Efficient agents for Cliff-Edge environments with a large set of decision options

Ron Katz, Sarit Kraus
Bar-Ilan University, Ramat-Gan 52900, Israel, sarit@cs.biu.ac.il

ABSTRACT

This paper proposes an efficient agent for competing in Cliff Edge (CE) environments, such as sealed-bid auctions, dynamic pricing and the ultimatum game. The agent competes in one-shot CE interactions repeatedly, each time against a different human opponent, and its performance is evaluated based on all the interactions in which it participates. The agent, which learns the general pattern of the population's behavior, does not apply any examples of previous interactions in the environment , neither of other competitors nor its own. We propose a generic approach which competes in different CE environments under the same configuration, with no knowledge about the specific rules of each environment. The underlying mechanism of the proposed agent is a new meta-algorithm, Deviated Virtual Learning (DVL), which extends existing methods to efficiently cope with environments comprising a large number of optional decisions at each decision point. Experiments comparing the performance of the proposed algorithm with algorithms taken from the literature, as well as another intuitive meta-algorithm, reveal a significant superiority of the former in the average payoff and stability. In addition, the agent performed better than human competitors executing the same task.

1. INTRODUCTION

In many games and economic interactions agents are challenged by the strive of maximizing their profits while preventing the entire deal from falling through. Consider, for example, an agent in an online Web-store which needs to determine the price of a good: Increasing the price would increase its profits as long as the price it charges is not higher than the price the consumer is willing to pay. A little greedier price would make it lose the whole deal [11]. Similarly, a bidder in a sealed-bid first-price auction [3] attempts to bid exactly the amount that is slightly higher than others' biddings. The same goes for rational proposer in the ultimatum game (UG) [9], which tries to offer exactly the amount which equals to the acceptance threshold of his opponent. Higher offers would decrease self profits, while lower offers would yield no profits at all. This situation is somehow similar to a person standing on the edge of a cliff, trying to see the panoramic view. As closer as he gets to the cliff edge, the better he can see the view. However, a one step too many may cause him to fall off the cliff. Hence, this set of interactions and games are refered as Cliff-Edge (CE) interactions.

We focus on one-shot CE interactions which are repeatedly competed with different opponents. Such repeated interactions occur, for example in the important application of dynamic pricing, i.e. periodically changing prices according to the reactions of consumers that change over time. This task becomes even more common in the E-commerce world, where communication with the consumers is restricted [11, 4]. Repeated periodically sealed-bid auctions with the same goods are very popular, especially via the internet (see [3, 5]). Similarly, the sequential version of UG with different opponents is under thorough investigation [13, 2, 1]. Thus, designing efficient agents for repeated CE environments is very important for the daily commercial world.

Moreover, our research deals mainly with interactions between humans and automated agents. This differs from conventional environments of pure computerized agents, where most competitors have large amounts of computational power and act as rational negotiators. In particular, when interacting with humans, the theoretical equilibrium strategy is not necessarily the optimal strategy since human subjects, inherently rational as well as computational bounded, commonly do not behave according to perfect equilibrium. Thus, this study joins an important growing research field of interactions between humans and automated agents (e.g. [6, 12]).

Previous studies of automated agents interacting with humans have been concerned mainly with agents who interact repeatedly with a single human opponent (e.g. [12]). The general approach for such a problem is to model the opponent's behavior, and to adjust a strategy which optimally reacts to the opponent's predicted reaction. In the current study, however, a specific opponent modeling is not relevant, since the agent interacts with each opponent only once. Alternatively, agents who interact with a series of different people, should learn the general pattern of the opponent population.

A possible approach to efficiently adjust to an unknown opponent population, is to simply observe previous samples of interactions within this population. Several studies have successfully used this approach in order to develop automated agents that can successfully compete with human opponents (e.g. [6]). This approach however, requires a large number of historical examples of human behavior. Moreover, the option of using a historical database is not always possible. For example, in a sealed-bid auction, a bidder usually cannot obtain information about the bids which were offered in previous similar auctions. Therefore, we propose a mechanism to develop an agent which does not depend on examples of previous interactions. Our agent learns while interacting with others, and its performance is evaluated from the first interaction.

Moreover, this study focuses on settings where in each interaction the agent is required to choose an action from a large set of possible options. Most of the previous studies on CE environments focus on settings in which an agent has to choose from at most 10 options during each interaction [13, 11, 1]. A large scale of option settings more realistically reflects commercial environments, where the number of decision options is commonly much higher than 10 (e.g. [4, 5]). Unfortunately, as we show in this study, the performance of the existing algorithms choosing from a large set of options such as 100 options is insufficient. The existence of a large set of options demands a construction of a fast and efficient screening procedure. Furthermore, the importance of quickness is emphasized, since we examine short term interactions with only several dozens of opponents.

In this paper we propose a meta-algorithm which extends existing algorithms for efficiently competing in multi-interaction CE environments that consist of a large set of decision options. The general idea of the proposed algorithm is to reinforce options which are slightly greedier than the currently considered optimal option. Thus, in the conflict between exploration and exploitation of current information, we compromise by a small deviation from the current optimal option. A broad description of this method, named Deviated Virtual Learning (DVL), is presented below.

Several approaches for automatically competing in CE environments have been previously proposed [17, 15, 11]. Those approaches are broadly described below. In this paper, we experimentally show that the extension of the DVL meta-algorithm over the basic Reinforcement Learning (RL) algorithm [14], performs significantly better than previous algorithms when competing against human opponents in various CE environments. Moreover, the DVL extension of the RL algorithm (DVRL) is shown to perform better than another intuitive meta-algorithm named Segment-Based, which appears here for the first time. We also found that the performance of the DVRL algorithm is better than human competitors who face the same task.

It is noteworthy that the algorithms being discussed here are all generic and suitable to various CE environments, without knowing the specific rules of each environment. Thus, the agent is not aware of whether it competes in an auction or in a UG. Moreover, in all the simulations presented hereinafter, each algorithm was run with a fixed configuration setting, that was not changed from environment to environment.

In the next section, we formally describe the CE environment. In section 3 we present the proposed DVL meta-algorithm. In section 4 we survey other relevant algorithms which appear in previous studies, and present the Segment-based meta-algorithm. We then compare the performance of the DVRL algorithm with the performance of other relevant algorithms, and analyze the results. We conclude and present directions for future work in section 5.

2. THE CE ENVIRONMENTS

The general pattern of one-shot CE interactions considers an agent required to choose an offer *i*, which is an integer $0 \leq i \leq N$, where *N* is the maximal optional choice. Then a positive reward *r* corresponding to the offer *i* is determined, depending on whether the offer passed a certain acceptance threshold set by the opponent (in auctions we refer to the acceptance threshold of the auctioneer, which is the second highest bid proposed). Since the CE set includes various environments, we detail the model of each of the three environments mentioned above: auction, UG and pricing. Other CE environments can be similarly modeled.

- In the sealed-bid first-price auction model an amount *N* is auctioned [1]. The agent is required to choose its bid which is an integer $i, 0 \leq i \leq N$. Given the bid, a reward *r* is determined according to the highest bid τ, among all the bidders (two or more) in the current auction. If $\tau \leq i, r = N - i$ (the amount gained subtracted by the bidding amount), otherwise r=0 [2]. In the all-pay version, where all the bidders must pay their bids, even had they not won the auction [10]: if $\tau \leq i, r = N - i$ otherwise $r = -i$.

- In the UG, the agent should divide an amount of *N* between itself and its opponent. It is required to choose an integer $i, 0 \leq i \leq N$, which is the amount proposed to the opponent. The reward *r* is determined according to the opponent's acceptance threshold τ . If $\tau \leq i, r = N - i$ otherwise *r*=0.

- In the pricing task, an agent is required to select a price *i* for an item being sold to a certain consumer (*i* actually represents the profit of the seller, beyond the cost price). In our model, *i* is an integer $0 \leq i \leq N$, where *N* is the maximal reasonable price of the item. After the decision on the price, a reward, *r*, is determined according to the current consumer's acceptance threshold τ. If $\tau \geq i, r = i$ otherwise *r*=0.

In this paper, which considers environments with a large set of decision options, we set *N* to be 100.

To demonstrate the challenge of a competitor in CE environments, let *S(i)* be the corresponding reward of offer *i* if the move succeeds (if the offer is accepted by the consumer in the price setting task, higher than the other bids in the case of an auction, or accepted by the responder in the UG), let *F(i)* be the corresponding reward of offer *i* if the move fails, and let *P(i)* be the probability that an offer *i* would succeed. An efficient agent interacting in multi-interaction CE environments with different opponents must find the optimal offer *i* for the opponent population that maximizes the accumulative utility function:

$$U(i) = S(i)*P(i) + F(i)*(1-P(i))$$

Obviously, there is a trade-off between *S* and *P*: choosing an offer which increases the expected reward, *S*, decreases the probability of success, *P*, and vice versa. Similarly, in the all-pay auction, which is the only case where *F(i)* values are negative and are not equal to zero, choosing an offer which decreases the expected loss, *F*, increases the probability of failure, *(1-P)*.

The naive approach for such problems would be to apply the Monte-Carlo (MC) method (see [14] chapter 5). According to this method each optional proposal or bidding amount is examined many times, in a serial or random order, and finally (the later the better) a model for success probabilities (P) distribution of the opponent population is constructed. However, the learning process in this approach is extremely time consuming. Moreover, a large number of actions made by the MC method are totally irrational. An efficient agent should quickly realize, for example, that most responders in UG would accept a share of 50 percent, and thus a proposal of more than 50 percent, is not advised. Therefore, we would rather use a less naive attitude, in which directed exploration is conducted.

Nevertheless, for a very large number of interactions, the MC method may provide a more precise model of the real success probabilities distribution in the population, than algorithms which use directed exploration. However, in this study we are more concerned with finding fast and efficient approaches for the first several dozens

[1] In order to evade considerations of value estimations [5] the item auctioned is an amount of money.

[2] For reasons of compatibility with other environment models, equal bidding amounts yield a gain for the learning agent.

of opponents. Handling the initial interactions, when almost nothing is known about the environment is obviously the most difficult stage. Moreover, considering a larger number of interactions, the severity of the accuracy problem caused by directed exploration, could be reduced by combining a sophisticated algorithm with a naive one. A method such as MC, may replace a more sophisticated algorithm in the progressive interactions, based on previous interactions conducted by the latter.

3. THE PROPOSED APPROACH

In this section we present a detailed description of the proposed meta-algorithm for competing in CE environments. As a meta-algorithm, it extends basic algorithms and improves their performance. We assume that the basic algorithms select their actions according to an evaluation of the expected utility that each decision option would yield, provided that it is chosen. The evaluation of the expected utilities is determined according to the results of previous interactions. Hereinafter, we will refer to the database which stores the evaluations of each decision option as Q-vector [3]. Each algorithm includes its own UPDATE procedure, which determines how to update the Q-vector after observing the successfulness of the latest action. It should be noted that the Q-values stored in those Q-vectors, do not necessarily express the real evaluations of the expected utility, but rather the relative profitability of choosing each option. In addition, we assume each algorithm includes its own SELECT procedure, which determines how to select the next action (apparently according to the current Q-vector). The SELECT procedure moves between exploiting the evaluations stored in the current Q-vector and exploring options with lower current Q-values, in order to improve the correctness of the Q-vector's evaluations. The first action is chosen randomly in all the algorithms presented in this paper.

A reasonable approach for the UPDATE procedure in CE environments is *Virtual Learning* (VL) [16]. According to the VL principle, the agent in UG, for example, treats all the offers (not actually proposed) higher than a successful offer as if they were (virtually) successfully proposed as well. Similarly, it considers all the offers lower than an unsuccessful offer as unsuccessfully proposed. The rationale behind this principle is that the higher the amount proposed to the opponent - the higher the probability of the proposal being accepted [4]. Thus, we should reinforce the Q-values of all the offers higher than the actual offer, provided it was successful, and we should reduce the Q-values of all the offers lower than the actual offer, provided it was unsuccessful. A crucial problem with this approach is an *information asymmetry*[16], which causes an intensified reinforcement of Q-values higher than the optimal option's Q-value. The reason for this phenomenon, as demonstrated in [16], is that whenever the optimal offer is successfully chosen, offers which are higher than the optimal offer are reinforced as well. Since those offers are reinforced whenever offers which are higher than the optimal offer are successfully chosen as well, their Q-value may be higher than that of the optimal offer.

In order to overcome this information asymmetry problem, we propose the *Deviated Virtual Learning* principle. This principle can be implemented in various basic algorithms, thus produce a general meta-algorithm, as presented in **Algorithm 1**. According to the DVL principle, we should deviate from the strict rationale underlying the VL principle, and extend the range of offers updated after each interaction. Thus, after a successful interaction, we would raise the Q-values of all the offers higher than the actual offer, as well as a few offers under the actual offer, as described in line 8 in **Algorithm 1**. Similarly, after an offer has failed, we would reduce the Q-values of all the offers lower than the actual offer, as well as a few offers above the actual offer, as described in line 6 [5].

Algorithm 1 THE DVL META-ALGORITHM

Notation: α, β are two integers $0 \leq \alpha < \beta \ll$ N, (where N is the upper bound of possible offers), which denote the deviation rate. The values α and β gradually decrease during the learning process.

```
1: For j=0 to N, Do Q(j)=1
2: For each interaction, Do
3:    Select offer i according to a SELECT procedure
4:    Observing opponent's move, calculate reward
5:    if offer i has failed then
6:       For j=0 to (i + $\alpha$), Do
            reduce Q(j) according to an UPDATE procedure
7:    else
8:       For j=(i - $\beta$) to N, Do raise Q(j) according to an UPDATE procedure
```

As mentioned above, the DVL meta-algorithm can extend any basic algorithm, according to its SELECT and UPDATE procedures. However, we claim that the best algorithm for human environments, as we will demonstrate, is the DVL extension of the Reinforcement Learning [14] algorithm (DVRL), as presented in **Algorithm 2**. The SELECT procedure of the RL version we have used, was to select the offer with the current maximal Q-value (line 3). The UPDATE procedure was to divide the accumulated reward of each offer by the number of previous interactions where offer i was actually or virtually (according to the DVL principle) proposed (lines 6 and 8).

Algorithm 2 THE DVRL ALGORITHM

Notation: n(j) denotes the number of previous interactions where offer j was actually or virtually proposed, s(j) is the corresponding reward for a successful offer j and f(j) is the corresponding reward for offer j when it fails. α, β denote the deviation rate.
We present here the configuration used in our environment. However, the initial values of parameters N, α and β (line 1) as well as the update function of α and β (line 10) can be adjusted to other environments as well.

```
1: N=100, $\alpha$=10, $\beta$=15, t=0   For j=0 to N, Do Q(j)=1, n(j)=0
2: For each interaction t, Do
3:    offer i=$\arg\max_j Q(j)$
4:    Observing opponent's move, calculate reward
5:    if offer i has failed then
6:       For j=0 to (i + $\alpha$), Do   n(j)=n(j)+1,  $Q(j) = \frac{Q(j)(n(j)-1)+f(j)}{n(j)}$
7:    else
8:       For j=(i - $\beta$) to N, Do   n(j)=n(j)+1,  $Q(j) = \frac{Q(j)(n(j)-1)+s(j)}{n(j)}$
9:    t = t+1
10:   $\alpha = \frac{10}{\lfloor t/10 \rfloor + 1}$   $\beta = \frac{15}{\lfloor t/10 \rfloor + 1}$
```

It is important to note that in this study DVRL was run in all the environments with the same configuration of α and β as detailed in **Algorithm 2** (lines 1 and 10).

There are two additional motivations for the deviation principle underlying the DVL approach. First, from the statistical aspect it

[3]It is noteworthy that despite the name, this paper does not concern the Q-learning algorithm, which handles situations in which the state may be changed at each interaction (see [14] chapter 6.5).

[4] The same rationale stands behind the auctions environment. However, in the pricing environment the opposite is true: all the offers lower than a successful offer are considered successful, since the lower the price of an item the higher the probability of it being purchased by the consumer.

[5]It is worth noting that all the VL-based algorithms in this paper are presented in a version suitable for auctions and for the UG. In the pricing environment, where increasing the price decreases the acceptance probability, the update ranges in lines 6 and 8 should be swapped.

is not necessarily a mistake to consider an offer which is slightly lower than a successful offer, as successful as well. Using UG terminology, it is quite reasonable to assume that if proposal *i* was accepted (rejected) by an opponent, she would have also accept (reject) a slightly lower (higher) proposal. The chance that the proposal would exactly hit the acceptance threshold of the opponent is not high, especially for a large set of options. Second, and even more important, the deviation principle actually outlines a direction of the optimal solution searching, rather than the random trial-and-error approach that underlies conventional methods, such as RL. A DVRL agent that offered 70% of the cake (N) to his UG opponent in the first interaction, and his offer was accepted, would offer 55% (according to our configuration of β) in the next interaction. The agent continues to decrease the offer until it is rejected. A similar pattern of this directed searching was observed in human learning, and is compatible with the directional learning theory (see [8]). According to this theory, if an offer *i* is rejected at interaction *t*, then at interaction *t+1* the proposer will offer his opponent a higher offer, while if offer *i* is accepted at interaction *t*, then at interaction *t+1* the offer will be decreased. This simple method, which was simulated by [2] without yielding any impressive performance, lacks all the data experienced before the previous interaction.[6] In our algorithm, on the other hand, all the previous interactions are taken into account, and the model comes closer to the real distribution of the opponents' population, during the learning process. This is the reason for the gradual decrease of α and β values. Another advantage of our approach over the naive directional learning is achieved by setting $\alpha < \beta$, asymmetrically. This property enables fast converging to the optimal solution, by speedily decreasing the proposal amount, and slowly increasing it to the optimal amount.

4. EXPERIMENTAL DESIGN AND ANALYSIS

In this section we examine the performance of the proposed DVRL algorithm competing in various CE environments with human opponents, and compare it to other algorithms' performance. A broad survey of the algorithms examined is presented, followed by a description of the experiment process and the results. After analyzing the results, we compare the performance of DVRL to human negotiators' performance. At the end of this section, we briefly discuss the situation of a small set of decision options' environments.

4.1 Comparative Algorithms description

Here we survey several algorithms which can be used in repeated CE environments. Subsections 4.1.1-4.1.4 describe previously published algorithms, while in 4.1.5 we present a new intuitive meta-algorithm for a large set of options environments.

4.1.1 Roth and Erev's Algorithm (R&E)

Concerning the UG, Roth and Erev [13] designed human proposer model which is based upon basic RL [14]. The general idea of the RL method is that each offer is chosen with a probability according to its current Q-value - the larger an offer's Q-value, the higher the probability it will be chosen. After each interaction, provided that the latest offer has succeeded, its probability to be chosen again is increased by increasing its Q-value by the reward this offer yields. Roth and Erev showed that despite the fact that UG is characterized by having a significant gap between empirical human behavior and subgame perfect equilibrium predictions, the RL algorithm with a few additional modifications can successfully model typical human empirical behavior in UG. The most noticeable modification, which was based on the Persistent Local Environment principle, used a "generalization" parameter which prevents the probability of an option from going to zero if it is adjacent to a successful option. The criterion of adjacency was configured in [13] to be +-1, and thus, if an offer of 4 out of 10 is accepted by the opponent, the agent would reinforce the Q-values of offers 3,4 and 5. Their assumption, which is valid in all of the CE environments, was that an adjacent of an optimal solution cannot be an extremely bad solution. The detailed algorithm can be found in [13].

4.1.2 Modified version of Zhong, Wu and Kimbrough's (ZWK) RL algorithm

Zhong et al.[17] developed an agent which played UG against various simulated opponents, using a version of RL which is different from R&E's. The UG agent designed by these authors used an ϵ-greedy selection procedure, in which the agent selects, with a probability of (1-ϵ), the offer with the current maximal Q-value, and with a probability of ϵ it uniformly selects an offer for exploration purposes. The advantage of this selection procedure over a selection according to the Q-value's distribution as in R&E, is that it mainly chooses the best option. The disadvantage is that the exploration is not weighted by the estimation of expected utility considerations, as in R&E, but is rather completely random. In order to decrease the effect of this disadvantage, we have modified the selection procedure in our experiment, to take into account the generalization principle of R&E. Thus, when not selecting the offer with the current maximal Q-value, our ZWK algorithm selects an offer giving higher priority to offers which are adjacent to the offer with the current maximal Q-value (**Algorithm 3**, state 4), rather than selecting them uniformly. In addition, since the exploration is much more important in the initial interactions, we decreased the probability for random selection gradually during the learning process. These two modifications were examined and were found to be significantly efficient in our environment. The UPDATE procedure in ZWK's algorithm was also different from that of R&E, by taking into account the previous failures of offer *i*, rather than merely the positive previous rewards (line 6).

Algorithm 3 THE ZWK ALGORITHM

Notation: ϵ denotes the probability for random selection, γ is the probability for selecting an offer which is adjacent to the offer with the current maximal Q-value ($\epsilon + \gamma < 1$), and δ determines the adjacency range ($\delta \ll$ N). The values ϵ and γ are gradually decreased during the learning process.

```
1: For j=0 to N, Do Q(j)=1, n(j)=0
2: For each interaction, Do
3:    m = $\arg\max_j Q(j)$
4:    With a probability of (1- $\epsilon$- $\gamma$), offer i = m
      With a probability of $\gamma$, select offer i uniformly, for i's such that
        max(0, m - $\delta$) $\leq$ i $\leq$ min(N, m + $\delta$)
      With a probability of $\epsilon$, select offer i uniformly, for i's such that
        0 $\leq$ i $\leq$ N
5:    Observing opponent's move, calculate reward r
6:    n(i)=n(i)+1,   Q(i) = $\frac{Q(i)(n(i)-1)+r}{n(i)}$
```

4.1.3 Virtual Reinforcement learning (VRL)

Despite the information asymmetry in VL found by [16], Todd and Borgers [15], have shown that the addition of a virtual learning

[6]Simulations of the directional learning method in our environment also does not yield an impressive performance. A similar approach, namely the Derivative-Following strategy, was proposed for the dynamic pricing problem [4]. However, each round (one commerce day) was considered as a set of several interactions with a number of opponents, which is totally summed up by thousands of interactions.

mechanism to the RL algorithm when playing UG, may be very efficient. However, the efficiency is dependent on several configurations of the learning procedure, such as the selection procedure (R&E's or ZWK's) and the Q-values update procedure (reinforcement of a successful offer by a unit payoff, or by the real amount gained, or rather by the virtual amounts the agent would keep if he hypothetically offered different proposals).

After examining the performance of virtual RL with all the optional configurations suggested by Todd and Borgers, in **Algorithm 4** we present the version that yields the best performance in our environment. This algorithm includes the same SELECT procedure as in ZWK.

Algorithm 4 THE VRL ALGORITHM

Notation: s(j) denotes the corresponding reward for a successful offer j and f(j) is the corresponding reward for offer j when it fails.

```
1-5:  As in **Algorithm 3**
6: if offer i has failed then
7:    For j=0 to i, Do    n(j)=n(j)+1,   $Q(j) = \frac{Q(j)(n(j)-1)+f(j)}{n(j)}$
8: else
9:    For j=i to N, Do    n(j)=n(j)+1,   $Q(j) = \frac{Q(j)(n(j)-1)+s(j)}{n(j)}$
```

4.1.4 *Gittins' index strategy*

Some researchers have observed CE environments as a special case of the multi-armed bandit problem [11, 1]. In this problem every period a decision maker has to decide on which one of *n* slot machines he wants to play, given that each machine has different gain probabilities (unknown a priori). These researchers used Gittins' index strategy [7], an optimal strategy for the multi-armed bandit problem, to model the behavior of proposers in UG and of sellers in the pricing problem, considering each optional proposal or price as an arm. The Gittins' strategy was applied to CE environments by multiplying the Bernoulli reward process index value of each option (see [7] p. 222) by its reward. However, these CE problems are different from the classical multi-armed bandit problem since the arms are independent, while in CE environments the success probability of an offer is gradually influenced by the size of the offer. This difference may become more critical when there is a large number of options rather than the 10 optional arms problems considered in these papers. Thus, Brenner and Vriend [2] tried to adjust to the interdependent arms in UG by adding the VL mechanism to the Gittins' strategy. In the following section we present the performance of both the basic Gittins' strategy (GS) (as in [11, 1]) and the virtual Gittins' strategy (VG) (as in [2]) in our environment.

4.1.5 *Segment-based meta-algorithm*

This algorithm, suggested here for the first time is an intuitive meta-algorithm which can extend any of the former basic methods. This algorithm is specifically designed for environments with a large number of decision options. The idea of the segment-based approach is to divide all the options into segments, and to activate the learning method in the initial interactions on these segments, rather than on specific discrete options (**Algorithm 5**, state 3). After several interactions the resolution can be increased by focusing on smaller segments. This process continues gradually towards the last segmentation hierarchy with the smallest segments, i.e. specific discrete options, where final fine tuning is performed (end of state 3). This algorithm is based on the assumption of locality, i.e. adjacent options yield similar average profits. The advantage of this approach is the gradual filtering of the optimal solution. A common problem in conventional RL, for instance, is that an option might have a high Q-value in a progressive stage of the learning process, although it is far from the optimal option. With the proposed meta-algorithm, this situation is prevented already in the preliminary stages of learning, by weakening the Q-value of the entire range surrounding this non-beneficial solution.

Algorithm 5 THE SEGMENT-BASED META-ALGORITHM

Notation: H denotes the number of segmentation hierarchies configured by the user according to the size of the options set (setting this parameter, the order of magnitude of the expected number of interactions must be considered), S_h is the size of each segment in the h^{th} segmentation hierarchy, Q_h are the Q-values vectors corresponding to the h^{th} segmentation hierarchy, and T_h is the serial number of the last interaction of the h^{th} segmentation hierarchy.

```
1: For j=0 to $\lceil \frac{N}{S_h} \rceil$, Do   For h=1 to H, Do  $Q_h$(j)=1,
2: For each interaction, Do
3:    For interactions $0 - T_1$ :   c = 1
         Select segment j from the 1st segmentation hierarchy according to
           a SELECT procedure, using vector $Q_1$
         Select offer i uniformly from segment j
      For interactions $T_{x-1} - T_x$ :  c = x
         Select segment j from the $x^{th}$ segmentation hierarchy according to
           a SELECT procedure, using vector $Q_x$
         Select offer i uniformly from segment j
   ...
      For interactions $T_{H-1} - T_H$ :  c = H
         Select offer i according to a SELECT procedure, using vector $Q_H$
4:    Observing opponent's move, calculate reward
5:    For h=c to H, Do update $Q_h$ according to an UPDATE procedure
```

4.2 Experiment Description

In order to evaluate the performance of the proposed algorithm, and to compare it with other algorithms, we've experimentally examined agents interacting with human opponents in 4 different CE environments, as follows:

1. UG, where the players had to divide 100 new Israeli Shekels (NIS, where 1 U.S. $ $\approx$ 4.5 NIS), i.e. N=100.
2. A variant of UG in which the responder determines only whether the proposer gets his share. The responder himself always receives his own allocation [9]. In this version the responder's decision is influenced by fairness and vindictiveness considerations, rather than profitability as in the original UG version. Here also N=100.
3. Sealed-bid 2-bidders auction. The auction was for 100 NIS.
4. All-pay sealed-bid auction for 100 NIS, where all the bidders must pay their biddings[10]. In this version risk taking considerations is added to the bidder's decision.

The examination of different environments which activate different personality characteristics of human opponents, guarantees generality of the results. Dynamic pricing was not experimentally examined, since no specific good can be unquestionably desired by all of the participants in the experiment. The following algorithms were examined:

1. R&E, as described in section 4.1.1.
2. ZWK, as specified in section 4.1.2.
3. Virtual RL (VRL), as specified in section 4.1.3.
4. Basic Gittins' index strategy (GS), as described in section 4.1.4.
5. Virtual Gittins' index strategy (VG), as described in 4.1.4.
6. Segment-based extension of the ZWK version of RL (SZWK).
7. Segment-based extension of VRL. (SVRL).
8. DVRL, as specified in section 3.
9. Segment-based extension of VG (SVG).
10. Deviated virtual extension of GS (DVG).

The first five algorithms were suggested by others and adapted to our environment as detailed above, while the last five algorithms are based on the two meta-algorithms we developed [7].

It is important to mention that each algorithm was run with fixed parameter configurations (such as the ϵ value's decreasing rate, and deviation rates α, β in the DVL algorithms) for all the different environments. Though specific configurations for each environment could yield better performance, we preferred the generality of the algorithms over a variety of CE environments, ensuring that no environment specific characterization would be used.

In the first experiment human participants were used as responders in the UG games, and as bidders in the auctions. Each person participated once in each of the 4 environments. The automated agents, on the other hand, interacted serially against different human opponents. Evaluating agents that were designed for human involved environments by examining their performance with real human data is necessary. As mentioned above, human competitors do not obey subgame perfect equilibrium, and thus their behavior cannot be a priori simulated. Additionally, human behavior cannot be accurately statistically modeled, especially in small populations as in this case. Another benefit from empirical experiments is the ability to provide a concrete algorithm, namely DVRL (with a concrete configuration of parameters), which successfully competed against human opponents. Thus, this agent can be immediately applied in real applications, at least as a starting point.

In the first stage of the experiment we surveyed 34 students (17 males and 17 females) participating in four CE environments. The participants were students at Bar Ilan University, aged 20-28 (average 23.75), who were not experts in negotiation strategies nor in economic theories directly relevant to the experiment (e.g. game theory, decision theory). Each of the participants, who sat in an isolated room at a computer work-station, interacted in the 4 CE environments, one after the other.

In the auctions the participants were required to propose a bid, which could be any integer from 0 to 100 NIS. The winner gains a virtual 100 NIS. In the ultimatum games, where 100 NIS was the full amount to be shared, the participants were required to respond to various proposals, which were artificially provided in order to determine the minimal acceptance amount of the players [8](though the participants were told that the proposals were provided by other people). Data from a few participants who were inconsistent in their decisions was excluded. At the end of the experiment, each participant was paid between 15 to 30 NIS, proportionally to her earnings in the interactions she participated in.

After extracting the bids from people in the auctions, as well as their acceptance thresholds in the ultimatum games, we constructed sets of opponents' reactions for each environment. The sizes of the sets were 34 for the auctions, 28 for the UG and 27 for the UG variant, due to the exclusion of inconsistent players. At this stage we examined the performance of each of the ten algorithms detailed above, which were run serially against the sets of opponents' thresholds. Thus, each algorithm had one interaction with each of the human opponents in each of the four environments, without knowing in advance the number of interactions. Since there is importance to the order of the opponents, we constructed 200 random permutations of the human decisions series, for each environment, and compared the average payoffs of the different algorithms for each permutation. In addition, these algorithms were run against an artificial series of 50 auction opponents, constructed randomly according to a normal distribution N(71,10). In this manner, we wanted to examine the performance of the algorithms with a theoretical population which distributes normally, though there is no evidence of such a distribution in any CE environment.

4.3 Experimental results

Table 1 presents the average payoffs for each algorithm. The average payoffs integrate the data of all the 200 permutations. Due to the fact that the algorithms' random factors cause variation in the results we ran each algorithm repeatedly for 50 times for each permutation,. The standard deviations of the 50 results of each algorithm with each permutation are also presented in order to measure the stability and self-consistence of each algorithm.

4.3.1 Payoff analysis

As can be derived from the results, the DVRL algorithm was almost always the most profitable algorithm. A non-parametric Friedman test revealed significant differences in the ranking of the algorithms ($p < 0.001$). Further pairwise Wilcoxon tests showed that the most efficient algorithm was significantly the DVRL algorithm, except for the auction, where DVG was significantly though only slightly better, and for the normal distributed auction, where no significant difference was found between DVRL and SVG.

Moreover, the extension of both DVL and Segment-based meta-algorithms almost always improved the payoff. Pairwise Wilcoxon tests revealed that the algorithms SVRL, DVRL, SVG and DVG always achieved higher payoffs than the first five algorithms ($p < 0.001$). The SVRL method always achieved the lowest payoff of the leading quartet (SVRL, DVRL, SVG and DVG) ($p < 0.001$), except for the UG, where no significant difference was found between SVRL and DVG. The internal ranking of SVG and DVG was inconsistent, with no significant differences in the all-pay auction and the UG variant. As for the SZWK algorithm, we found that its performance was always significantly higher ($p < 0.001$) than the basic ZWK's algorithm, except for the all-pay auction, where no significant difference was found between these methods.

Virtual learning was revealed as an efficient method, as can be concluded by the general superiority of VRL over ZWK (except for the UG variant), of VG over GS (except for the UG variant, and for UG, where no difference was found) and of SVRL over SZWK (except for the auction). A non-virtual version of SVG (a segment based extension of GS), which is not presented here because of space considerations, was similarly overpowered by SVG.

4.3.2 Results explanations

In order to understand the origin of the advantage of DVRL over DVG, SVRL and over SVG, we should observe **Figure 1**. This plot describes the average payoff of each algorithm - SVRL, DVRL, DVG and SVG - (the y-axis), during the interactions (the x-axis). The plot, which enables an easier tracking of the learning process, presents only the data of the all-pay auction. However, the data of the other environments were quite similar, thus we can refer to the all-pay auction as a representative case. The left side of the plot shows a noticeable preliminary advantage of the DVRL method. Particularly, the payoff of DVRL is higher than the others till the 12th round, where it becomes proximately equal to DVG. This preliminary advantage can be attributed to the low extent of reinforcement in the RL's Q-values update procedure, compared to the extensive reinforcement conducted by Gittins' indices. For example, the Q-value of an offer which succeeded 2 out of 3 interactions, would be reinforced in the RL method by 66% of its reward, while

[7]A combination of the 2 meta-algorithms is not presented in this paper since its performance was experimentally worse than DVRL's. Moreover, the implementation of the combined agent is much more complicated than DVRL's, since it includes a large number of configurational parameters.

[8]This method of responders' behavior examination is widely accepted in UG literature [1].

Environment		R&E	ZWK	VRL	GS	VG	SZWK	SVRL	DVRL	SVG	DVG
UG	payoff	35.55	39	41.3	37.43	37.44	40.13	41.59	**43.28**	42.35	41.79
	SD	5.56	4.2	4.03	4.02	3.90	3.86	3.47	2.3	2.17	2.12
UG variant	payoff	28.93	33.99	32.41	31.88	30.44	35.23	36.37	**39.24**	37.42	37.23
	SD	5.86	5.01	5.6	5.31	5.96	4.85	4.19	1.99	2.68	2.30
Auction	payoff	13.35	15.95	16.18	15.89	16.52	17.35	17.05	19.06	17.69	**19.37**
	SD	3.21	3.81	3.64	4.12	3.57	3.31	2.87	1.47	3.15	1.69
All-pay auction	payoff	-3.27	-1.88	-1.09	-2.14	-1.11	-1.99	-0.26	**2.91**	1.65	1.5
	SD	6.07	5.2	5.14	5.22	4.43	6.12	5.14	2.6	3.08	2.81
Normal distributed auction	payoff	7.83	11.66	12.39	11.08	12.63	13.03	14.14	**14.88**	14.82	14.74
	SD	2.12	3.18	2.94	4.16	2.87	2.87	1.72	0.57	1.09	0.84

Table 1: Average payoff and standard deviation of various algorithms against human opponents

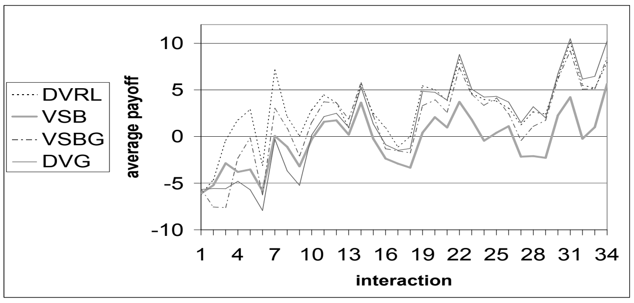

Figure 1: Average payoff during the all-pay auction

using Gittins' method it would be reinforced by 91%. An offer which was successful only once out of 5 interactions, would be reinforced in the RL approach by 20% of its reward, while using Gittins' method it would be reinforced by 47%. Thus, offers which are extremely far from the optimal will be filtered more quickly with the pedant RL methods.

The pedantry of RL is also the reason for the relative success of the SVRL in the first learning stages. However, in SVRL this pedantry becomes an obstacle in the progressive learning stages: Optimal segments, where the probability of success is usually not 100%, were sometimes rejected after a few interactions, and were hardly focused in later stages[9]. However, this problem in the progressive learning stages, was much less harmful in the DVRL method. The reason for this is that, in contrast to SVRL, (optimal) options with currently low Q-values will not necessarily be abandoned for long. In addition, according to the DVL update protocol, (optimal) options are also reinforced by the success of options which are less risky. Thus, the DVRL becomes less sensitive to over filtering of the pedant RL method.

This difference in sensitivity between the segment-based and the DVL approaches underlies the superiority of DVG over the VSBG algorithm, in the progressive learning stages, as noticeable on the right side of **Figure 1** (from the 16th round on). However, the difference is much less significant than the difference between the RL-based algorithms, since the Gittins approach is much less pedant. Therefore, the SVG is not critically harmed by the abandonment of low valued options. Moreover, the fast convergence to an optimal (large) segment which usually occurred in SVG, provided a superiority over the DVG algorithm, in the initial interactions.

Another finding of our experiment was the general improvement caused by virtual learning. This could be explained by the fast updating of a large set of offers after every interaction, with the VL method. Thus, offers which are significantly irrelevant are filtered within a few interactions, while it is easy to find a non virtual learner that proposes 90% in UG after the 20th interaction, for example. This significant advantage offsets the disadvantage in accuracy, caused by the information asymmetry in the virtual updating process mentioned above.

In order to measure the stability and self-consistence of each algorithm we compared the standard deviations of the payoff results of 50 runs with each permutation of the ten algorithms. Observing the standard deviations in **Table 1**, it can be concluded that the standard deviation of DVRL was always the least ($p < 0.001$), except for the UG, where both DVG and SVG were lower than DVRL. Since low values of standard deviation guarantee more certain and stable results, using the proposed DVRL algorithm is preferable from this aspect as well. This property may be attributed to the fact that the DVL meta-algorithm includes no random decision, except for the first interaction. In addition, the fast screening in the DVRL as well as the lower sensitivity to over-filtering of the optimal solution discussed earlier, provide a stable and consistent pattern of solution search. For similar reasons, the relative ranking of other algorithms in the deviations measures were compatible with the relative ranking in the payoffs measures.

4.3.3 *Comparison with human performance*

In this section we compare the performance of the DVRL algorithm with the performance of human competitors. For this purpose we asked our participants to compete iteratively against a series of other human opponents, exactly in the same manner the automated agents had been competing. In this manner we wanted to reveal whether the DVRL agent can outperform human natural intuitions and life experience in this context. Note that the automated agent was run with the same parameters for all the environments, and no domain specific knowledge had been applied to the agents.

We began our examination after collecting a series of acceptance threshold amounts in the ultimatum games from the first 21 participants, as well as their biddings in the auctions (whereas in previous sections the series included all the 34 participants). In order to enlarge the size of the series we inserted each opponent twice, creating a 42-round auction, a 34-round UG and a 36-round UG variant (when using the original series without the enlargement, the results were not qualitatively different). Similarly, we constructed an artificial series of 40 biddings in an auction normally distributed, as before. The consequent participants were asked to interact

[9]Extending the duration of the first stages, where a few number of large segments are observed, would have reduced this problem. However, since we considered environments with a total number of 50 interactions at most, the average total payoff would not have risen by extending the duration of the first stages, since it would often harm the later interactions, where the fine tuning is managed. The final stage of fine tuning is critically important, especially in the context of CE environments.

Environment (# of human participants)	**Human**	**DVRL**
UG (13)	49.56 (3.41)	52.59
UG variant (13)	40.8 (7.45)	44.36
Auction (13)	15.74 (4.02)	19.01
Normal distributed auction (15)	11.79 (2.78)	14.35

Table 2: Average payoff (SD) of humans and of a DVRL agent competing with human opponents.

with those series, functioning as proposers in the ultimatum games and as bidders in the auctions. After each decision, the participants were informed of the success of their offer by the ”current” opponent. Actually, these proposers were expected to learn the acceptance thresholds distribution of the responders' population, exactly as the agents learn. All the learners, including the automated learner, competed with the same series of opponents. The all-pay auction was not tested on human learners, due to time restrictions.

The average payoffs of the human learners, as well as the payoffs of the DVRL agent, in the exact same scenarios, are presented in **Table 2**. The results show a clear advantage of the DVRL agent over the human learners in all the environments. Although this statement cannot be proved statistically, since we ran only one permutation from each environment, the results show that the DVRL average payoff was usually about one standard deviation above the human's average payoff.

4.3.4 A small set of decision options environments

In order to check whether the suggested meta-algorithms are not harmful in a small set of options, we performed the exact same simulations with merely 10 decision options. The agents could choose an integer from 0 to 10 in each interaction, and the original values of the opponents' vectors were divided by 10 and then rounded to the closest integer. The results, not presented here due to space restrictions, showed a significant advantage of the DVRL over the other algorithms, both in payoffs and in standard deviations, almost in all cases. Moreover, the internal hierarchy among the algorithms was similar to the 100-options version, and we found a general significant advantage of both meta-algorithms over the basic algorithms. Nonetheless, the gap between the performance of the basic algorithms and their extensions was much smaller in the 10-options version.

5. CONCLUSION AND FUTURE WORK

We presented a new meta-algorithm, namely DVL, which extends existing methods for efficiently competing in multi-interaction one-shot CE environments with different human opponents. Our experimental findings show that the DVL extension of an RL basic algorithm is significantly better than humans and other algorithms surveyed in this paper, especially in environments which contain a large set of optional decisions at each decision point. Moreover, they show that both DVL and the segment-based meta algorithm, also presented here for the first time, almost always significantly improve the performance of the basic algorithms they extend.

In future work we intend to design agents which efficiently compete in CE environments for a high number of interactions. As mentioned above, in this study we were principally concerned with adaptable agents, which compete in no more than several dozens of repeated interactions. However, the DVRL algorithm does not necessarily grant efficiency for environments which last for hundreds of interactions. The reason is mainly due to the fact that the directed search for an optimal option, partially neglects the exploration of other options. Thus, we would like to explore the ”damage” DVRL and other algorithms cause in the long run, and to suggest efficient approaches for interacting in CE environments for long durations.

In addition, we intend to extend the approaches discussed in this paper to more sophisticated classes of interactions. Particularly, we would like to examine the repeated version of CE interactions, in which several negotiation rounds can be conducted against each opponent. When competing repeatedly against the same opponent, a specific modeling of the current opponent must be done, in addition to the generic modeling of the population. Moreover, in contrast to the one-shot version, a current move may influence the future behavior of the opponent, a fact that must be taken into account. Therefore, we intend to design an agent that develops several optional models of typical opponents in the population, and matches the appropriate model to each opponent with which it interacts.

6. REFERENCES

[1] P. Bourgine and B. Leloup. May learning explain the ultimatum game paradox ? Technical Report GRID Working Paper No. 00-03, Ecole Polytechnique, 2000.

[2] T. Brenner and N. Vriend. On the behavior of proposers in ultimatum games. *J. Econ. Behav. Organ.* forthcoming.

[3] A. Byde, C. Preist, and N. Jennings. Decision procedures for multiple auctions. In *AAMAS*, pages 613–620, 2002.

[4] J. DiMicco, A. Greenwald, and P. Maes. Dynamic pricing strategies under a finite time horizon. In *EC'01*, 2001.

[5] S. Fatima, M. Wooldridge, and N. Jennings. Sequential auctions for objects with common and private values. In *AAMAS*, pages 635–642, 2005.

[6] Y. Gal, A. Pfeffer, F. Marzo, and B. Grosz. Learning social preferences in games. In *AAAI*, pages 226–231, 2004.

[7] J. Gittins. *Multiarmed Bandits Allocation Indices*. Wiley, New York, 1989.

[8] B. Grosskopf. Reinforcement and directional learning in the ultimatum game with responder competition. *Experimental Economics*, 6(2):141–158, 2003.

[9] W. Guth and S. Huck. From ultimatum bargaining to dictatorship - an experimental study of four games varying in veto power. *Metroeconomica*, 48(3):262–279, 1997.

[10] V. Krishna and J. Morgan. An analysis of the war of attrition and the all-pay auction. *J. Econ. Theory*, 72:343–362, 1997.

[11] B. Leloup and L. Deveaux. Dynamic pricing on the internet: Theory and simulations. *Electronic Commerce Research*, 1(3):265–276, 2001.

[12] L. Niklasson, H. Engstrom, and U. Johansson. An adaptive rock, scissors and paper player based on a tapped delay neural network. In *ADCOG21*, pages 130–136, 2001.

[13] A. Roth and I. Erev. Learning in extensive form games: Experimental data and simple dynamic models in the intermediate term. *Games Econ. Behav.*, 8:164–212, 1995.

[14] R. Sutton and A. Barto. *An Introduction to Reinforcement Learning*. MIT Press, 1998.

[15] P. Todd and B. Borges. Designing socially intelligent agents for the ultimatum game. In K. Dautenhahn, editor, *Socially intelligent agents-Papers from the 1997 Fall Symposium*, pages 134–136. AAAI Press, Menlo Park, CA, 1997.

[16] N. Vreind. Will reasoning improve learning? *Economics Letters*, 55(1):9–18, 1997.

[17] F. Zhong, D. Wu, and S. Kimbrough. Cooperative agent systems: Artificial agents play the ultimatum game. *Journal of Group Decision and Negotiation*, 11(6):433–447, 2002.