, which specifies the state to which the automaton shifts when receiving a proposal $\vec{o}$ in state s. In principle, arbitrarily complex mappings from actions to states are allowed. It is for instance possible to construct automata with q different transitions per state. Such automata are very complex, however, and are therefore not useful in a computational setting. We therefore propose a more efficient transition model, which requires automata with only two different transitions per state.

We first associate an (m-dimensional) transition threshold $\vec{\tau_{tr}}$ with each state. The proper transition is then determined by comparing the m components of the opponent's proposal $\vec{o}$ with the corresponding elements of $\vec{\tau_{tr}}$. In our transition model, only two transitions (denoted as f_0 and f_1) are possible per state. Transition f_1 is selected if $o_i \geq \tau_{tr,i}$ for all $i = 0, ..., m-1$. Otherwise, transition f_0 is selected. The transition threshold $\vec{\tau_{tr}}$ thus partitions the set A in two disjunct sets, for each of which a different transition can be made.

Figure 2 gives an example of an automaton with two states which is able to negotiate over two issues. When this automaton is designated to submit the first offer in the game, the initial state is s_1

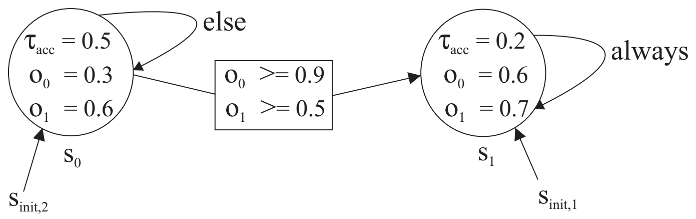

Figure 2: Example of an automaton with two states which is able to negotiate over two issues.

(because $s_{init,1} = s_1$). We assume that $f_0 = f_1 = s_1$ for this state. This implies that the automaton will remain in state s_1 and repeat the offer $(0.6, 0.7)^T$ until the utility of the opponent's offer exceeds (or equals) the acceptance threshold $\tau_{acc} = 0.2$ (and a deal is made).

When this automaton is the initial responder, it shifts to state s_0 (because $s_{init,2} = s_0$). We assume that $\tau_{tr,0} = 0.9$, $\tau_{tr,1} = 0.5$, $f_0 = s_0$ and $f_1 = s_1$ for this state. The automaton will thus make a transition from state s_0 to state s_1 if the opponent demands at least 0.9 for issue no. 0 and at least 0.5 for issue no. 1. Otherwise, the automaton remains in state s_0. After the appropriate transition has been made, the utility of the opponent's offer is compared with the acceptance threshold which is associated with the current state. If this threshold is not reached, the opponent's proposal is rejected and a counter offer is made, etc.

5. A test suite of bargaining strategies

The opponents of the adaptive agent are briefly introduced in this section. Appendix 1 contains a more detailed description of the different opponents. We select the opponents from three important "families" of bargaining tactics: time-dependent, minimal-concession, and behavior-dependent tactics [9]. Time-dependent strategies do not take the behavior of their opponent into account, but simply base their move on the passage of time. Minimal-concession strategies typically make small concessions on (one on more) issues each time they propose an offer. Behavior-dependent strategies base their move on the behavior of their opponent during the game.

It is rather straightforward to derive an optimal strategy against an opponent with a known tactic and preferences. The problem becomes much more complicated, however, when an agent is bargaining with a variety of different opponents without knowing the opponents' tactics or preferences. A strategy that performs well against one opponent type may in fact lead to very poor results against other opponents in this case.

The combination of time-dependent, minimal-concession, and behavior-dependent strategies is particularly challenging in this respect. To reach an agreement against a behavior-dependent opponent, one needs to concede on one or more issues (otherwise the opponent will not make a concession as well). Against a time-dependent strategy conceding is an inefficient strategy, on the other hand, because such an opponent will automatically concede the entire bargaining surplus when the deadline is almost reached. Waiting until the deadline approaches is also a good strategy when playing against a minimal-concession strategy. A significant concession should be made just before the deadline, however, otherwise a deal might not be reached with this kind of opponent.

An effective bargaining agent should thus be able to discriminate between various opponent types and behave differently against different opponents. This is a challenging task, especially when the tactics and preferences of the opponents are unknown. We show, however, in the next section that adaptive agents (based upon evolving finite automata) can meet these requirements.

6. Results

Section 6.1 discusses the performance of adaptive agents in a bounded negotiation environment (like specific business-to-business markets). In these computational experiments, the pool of opponents is not changing over time, i.e., the environment is *static.* We show that adaptive agents can effectively exploit a set of fixed opponents when operating in such an environment.

Section 6.2 extends these results by considering situations where the number of opponents is *changing* over time. This models an open and dynamic environment (like the Internet), where new competitors arrive at the marketplace and others leave. We show, in a series of computational experiments, that adaptive agents are able to adjust their behavior rapidly when new opponents enter the marketplace. The adaptive agents are also able to learn effective negotiation strategies when the group of opponents is changing very frequently over time.

6.1 Bounded environments

In Section 6.1, we consider the situation where an adaptive agent bargains over two issues with a limited set of opponents (nine in total, see Appendix 1.1). The issue weights are set equal to $w_0 = 0.7$ and $w_1 = 0.3$ for the adaptive agent in this case. For the opponents, $w_0 = 0.3$ and $w_1 = 0.7$. This means that the adaptive agent clearly prefers the first issue, whereas the opponents prefer the second issue. Such a bargaining problem is called "integrative" in the sense that both parties can reach a mutually beneficial agreement by conceding on their least important issue.

In Section 6.1, we consider a much more complex bargaining problem involving eight issues and a large number of opponents (22 in total, see Appendix 1.2). The opponent group contains members with different preferences in this case.[2] This increases the diversity in the group of opponents, and, consequently, the difficulty of the bargaining problem.

Negotiations over two issues Figure 3 shows the performance of the adaptive agent for a two-issue bargaining problem. The performance of the agent (shown on the vertical axis) is defined as the mean fitness across the evolving strategies maintained by the agent. The agent starts with a population of 100 randomly initialized automata (with eight states). Each generation, the agent's population of strategies is then updated by an EA (as indicated in Fig. 1).

The fitness of each evolving strategy is determined in a competition with the set of nine opponents described in Appendix 1.1. The maximum fitness level that can be reached against these opponents

[2]The (eight-dimensional) weight vector is set equal to $\frac{1}{3.9}(0.7, 0.3, 0.5, 0.2, 0.3, 0.4, 0.5, 1.0)^T$ for the adaptive agent in these experiments. The issue weights for each of the 22 opponents are drawn from a flat distribution in this case.

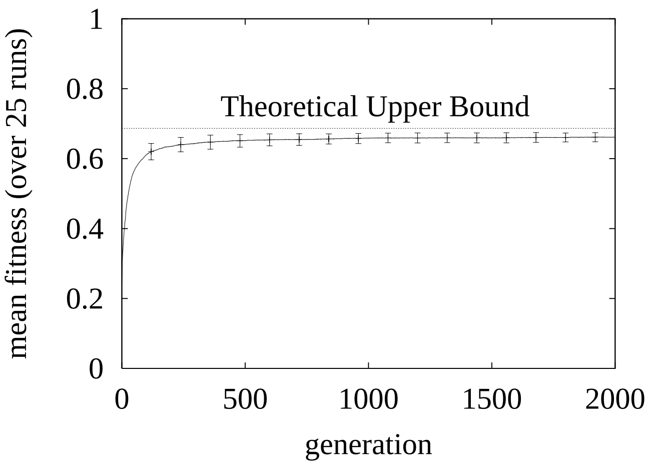

Figure 3: Performance of the adaptive agent for a two-issue bargaining problem. Means and standard deviations are calculated across 25 experiments with different random seeds.

(on average against the entire group) is $\approx$ 0.687. This theoretical upper bound is indicated with the horizontal line in Fig. 3. A strategy can only reach this performance level by executing a best-response tactic against each opponent.[3]

Figure 3 shows that the performance level of the adaptive agent approaches this upper limit in the long run. We can thus conclude that, after a training period, an adaptive agent can reach a very high performance level against a variety of opponents. This is a striking result, since the adaptive agent receives no explicit information about the identity or preferences of the opponents.

Example Figure 4 shows an example of a four-state automaton which is capable of negotiating about two issues. This automaton has a fitness of $\approx$ 0.664. This fitness level is only 3% below the theoretical upper limit, i.e., performance is nearly optimal against *all* opponents. Performance improves only slightly if automata with more than four states are considered. The initial state of the automaton is s_0 when it is designated to be the initial proposer. When the automaton is the initial responder, its starting state is s_3.

The various transition thresholds have been tuned by the EA to shift the automaton to the proper state(s) for each type of opponent. It is instructive to inspect the automaton's state when (or just before) an agreement is reached.

- Playing against the time-dependent opponents, the automaton is in state s_0 when making a proposal in the last round of the game. This implies (see Fig. 4) that the automaton claims the whole surplus just before the deadline. The time-dependent opponents all accept this take-it-or-leave-it offer and the automaton reaches an optimal deal.

- The automaton reaches an agreement early in the game when playing against the behavior-dependent opponents. The automaton shifts to state s_3 against these opponents and properly makes a large concession on its least important issue (issue no. 1). This offer is accepted by the two behavior-dependent opponents, leading to a mutually beneficial outcome for both parties.

[3]When the number of issues m is small, e.g., when $m = 2$, the optimal strategy against each of the nine opponents can easily be determined by hand. For large values of m (e.g., for $m = 8$), finding the optimal strategy against the behavior-dependent players (see Appendix 1) becomes rather difficult by hand, but is still feasible using linear programming techniques.

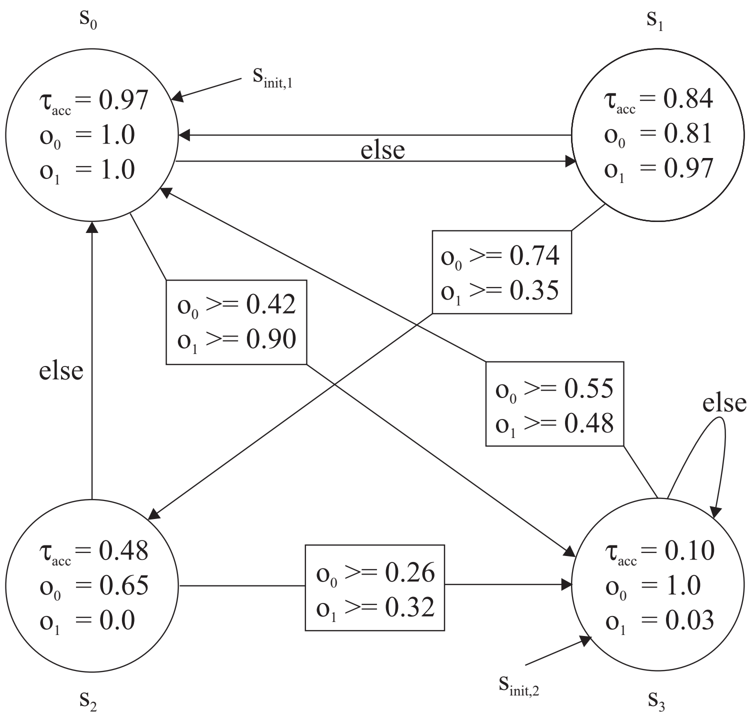

Figure 4: An automaton (with four states) which is capable of negotiating about two issues.

- Playing against "Minimal" (a slowly conceding strategy, see Appendix 1.1), the automaton is in state s_2 when the agreement is reached. The automaton concedes partly on issue no. 0 and fully on issue no. 1 when making a proposal in this state. The partial concession on issue no. 0 is necessary to reach an agreement against the Minimal strategy.

This example shows that very efficient bargaining automata, consisting of only a limited number of states and transitions, can be developed by an adaptive agent. The inspection of the internal structure of the automaton also provides much insight in what constitutes a successful bargaining strategy (against a variety of opponents). In particular, different internal states are used to reach (near-optimal) agreements against different opponent types.

Negotiations over eight issues A complex bargaining problem, involving eight issues and a large number of opponents with different preferences, is considered in this section. The maximum fitness level that can be reached theoretically against the set of opponents considered here (see Appendix 1.2) is ≈ 0.692. Computational experiments show that the fitness of the adaptive agent increases rapidly from 0.265 ± 0.005 in the first generation to 0.54 ± 0.02 after 100 generations. The fitness of the agent then increases (more slowly) to 0.59 ± 0.02 after 500 generations and 0.61 ± 0.02 after 2000 generations.[4] We can thus conclude that adaptive agents continue to perform well in complex bargaining situations with a large number of issues and a variety of opponents with different preferences.

6.2 Dynamic environments

We now consider more open and dynamic negotiation environments. We first study a market place with a increasing number of participants (see Section 6.2). In Section 6.2, we then consider the situation where some participants temporarily leave the market place, while others return.

[4]Means and standard deviations are again measured across 25 experiments with different random seeds.

The computational experiments are performed with similar settings as in Section 6.1. That is, negotiations involve eight issues and preferences vary across the different opponents. The opponents are again drawn from the extended test suite given in Appendix 1.2.

An increasing number of opponents Figure 5 shows the performance of an adaptive agent when the number of opponents increases over time. The adaptive agent has a single opponent in the beginning

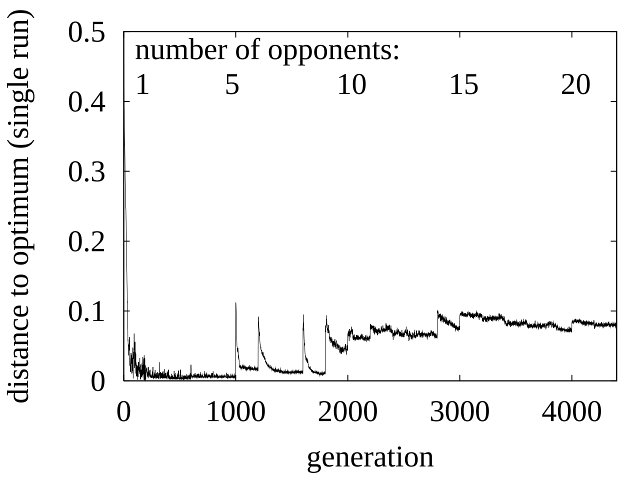

Figure 5: Performance of an adaptive agent when the number of opponents increases over time (an optimal performance of the agent corresponds with the zero-line).

of the experiment. Each 200 generations a new opponent is then added to the group of opponents of the agent until (after 4200 generations) 22 opponents are present.[5] The opponents thus arrive one-by-one, with a time interval of 200 generations.

Figure 5 shows the (absolute) difference between the fitness of the adaptive agent[6] and the highest fitness level that can be reached (theoretically) against the current group of opponents. An optimal performance of the adaptive agent thus corresponds with the zero-line in Fig. 5. Note that the distance to the optimum sometimes increases significantly when a new opponent arrives. The adaptive agent then needs a certain learning period to improve its performance (by developing an efficient strategy against the new opponent). In the long run, when the number of different opponents becomes large, the performance of the adaptive agent remains quite close to optimal.[7] Adaptive agents thus appear to be capable to adjust and improve their strategies when new opponents arrive on the marketplace.

Changing opponent groups We now consider the performance of the adaptive agent when the composition and size of the opponent group fluctuates rapidly over time. This situation can occur in open market places (like the Internet), where the participants may become active at different points in time. Note that the learning problem for the adaptive agent becomes rather difficult in this case. This is due to the fact that the only feedback used by the agent's learning mechanism (the EA) is the performance (fitness) of each strategy against the *current* group of opponents. The information available for the agent's EA thus becomes "noisy" when the group of opponents changes randomly over time. As a result, the difficulty of the learning problem increases significantly.

[5]The opponents are selected at random (without replacement) from the pool of 22 opponents described in Appendix 1.2.

[6]Recall that the fitness of the adaptive agent is defined as the mean fitness across the evolving strategies maintained by the agent.

[7]We verified in additional experiments that these positive results are not due to stochastic factors (we varied the random seeds and the order of appearance of the opponents in these alternative runs).

Figure 6 shows, however, that the adaptive agent continues to operate successfully in dynamic environments. We consider two (extreme) cases in Fig. 6. In the first (static) case, the group of opponents

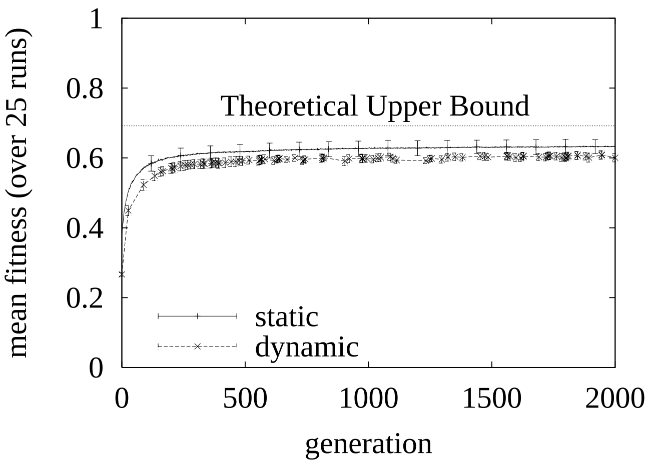

Figure 6: Performance of the adaptive agent when operating in a dynamic environment. The size and composition of the opponent group changes from generation to generation in this case. We show the performance of the adaptive agent when competing against the complete set of (22) opponents. For comparison, we also show the agent's performance in a static environment (the agent is matched with the complete set of opponents in each generation in this case).

does not change over time (this is the same setup as in Section 6.1). In the second (dynamic) case, the size and composition of the opponent group is changing randomly from generation to generation.[8] The agent thus only encounters a subset of all possible opponent types in most generations (on average, the agent is only matched with the full set of opponents once per 22 generations).

Figure 6 shows, however, that the agent's performance against the full set of opponents increases steadily over time in the dynamic environment. So, even though the feedback received by the agent is incomplete and noisy for most generations, high performance levels are reached in the long run. This implies that the application of adaptive agents is not limited to bounded negotiation environments: such agents also appear to be capable of negotiating effectively with changing groups of opponents in open environments.

7. Conclusions

The rapid growth of the Internet currently leads to the development of multi-agent architectures in which artificial agents can negotiate on behalf of their users. Most of today's (prototype) systems for automated negotiations, like Kasbah or Tête-à-Tête, use simple and static negotiation rules. We show, however, that such "fixed" bidding agents can be exploited by more sophisticated "adaptive" agents. These adaptive agents are able to learn strategies which perform (almost) optimally against a variety of fixed opponents. Furthermore, they are able to adapt their strategies online to deal with changing opponents and open environments.

We implement an adaptive agent as a collection of bargaining strategies (represented as finite automata) which is optimized by an evolutionary algorithm (EA). This evolutionary approach has been advocated recently as an innovative and powerful way to develop flexible agents for automated negotiations. An adaptive agent (using an EA) is able to improve its performance by (i) experimenting

[8]The size of the opponent group fluctuates randomly between 1 and 22 in this case. The opponents are again selected at random (without replacement) from the pool of 22 opponents described in Appendix 1.2. All 22 opponents are selected in the first and last generation.

with novel strategies ("mutation"), (ii) combining ("crossing-over") previously-used strategies, and (iii) removing inferior strategies ("selection"). Such an evolutionary approach is especially powerful if the agent participates in frequently occurring micro-transactions (i.e., transactions with a small value). In this case, the agent has sufficient opportunity to experiment with different strategies and learn online from past negotiations.

More in general, research presented in this paper establishes a solid foundation for further research on the application of adaptive agents to Internet trading.

Acknowledgement
This research has been performed within the framework of the project "Autonomous Systems of Trade Agents in E-Commerce", which is funded by the Telematics Institute in the Netherlands.

1. Description of the opponents

In Appendix 1.1, we introduce a test suite of nine opponents (Linear, Conceder, Boulware, LastMinute, Minimal, Careful, HardHead, Tit-fot-tat, and Tit-for-tat*). The composition of an extended test suite (with 22 opponents) is given in Appendix 1.2. The design of most opponents is closely related to the bargaining tactics described in [9]. All opponents use the same criterion to determine whether an offer of the opponent is acceptable. This is the case if the utility of the received offer is greater than that of the counter offer the player would propose at this point. Otherwise, a counter offer is submitted.

1.1 Basic test suite

Time-dependent strategies These strategies do not take the behavior of their opponent into account, but simply base their move on the passage of time. For each issue no. i these strategies demand the same fraction $o_i = 1 - (t/t_{max})^{\frac{1}{\beta}}$, where $t = 0, ..., t_{max}$ is the round number, t_{max} is the last round of the game, and the parameter $\beta \in \mathbb{R}^+$ can be varied to obtain a range of different polynomial functions. When β is small, the strategy concedes slowly initially, but relatively fast as the deadline approaches (the opposite holds for large values of β). For the strategy "Linear" $\beta = 1$, for "Conceder" $\beta = 2$, for "Boulware" $\beta = 0.5$, and for "LastMinute" $\beta \downarrow 0$. (LastMinute concedes the entire surplus to the opponent when $t = t_{max}$.)

Minimal-concession strategies These strategies typically make small concessions on (one on more) issues each time they propose an offer. In the experiments, these strategies start by demanding $o_i = 1$. Each next demand is then reduced with a small amount ϵ ($\epsilon = 0.05$, unless stated otherwise). Note that these strategies do not explicity take the finite deadline of the game into account. Since we set $t_{max} = 9$ in the computer experiments, a concession of at most $5\epsilon = 0.25$ can be made for each issue (when $t = 9$).

A typical example of this tactic is the "Minimal" strategy, who concedes simultaneously on all issues. The strategy "Careful" is very similar, but differs in one respect: Careful starts to concede on its least important issue first. Only when a full concession is made on this issue, concessions will be made on the second least important issue, etc. The "HardHead" strategy always demands the whole surplus for each issue.

Behavior-dependent strategies This is another important class of bargaining strategies. These tactics base their move on the behavior of their opponent during the game. The "Tit-for-tat" strategy exactly mimics the opponent's behavior. When the opponent makes a concession $\delta o_i \equiv o_i(t-2) - o_i(t-1)$ on issue no. i, Tit-for-tat concedes exactly the same amount on the same issue in round t.[9] If the

[9]Tit-for-tat's offer $\vec{o}$ is restricted to the set $[0,1]^m$. Imitating the opponent's behavior can lead to counter offers (by Tit-for-tat) which are located outside this set. To avoid this, each offer o_i (for issue no. i) which exceeds the $[0,1]$ interval is reset to zero (if the offer would become negative) or unity (if the offer would become larger than this value). The remaining discrepancy is stored by Tit-for-Tat. In the next round, the concession of the opponent on issue no. i is then discounted with this amount.

duration of the negotiation does not permit this tactic to be applied (i.e., before the opponent has made two offers), Tit-for-tat employs a Minimal tactic.

Tit-for-tat is not an efficient strategy under all circumstances. For instance, in an integrative bargaining situation both parties need to concede on *different* issues to arrive at outcomes that are mutually beneficial. We therefore include a variant of Tit-for-tat (denoted as "Tit-for-tat*"). This strategy does not concede (in general) on the same issue as the opponent (as Tit-for-tat does) if $m > 1$. In case $m = 2$, Tit-for-tat* first determines the concessions δo_0 and δo_1 of his opponent and then concedes the same amount on the opposite issues (i.e., δo_0 on issue no. 1 and δo_1 on issue no. 0). In case $m = 8$, Tit-for-tat* concedes on the most important issue if the opponent also concedes on his most important issue. A concession of the opponent on his second most important issue then leads to a concession on the second most important issue by Tit-for-tat*, etc., in order of importance of the respective issues.[10]

1.2 Extended test suite

This test suite includes the nine strategies from Appendix 1.1. We also add seven new time-dependent strategies. These strategies are generated by varying the β-parameter which determines their bid function. This parameter is set equal to 0.25, 0.75, 1.25, 1.5, 1.75, 2.5, and 3 for the new opponents. We also add six new minimal-concession strategies. Three of them execute the "Careful" strategy, but with a concession step ϵ equal to 0.025, 0.1, and 0.2. Three "Minimal" strategies, with similar values for ϵ, are added to the opponent pool as well. This yields a total of 22 opponents.

2. Description of the evolutionary algorithm

An evolutionary algorithm (EA) is used to evolve the bargaining strategies of the adaptive agents. This appendix contains a detailed description of our EA implementation. The main components of the EA are visible in Figure 1: the EA starts with an initial population of strategies, generates new "offspring" strategies by *mutation* and *recombination*, and *selects* the best strategies for the next iteration ("generation"). The EA then repeats this iterative process to generate highly-fit strategies in the course of time.

In the remainder of this appendix we first describe the genetic representation of the bargaining strategies. We then discuss the main components of the EA.

Strategy encoding A bargaining strategy of the adaptive agent, which consists of a finite automaton (see Section 4), is represented as an ordered sequence of genes in our evolutionary model.[11] All genes together constitute the chromosome. The typical structure of such a chromosome is shown in Fig. 7. Figure 7a shows the global structure of a chromosome which encodes a finite automaton with q internal states. States are indexed by the numbers $0, ..., q-1$. From left to right, the chromosome consists of several packets of genes containing (i) a pointer to the initial state for the initial proposer and responder ($s_{init,1}$ and $s_{init,2}$) and (ii) the description of all internal states $s_0, ..., s_{q-1}$.

The structure of a single state is shown in detail in Fig. 7b. One state contains the consecutive description of the acceptance threshold τ_{acc}, the (m-dimensional) offer $\vec{o}$, the components of the transition threshold $\vec{\tau_{tr}}$, and the components of the transition function $\vec{f}$ (being f_0 and f_1).

The chromosomes are represented by binary-coded strings (see Fig. 7c). We represent each value on the chromosome (e.g., for an offer, threshold, etc.) with a sequence of 5 bits. The interval between two successive values is thus equal to 1/31 (because all offers and thresholds are restricted to the unit interval). Since we use automata with 8 states, the pointers to states are encoded with 3 bits. The chromosomes in the initial population consist of randomly initialized bitstrings.

[10]We thus assume that Tit-for-tat* knows the ranking of the opponent's weights (for the different issues). The adaptive agent receives no information about the opponents' preferences in the computational experiments.

[11]Our approach is inspired by Miller's work [10] on the iterated prisoner's dilemma. Miller encoded the game-playing automata as binary strings and adapted them using an EA.

Structure of an automaton (with q states):

(a) $s_{init,1}$ $s_{init,2}$ state s_0 state s_1 ... state s_{q-2} state s_{q-1}

(b) τ_{acc} $o_0 \dots o_{m-1}$ $\tau_{tr,0} \dots \tau_{tr,m-1}$ f_0 f_1

(c) 010011100101100101101011001110011011000100...

Figure 7: Genetic representation of an automaton with q states. (a) The global structure of the chromosome. The initial states for the initial proposer and responder ($s_{init,1}$ and $s_{init,2}$) refer to one of the q machine states $s_0 - s_{q-1}$. (b) Structure of a single internal state. (c) Example of a binary-coded automaton.

Mutation and recombination Binary-coded individuals are typically mutated with a small probability in EAs. We set this probability equal to 0.01 (per bit). In addition, two-point crossover (with a crossover probability of 0.6) is used. We follow the common practice to use a Gray-code interpretation of the bitstring segments. Additional experiments show that our settings for the mutation and crossover probabilities are chosen properly.

Selection Selection is performed by ranking all μ parents and their λ offspring, and transferring the μ top-performing strategies to the next generation. We set $\mu = \lambda = 100$. This selection scheme is formally denoted as $(\mu + \lambda)$ selection [2]. We also investigated the performance of various alternative selection methods (e.g., fitness proportional and tournament selection). Results obtained with $(\mu + \lambda)$ selection are superior, however, to results obtained with these alternative selection schemes.

References

1. D. Ashlock. GP-automata for dividing the dollar. In *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pages 18–26, Stanford University, CA, USA, 13-16 July 1997. Morgan Kaufmann.
2. T. Bäck. *Evolutionary Algorithms in Theory and Practice.* Oxford University Press, New York and Oxford, 1996.
3. K. Binmore, M. Piccione, and L. Samuelson. Evolutionary stability in alternating-offers bargaining games. *Journal of Economic Theory*, 80:257–291, 1998.
4. A. Chavez and P. Maes. Kasbah: An agent marketplace for buying and selling goods. In *Proceedings of the 1st International Conference on the Practical Appication of Intelligent Agents and Multi-Agent Technology*, London, April 1996.
5. G. Dworman, S.O. Kimbrough, and J.D. Laing. Bargaining by artificial agents in two coalition games: A study in genetic programming for electronic commerce. In *Genetic Programming 1996: Proceedings of the First Annual Conference*, pages 54–62, Stanford University, CA, USA, 28–31 July 1996. MIT Press.
6. E.H. Gerding, D.D.B. van Bragt, and J.A. La Poutré. Multi-issue negotiation processes by evolutionary simulation: Validation and social extensions. Technical Report SEN-R0024, CWI, Amsterdam, 2000. Presented at the Workshop on Complex Behavior in Economics, Aix-en-Provence, France, May 4-6, 2000.
7. J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, Reading, MA, 1979.
8. P. Maes, R.H. Guttman, and A.G. Moukas. Agents that buy and sell. *Communications of the ACM*, 42(3):81–91, 1999.
9. N. Matos, C. Sierra, and N.R. Jennings. Determining successful negotiation strategies: An evolutionary approach. In *Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS-98)*, pages 182–189, Paris, France, 1998.
10. J.H. Miller. The coevolution of automata in the repeated prisoner's dilemma. *Journal of Economic Behavior and Organization*, 29(1):87–112, 1996.
11. M. Mitchell. *An Introduction to Genetic Algorithms.* The MIT Press, Cambridge, MA, 1996.
12. J.R. Oliver. A machine learning approach to automated negotiation and prospects for electronic

commerce. *Journal of Management Information Systems*, 13(3):83–112, 1996.

13. M.J. Osborne and A. Rubinstein. *Bargaining and Markets*. Academic Press, San Diego, CA, 1990.
14. D. D. B. van Bragt and J. A. La Poutré. Co-evolving automata negotiate with a variety of opponents. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC-2002)*, Honolulu, HI, USA, May 2002.
15. D.D.B. van Bragt, C.H.M. van Kemenade, and J.A. La Poutré. The influence of evolutionary selection schemes on the iterated prisoner's dilemma. *Computational Economics*, 17(2/3):253–263, 2001.
16. G. Weiss, editor. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. The MIT Press, Cambridge, MA, 1999.