Learning Against Non-Stationary Opponents in Double Auctions

Pablo Hernandez-Leal
Instituto Nacional de Astrofísica, Óptica y Electrónica
Sta. Maria Tonantzintla, Puebla, México
pablohl@ccc.inaoep.mx

Matthew E. Taylor
Washington State University
taylorm@eecs.wsu.edu

Enrique Munoz de Cote and L. Enrique Sucar
Instituto Nacional de Astrofísica, Óptica y Electrónica
Sta. Maria Tonantzintla, Puebla, México
{jemc,esucar}@inaoep.mx

ABSTRACT

Energy markets are emerging around the world. In this context, the PowerTAC competition has gained attention for being a realistic and powerful simulation platform that can be used to perform robust research on retail energy markets. Agent in this complex environment typically use different strategies throughout their interaction, changing from one to another depending on diverse factors, for example, to adapt to population needs and to keep the competitors guessing. This poses a problem for learning algorithms as most of them are not capable of handling changing strategies. The previous champion of the PowerTAC competition is no exception, and is not capable of adapting quickly to non-stationary opponents, potentially impacting its performance. This paper introduces DriftER, an algorithm that learns a model of the opponent and keeps track of its error-rate. When the error-rate increases for several timesteps, the opponent has most likely changed strategy and the agent should learn a new model. Results in the PowerTAC simulator show that DriftER is capable of detecting switches in the opponent faster than an existing state of the art algorithms against switching (non-stationary) opponents obtaining better results in terms of profit and accuracy.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems

General Terms

Algorithms, Experimentation

Keywords

Non-stationary strategies, PowerTAC, opponent modelling, Markov decision process, energy markets

1. INTRODUCTION

Together with the adoption of smarter energy grids comes the idea of deregulating the energy supply and demand. Energy markets are emerging in many different parts of the world due to this shift. In fact, these smarter (deregularized) grids have enabled producers to sell energy to consumers by using a *broker* as an intermediary. This has led to the development of several strategies for trading energy. However, this domain remains as a challenge due to its complexity (rich state spaces, high dimensionality and partial observability [18]), where straightforward game-theoretic analysis, machine learning and artificial intelligence techniques fall short. Moreover, in this complex environment it is reasonable to expect that agents will use different strategies throughout their interaction and change from one to another. This will pose a problem for learning algorithms since most of them are not capable to deal with changing strategies.

Recent approaches that are based on multiagent systems have studied the operation of energy markets. Related to this, one particular competition that has gained attention recently is PowerTAC [8]. It simulates an entire energy market with producers, consumers and brokers buying and supplying energy. In particular, PowerTAC simulates a wholesale market that operates similar to stocks. In this market, several brokers interact with each other buying and selling energy. Their objective is to submit optimal offers to the market for delivery between one and 24 hours in the future. The champion agent of the PowerTAC competition models the wholesale market as a Markov decision process (MDP) and its solution dictates the optimal offers to submit at each timeslot. Even though this formulation is able to perform efficiently with reasonable results, it does not handle non-stationary opponents (that change among one stationary strategy to another).

This paper's main contribution is to introduce DriftER —Drift (based on) Error-Rate—, an algorithm based on concept drift ideas that is capable of adapting quickly to non-stationary opponents. In order to learn the environment, DriftER treats the opponent as a stationary (Markovian) environment, but keeps track the quality of the learned model as an indicator of a possible change in the opponent strategy. When a strategy switch in the opponent is detected, DriftER stops using the learned model and restarts the learning. We compare our algorithm in the PowerTAC simulator against two algorithms: the champion of the inaugural competition and a state of the art algorithm for non-stationary opponents. The results show the effectiveness of our approach, obtaining better results in terms of total profit, and accuracy.

2. RELATED WORK

In this section we review recent works in energy markets, non-stationary environments and then we present related concept drift approaches which provide the basis for DriftER.

2.1 Energy markets

Markets for renewable energy are expanding, shifting investment patterns away from traditional government and international sources to greater reliance on private firms and banks. This results in increased decision-making and participation from a wider variety of stakeholders (brokers) [10]. However, by so doing, new challenges have emerged, such as predicting on energy demand [1], coordination on power distribution [13], and energy generation and distribution [23].

Different markets appear in this area, for example the wholesale energy market, which is commonly known as day-ahead market, is one of the most important in the energy sector. In this market, brokers make bids (offers) for buying or selling energy delivery between one and 24 hours in the future. The interaction happens between brokers and commodities that they own (the good and money). The interactions between the agents lead to exchange, a process whereby the traders freely alter the allocation of commodities. Thus traders may choose to swap a good for some quantity of money, and the amount they choose is related to the value the trader places on the good. This value is known as limit price. Some works do not explicitly model other brokers, like [4], where the authors start by predicting hourly prices, from that, a schedule is obtained by an optimization model. Solving this model allows deriving a simple bidding rule.

A second important market is the tariff (customer) market which handles the interaction between brokers and local producers and consumers. Previous works [15] used a simulation approach to investigate a heavily simplified competitive tariff market that was modelled by a Markov decision process [14]. The authors proposed different actions to alter tariff prices. In [19] the authors propose a prediction-of-use tariff that asks customers to predict a baseline consumption, charging them based both on their actual consumption and the deviation from their prediction.

In contrast to these two works that do not perform any type of modelling for the wholesale energy market, our approach focuses on it. The wholesale and tariff markets are modelled in PowerTAC, which we present in the next section.

2.2 PowerTAC

Power TAC is a competitive simulation that models a retail electrical energy market, where brokers offer energy services to customers through tariff contracts, and must then serve those customers by trading in a wholesale market [8]. Brokers are challenged to maximize their profits by buying and selling energy in the wholesale and tariff markets, subject to fixed costs and constraints. The simulation environment consists of different markets where brokers have to take actions at each timestep (timeslot) which simulates one hour of real time. The three main markets in PowerTAC are the following.

- The tariff market, where brokers buy and sell energy by offering tariff contracts that specify price and other characteristics like early withdraw fee, bonus for subscription and expiration time. Customers choose among those different contracts and later they decide to continue or to change to a different one.
- The wholesale market allows brokers to buy and sell quantities of energy for future delivery. It operates as a periodic double auction [22], and its similar to many existing wholesale electric power markets, such as Nord Pool in Scandinavia or FERC markets in North America [9].
- The balancing market is responsible for the real-time balance of supply and demand on the distribution grid. The market creates an incentive for brokers to balance their own portfolios of energy supply and demand in each time slot by ensuring that they would be better off balancing their portfolios than relying on the balancing market to do it.

2.2.1 TacTex

The champion agent from the inaugural competition in 2013 was TacTex [18], which uses an approach based on reinforcement learning for the wholesale market and prediction methods for the tariff market. For modeling the wholesale market TacTex uses a modified version of Tesauro's representation of a double auction market [17]. The idea is that states represent agent's holdings, and transition probabilities are estimated from the market event history. The model is solved via dynamic programming every time the agent had an opportunity to bid. TacTex uses a MDP to model the sequential bidding process. TacTex starts a game with no data and learns to bid online, while acting its estimates are refined during the game. At each timeslot, it solves a MDP with all the data collected thus far, providing the optimal limit price of the biddings for the next hours. Even tough TacTex learns to quickly bid in an online environment, it is not capable of adapting to non-stationary opponents. This is a large drawback, as many real-life strategies do not follow a static (stationary) regime throughout their interaction. Instead, they either slowly change or drastically switch from one strategy to another (either to leave the opponent off-guard and guessing or just as a best response measure).

Now we review relevant work regarding learning in non-stationary environments.

2.3 Non-stationary environments

Energy markets are complex and agents in that environment will probably change behaviours during their interaction, which renders the environment non-stationary.

A recent approach that studies non-stationary environments is the MDP-CL framework (MDP-continuous learning) [7] which is composed of three parts: 1) A learning phase to learn a model of the opponent in the form of a MDP, 2) a planning phase that solves the MDP to obtain an optimal policy against the modelled opponent, and 3) a process that embeds the learning and planning phases to identify switches in the opponent strategy. The approach has been tested on the iterated prisoner's dilemma. However, this approach has not been evaluated in real world scenarios such as energy markets.

The machine learning community has developed an area related to non-stationary environments and online learning which is called concept drift [20]. The approach is similar to a supervised learning scenario where the relation between the input data and the target variable changes over time [6].

In particular, the work in [5] studies the problem of learning when the class-probability distribution that generates the examples changes over time. A central idea is the concept of context: a set of contiguous examples where the distribution is stationary. The idea behind the concept drift detection method is to control the online error-rate of the algorithm. When a new training instance is available, it is classified using the actual model. Statistical theory guarantees that while the distribution is stationary, the error will decrease. When the distribution changes, the error will increase. Therefore, if the error is greater than a defined threshold, it means that the concept has changed and it needs to be relearned. The method was tested on both artificial and real world datasets. Our algorithm also bases on concept drift ideas but it is adapted to a multiagent setting like is the case of Power TAC, where we study a set of changing opponents (environments).

3. PRELIMINARIES

Now we review some important concepts of reinforcement learning (RL), Markov decision processes, and algorithms for learning non-stationary environments.

In RL, an agent must learn an optimal policy for maximizing its expected long-term reward in an initially unknown Markov decision process. A MDP is specified by a tuple $< S, \mathcal{A}, T, R >$ where: S is the set of states in the world. $\mathcal{A}$ is the set of available actions that affect the environment. The transition function $T : S \times \mathcal{A} \rightarrow S$ represents the dynamics of the process. It gives a probability $T(s, a, s')$ of transitioning to state s' while being in state s and performing action a. The reward function $R(s, a)$, specifies a value obtained by the agent when performing action a in state s.

3.1 Modelling non-stationary strategies

Modelling non-stationary strategies requires the model to be updated frequently (every time a change occurs). One way to update such a model is by identifying when the model is inconsistent with the reality. When such inconsistency is revealed, one can safely discard the previously acquired model and learn a new model that captures the change in the opponent strategy. One approach that has been successful in identifying sudden strategy switches[1] is MDP-CL.

MDP-CL (MDP - continuous learning) [7] is a model based multiagent learning technique based on RL and MDPs, designed to handle non-stationary opponents. It consists of three parts. Learning is the initial phase and uses an exploration strategy in order to learn the opponent's model. MDP-CL learns MDPs to capture the opponent's dynamics (its strategy). Examples of exploration strategies include: random behaviour, softmax, and R-MAX[3]. The next phase is planning, where an optimal policy π^* is computed (using the learned model) to play against the opponent. To solve it, value iteration is used [2]. The computed policy is used and the switch detection process starts. Switching from an exploration strategy to and optimal strategy could trigger a response from the opponent, but the agent should to be able to detect such changes. For this another model is learned concurrently and compared every w rounds, if difference between models is greater than a threshold then it means a switch has happened and the algorithm restarts the learning phase discarding the previous model.

[1]Strategy switches are a way of inducing a non-stationary strategy.

4. DRIFTER

When facing non-stationary opponents two aspects are important: exploring the opponent actions to detect switches and to keep track of the opponent model. DriftER treats the opponent as a stationary (Markovian) environment however it uses concept drift ideas to keeping track on the quality of the learned model as an indicator of a possible change in the opponent strategy. When a switch in the opponent strategy is detected, DriftER resets its learned model and restarts the learning. In order to promote an efficient opponent switch detection a new type of exploration is applied, which is called drift exploration.

4.1 Modeling wholesale electricity markets

Recall that in PowerTAC a wholesale broker can place a bid for buying or selling energy by issuing a tuple $\langle t, e, p \rangle$ that represents the timeslot t the broker makes a bid/ask for an amount of energy e (expressed in megawatt-hour MWh) at a limit price p of buying/selling. At each timeslot, PowerTAC provides (as public information) market clearing prices and the cleared volume. It also provides as private information (only to each respective broker) the successful bids and asks [8]. A bid/ask can be partially or fully cleared. When a bid is fully cleared the total amount of energy will be sent at the requested timeslot, if a bid was partially cleared the offer was accepted but there is not enough energy and only a fraction of the requested energy will be sent.

Next is the MDP formulation of TacTex [18]:

- States: $s \in \{0, 1, \ldots, n, success\}$, represent the timeslots for future delivery for the bids in the market. $s_0 := n$, terminal States: $0, success$.

- Actions: values that represent limit prices for the offers in the wholesale market.

- Transition: a state $s \in \{1, \ldots, n\}$ transitions to one of two states. If a bid is partially or fully cleared, it transitions to the terminal state *success*. Otherwise, a state s transitions to state $s - 1$. The transition probability is initially unknown.

- In state $s = 0$, the reward is a balancing-penalty. In states $s \in 1, \ldots, n$, the reward is 0. In state *success*, the reward is the limit price of the successful bid.

The MDP's solution determines an optimal limit-price for each of the n states. TacTex is always in states $1, \ldots, n$ of n concurrent bidding processes. Therefore, it solves the MDP once per timeslot, and submits the n optimal limit-prices to the n auctions.

Since TacTex formulation uses MDPs and is the champion of the 2013 competition, we will use it as comparison in the experiments. With this in mind, we propose DriftER, which overcomes TacTex's limitation of not being able to handle non-stationary opponents and MDP-CL, which is dependent on a parameter w whose optimal value can only be known *a-posteriori* (i.e., after the game has been played). The proposed approach draws insights from concept drift and proposes a new type of exploration (drift exploration).

4.2 Learning an opponent model

We now define the MDP that models the wholesale market in DriftER. We use the same representation as TacTex (see Section 4.1). In order to learn the transition function we use the set of cleared transactions CT.[2] Each cleared transaction $tr \in CT$ is associated with a state s (timeslot for future delivery) and contains the information of cleared energy cE and the clearing price cP. Then for each next state s' executing action *limitPrice* we compute the value:

$$t_{s'}^{limitPrice} := \frac{P_{tr \in CT[s], t.cP < limitPrice} t.cE}{t \in CT[s] t.cE}$$

and the transition function is

$$T(s, a, s') = \begin{cases} t_{s'}^{a} & \text{if } (s' == success) \\ 1 - t_{s'}^{a} & \text{otherwise} \end{cases}$$

The value $t_{s'}^{limitPrice}$ captures the probability of being in state s using action *limitPrice* to go to state *success* (cleared transaction). It considers (from the set cleared transactions) only the clearing prices that are lower than the proposed limit price. If the transaction is not cleared, it goes to state $s-1$. Because the agent has no initial information, it must collect data to develop a transition function. It starts with random actions during the first k timeslots, called the learning phase. After this phase the MDP can be solved. We assume that during learning phase the opponent will remain stationary.

4.3 Drift exploration

Most exploration techniques decrease their exploration rate over time so that the learned (optimal) policy can be applied. However, against non-stationary opponents, it is well known that exploration should not decrease so as to detect changes in the structure of the environment at all times [11].

The problem with non-stationary environments is that opponent strategies may share similarities in their induced MDP (specifically between transition functions). If the agent's optimal policy produces an ergodic set of states, and this part of the MDP is shared between opponent strategies, the agent will not perceive such strategy change, which results in a suboptimal policy and performance. The solution for this is to explore, *even when an optimal policy has been learned.* We coined this type of exploration as "drift exploration". For this reason we applied drift exploration in the form ϵ-greedy even when we have an optimal action to perform (a learned MDP).

4.4 Switch detection

Approaches such as MDP-CL that compare pairs of models in fixed timesteps are not efficient because two parameters need to be tuned: the window size that controls the how often comparisons are made and the threshold value that defines a measure of how different models should be to mark switch in the strategy. Both parameters depend heavily on the domain and opponent. In contrast, our proposed algorithm keeps track on the opponent at every timestep in an efficient and practical way, with a measure independent of the model of the opponent.

The DriftER approach considers online learning: at each timestep the algorithm decides to continue with the current model or change to a new one. For instance, assume the agent already has a learned model of the opponent (as presented in Section 4.2). Using that model, it can predict at each timestep the next state of the opponent $\hat{s}_i$ and can be compared with the real value s_i. This comparison can be binarized with *correct/incorrect* values and this can be seen as a Bernoulli trial. Assuming a sequence of independent identically distributed events will produce a Bernoulli process. For each i in the sequence, the error-rate $error(s_i)$ is the probability of observing *incorrect*. Statistical theory guarantees that while the class distribution of the examples is stationary, the error rate $error(s_i)$ will decrease when i increases.

At this point the error rate can be improved by taking into consideration a confidence interval over the error rate $conf(s_i)$ by using the Wilson score [21]. In this way the number of examples seen in the sequence are taken into account and with more data our estimates will have more confidence.

The idea of the DriftER is keeping track of this $conf(s_i)$ value at each timestep. We will expect a decrease in this value which indicates the current model is correct and useful. However, $conf(s_i)$ may increase for two reasons. The first one is noise in the opponent, which can happen when trying to model opponents that make mistakes or explore. In this case, we do not want to learn a new model (since the model has not changed) but instead should stay with the current one. The second reason for an increase in the $conf(s_i)$ value is that the opponent has switched to a different strategy and the learned model is no longer useful for predictions. In this case, we want to stop using the current model and learn a new one. In order to fulfil these requirements we keep track of the first derivate $conf'(s_i)$ of the last n timesteps. if $conf'(s_i) > 0 == true$ in at least last m out of n steps the algorithm decides a switch is detected and stops using the current model (restarting the learning phase).

4.5 Tariff Market

Our approach considers bidding in the wholesale market. However, its not possible to isolate markets on PowerTAC thus to perform experiments we propose a simple strategy in the tariff market. The approach is to publish one flat tariff which is the average of the tariff's history.

5. EXPERIMENTS

Experiments were performed on the PowerTAC simulator. We first present results only on the wholesale market and then on the complete PowerTAC setting. To perform comparisons we selectTacTex-WM,[3] which is the champion of the 2013 competition, and MDP-CL which is not specific for PowerTAC but is designed for non-stationary opponents. The opponent is non-stationary in the sense that uses two stationary strategies: it starts with a fixed limit priced P_l and then in the middle of the interaction changes to a different (higher) fixed limit price P_h. Even that we used a fixed timeslot for the switch in the opponent this value is not known for the learning agents since in open environments it is reasonable to not known the possible times of switching of the opponents. Also even though we define a fixed limit price and there is only a single opponent (other buying broker), PowerTAC includes as default 7 wholesale

[2]Cleared transactions are those bids that were accepted in the market.

[3]We only use the wholesale market part of TacTex (TacTex-WM)

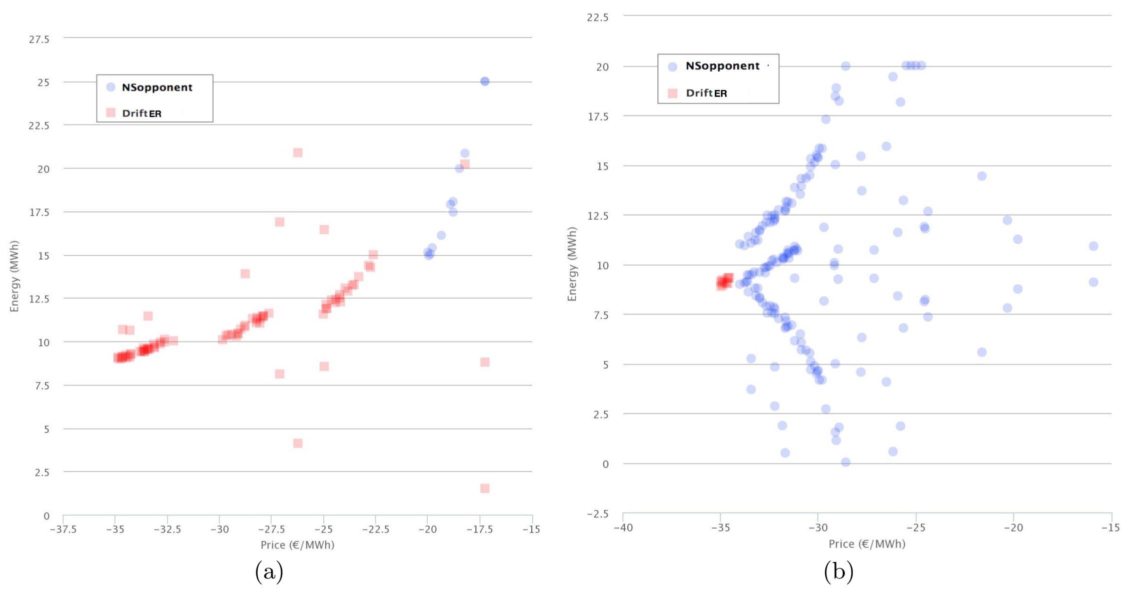


Figure 1: Cleared transactions of DriftER (red squares) and the non-stationary opponent (blue dots). In (a) the opponent uses as limit price the value 20 whereas in (b) it uses a limit price of 34.

energy providers as well as 1 wholesale buyer to ensure liquidity of the market [8] which introduces another source of uncertainty and randomness added in the simulation.

The MDP that models the opponent uses the following parameters: the number of states was set $n = 6$, and the actions represent limit prices with the values $\{15, 20, 25, 30, 35\}$. The opponent started with a $P_l = 20$ and then changed to $P_h = 34$. In the first case the learning agent needs to bid using a $price > 20$ $(25, 30, 35)$. Later when the opponent uses a limit price of 34, the only bid that will be accepted is 35. Both the learning agent and the opponent have a fixed demand $20Mwh$ for each timeslot which is greater than the energy generated by the providers.

We present results on terms of *average accuracy*, *confidence on error rate* and *profit*. The learned MDP contains a transition function for each (s, a) pair, comparing the predicted next state with the real one gives an accuracy value. At each timestep the agent submits n bids and its learned model predicts if those bids will be cleared or not. When the timestep finishes it receives feedback from the server and compares the predicted with the real transactions. An average of those n predictions is the accuracy of each timestep. A value close to 1 means perfect prediction. A similar measure is confidence over error-rate as described in Section 4.4, the main difference is that it takes into account the number of samples used to obtain the error rate. Finally, profit is defined on PowerTAC as the income minus the costs (balancing, wholesale and tariff markets). Each competition had a duration of 200 timeslots (each timeslot simulates one hour) equivalent to simulation of 8 days and 8 hours. We used default parameters for the all other settings in PowerTAC.

5.1 Learning a model of the opponent

The first experiment aims to determine the size of learning window for the learning agents. TacTex used only 6 timeslots [18] for gathering information and then stopped the learning period and used the MDP policy. However, since this setting is different we decided to evaluate different sizes of learning periods and compare their accuracy. In this case the opponent used a stationary strategy for the complete interaction. The duration of the competition was 200 timeslots.

Table 1: Accuracy results of TacTex-WM while varying the learning size. The competition lasted 200 timeslots.

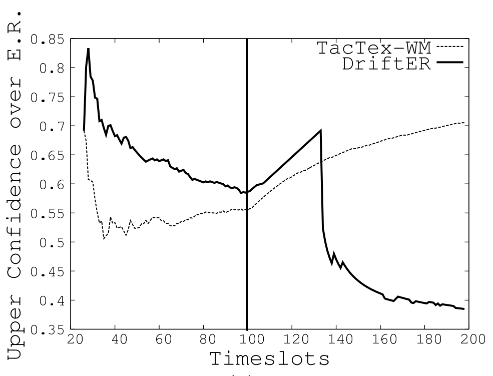
Learning Size	Accuracy
5	0.550 ± 0.05
15	0.563 ± 0.11
25	0.679 ± 0.05
35	0.674 ± 0.10
45	0.672 ± 0.06

In Table 1 we present the learning size and the respective accuracy for TacTex-WM. Results show that TacTex-WM performs well and learns fast with approximately 25 timeslots obtaining an accuracy of nearly 70%. From these experiments we set the learning phase to 25 for the learning algorithms. We note that increasing further the learning period does not have an impact on accuracy. This happens due to the limited set of actions and also because the proper limitation of this representation. The next section introduces a non-stationary opponent that changes from one strategy to another.

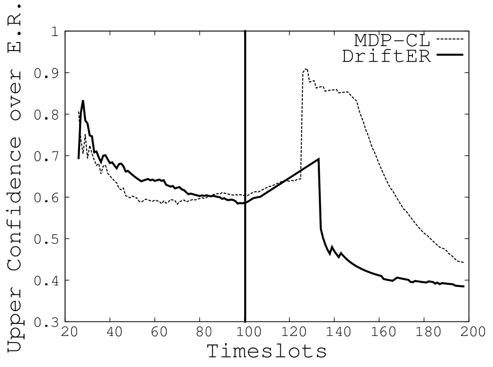
5.2 Non-stationary opponents

Figures 1 (a) and (b) show graphs of the cleared transactions of DriftER against the non-stationary opponent. In 1 (a), we note three clusters for the cleared transactions (squares) of DriftER, each cluster correspond to the limit prices $\{25, 30, 35\}$[4]. In contrast, accepted bids for the opponent (dots) always have a value lower to 20 (since it is the

[4]PowerTAC takes these prices as negative since it as a buy-



(a)



(b)

Figure 2: Error rate of (a) TacTex-WM and (b) MDP-CL while comparing with DriftER.

stationary limit price). In Figure 1 (b), we depict a similar graph but after the opponent has switched to the second strategy (the limit price is now 34) and DriftER has updated its policy. In this figure we can see that now the opponent dominates the cleared transactions. The only limit price that produces cleared bids for DriftER is 35, thus we can see a cluster near that value. These figures show how the optimal policy of the learning agent needs to be updated to cope with the opponent switching behavior.

Now we compare the three learning algorithms in terms of error-rate against the switching opponent. In Figure 2 (a) error rates of TacTex-WM and DriftER are depicted. We can observe that for both algorithms, after the learning phase (from timeslot 25) the error rate reduces. However, starting from round 100 (when the opponent changes its strategy) the error rate of TacTex-WM increases since it is not able to adapt to the opponent. In contrast, DriftER shows an increase in the error rate after the opponent switch (timeslots 100 to 110), however at timeslot 110 DriftER stops using the learned policy and restarts the learning phase which ends at timeslot 135. At this point its confidence over the error rate is high (since it is just a new model) and it shows a peak. From that point DriftER has learned a new MDP and a new policy which reduces the error rate consistently. In Figure 3 (b) error rates of MDP-CL and DriftER are depicted. We can observe that after the opponent's switch both algorithms are able to detect it. However, since MDP-CL performs comparisons to detect switches every w steps

ing action

Table 2: Number of average timeslots for switch detection (Avg. S.D. Time), accuracy, and traded energy of the learning agents against a non-stationary opponent.

	Avg. S.D. Time	Accuracy	Traded E.
MDP-CL	85.0 ± 55.0	57.55 ± 28.56	2.9 ± 1.3
DriftER	**33.2 ± 13.6**	**67.60 ± 21.21**	**4.4 ± 0.5**

($w = 25$ in this case) at least it has to wait 25 timeslots (in contrast to 10 timeslots of DriftER).

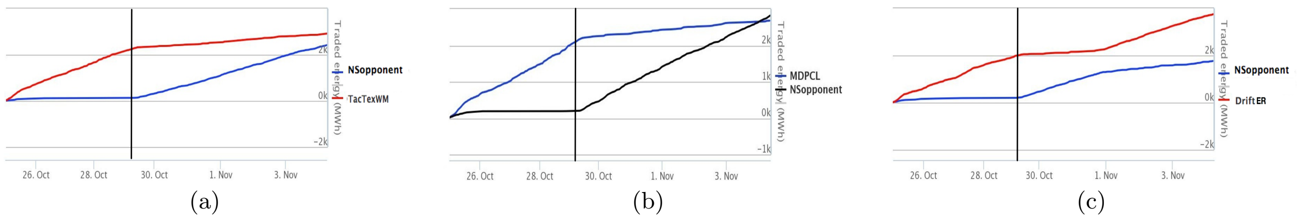
We performed more experiments reducing the w parameter in order to reduce the time to detect switches. However, we noticed that we also needed to select an appropriate parameter for the threshold. Optimizing these parameters is time consuming since $w \in \mathbb{N}$ and $threshold \in \mathbb{R}$. We set $w = 25$ and selected the best value (based on accuracy) for setting *threshold*. In next section we review directly both MDP-CL and DriftER against switching opponents.

5.3 Detecting switches in the opponent

Now we compare MDP-CL and DriftER since both approaches handle non-stationary opponents. We measure the average number of timeslots needed to detect the switch and we also measure the accuracy and the traded energy as a measure of indirect cost provided by PowerTAC, the more time it takes to detect the switch the more time the agent will trade less energy. In Table 2 we depict the results for MDP-CL (using w= 25) and DriftER. In this case the competition lasted 250 timesteps, the opponent switched at timestep 100. Results are the average of 10 iterations. Results show that DriftER needs less time for detecting switches and also shows a lower standard deviation showing that is more robust than MDP-CL. DriftER also obtained better accuracy (explained by the fast switch detection) and as a result is capable of trading more energy.

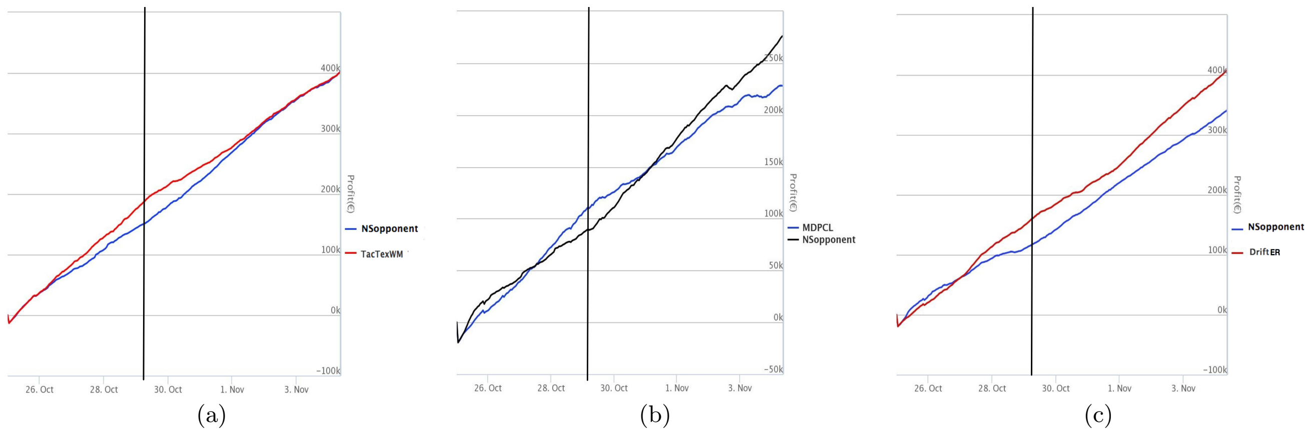
Now we review the three approaches in Figure 3 (a), (b) and (c) we depict the cumulative traded energy of the learning agents and the switching opponent (timeslot when the opponent switches is displayed with a vertical line). From the figures we note that in the first part of the game (before the vertical line) TacTex-WM (Figure 3 (a)), MDP-CL (Figure 3 (b)) and DriftER (Figure 3 (c)) increase consistently their traded energy. In contrast, the traded amount of energy for the opponent is severely limited since the learning agents are clearing most of their bids. However, at timeslot 100 the opponent changes its strategy and increases their cleared transactions (traded energy). Against TacTex-WM the opponent increases its traded energy for the rest of the competition (see Figure 3 (a)). As a result TacTex-WM reduces its traded energy. MDP-CL and DriftER (see 3 (b) and (c)) are affected by the change in strategy. However, after some timeslots (where it has detected the switch and learned a new policy) they start increasing its traded energy. Note that MDP-CL takes more timesteps than DriftER. Therefore DriftER achieves better results due to a faster switch detection.

We have discussed results only on the wholesale market. In order to evaluate the algorithms in terms of profit on PowerTac we implemented a simple strategy on all the learning algorithms in order to perform a fair comparison (see Section 4.5). In Figure 4 we depict the cumulative profit of



(a) (b) (c)

Figure 3: Amount of traded energy for (a) TacTex-WM, (b) MDP-CL and (c) DriftER against the non-stationary opponent in a competition of 250 timesteps. Timestep when the opponent switches is displayed with a vertical line.



(a) (b) (c)

Figure 4: Profit of (a) TacTex-WM, (b) MDP-CL and (c) DriftER against the non-stationary opponent in a competition of 250 timesteps. Timeslot where the opponent switches is displayed with a vertical line.

(a) TacTex-WM and (b) MDP-CL and (c) DriftER against the non-stationary opponent, the timeslot where the opponent switches between strategies is noted with a vertical line. From this figure we note that TacTex-WM profits increase before the opponent switch and the decrease after it. In the end of the interaction they both obtain a similar profit. In contrast, DriftER profits increase again after the switch. In terms of cumulative profits DriftER obtained in average 80k€ more profits than the opponent. MDP-CL was capable of detecting switches but took much more timesteps and it obtained even worse profit than TacTex-WM.

5.4 Noisy non-stationary opponents

In the previous experiments the opponent switched between two fixed strategies. In this section we present a better approximation to real-world strategies. The opponent has a limit price $P_l = 20.0$ with a noise of ± 2.5 (bids are in the range $[18.5 - 22.5]$). Then, it switches to $P_h = 34.0$, with bids in the range $[31.5, 36.5]$. The rest of the experiment remains the same as in the previous section.

Similar to the non-stationary opponent without noise, DriftER is capable to adapt to the switching opponent and therefore it increases its profits after the switch. In Table 3 we show the total profits of the learning agents against the non-stationary opponents with, and without, noise. From the table, note that against fixed opponents TacTex-WM obtains the lowest score, while obtaining a low standard deviation. When the opponent increases the noise by using a range of values to bid, TacTex-WM reduces its profits and increases its standard deviation. MDP-CL shows competitive profit scores with fixed opponents, but it shows a high standard deviation. Against a noisy opponent MDP-CL obtained lower scores than the opponent. DriftER shows the second best score in profit and obtains it with lower standard deviation than MDP-CL against fixed opponents. Against noisy non-stationary opponents shows a good compromise obtaining competitive profits, a reasonable standard deviation and its the only algorithm which obtains better scores than the opponent which is explained by the fact than it is capable to quickly adapt to the new strategy and obtaining an optimal policy which prevents the opponent to increase its profits.

6. CONCLUSIONS AND FUTURE WORK

Energy markets are emerging in different parts of the world. PowerTAC competition has gained attention for being a powerful simulation platform that can be used to perform robust research on retail energy markets. Wholesale market is one of the most important markets since needs to guarantee a certain amount of energy to the agent. The previous champion of the competition was not capable of adapting quickly to non-stationary opponents (which change from one stationary strategy to another), impacting their total profits. We propose DriftER, an algorithm that learns a model of the opponent in the form of a MDP and keeps tracks of its error-rate. When the error-rate increases for several timesteps, it means the opponent has changed its strategy and we must learn a new model. Results on the PowerTAC simulator show that DriftER is capable of de-

Table 3: Average profit of the learning agents against non-stationary opponents with and without noise.

	TacTex-WM		MDP-CL		DriftER	
	Agent	Opp	Agent	Opp	Agent	Opp
Fixed NS	219.0 ± 7.5	228.7 ± 31.7	261.3 ± 65.8	253.5 ± 75.5	253.0 ± 38.9	228.7 ± 64.2
Noisy NS	198.0 ± 41.3	197.6 ± 24.78	260.1 ± 75.0	305.6 ± 41.18	255.9 ± 39.9	229.0 ± 38.2

tecting switches in the opponent faster than a state of the art algorithms. Future work will address on using transfer learning ideas for not forgetting the previous model but using it to promote a fast learning.

7. REFERENCES

[1] M. L. Baptista, H. Prendinger, R. Prada, and Y. Yamaguchi. A cooperative multi-agent system to accurately estimate residential energy demand. In *AAMAS '14: Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, May 2014.

[2] R. Bellman. A Markovian decision process. *Journal of Mathematics and Mechanics*, 6(5), 1957.

[3] R. I. Brafman and M. Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *The Journal of Machine Learning Research*, 3:213–231, 2003.

[4] A. J. Conejo, F. J. Nogales, and J. M. Arroyo. Price-taker bidding strategy under price uncertainty. *IEEE Transactions on Power Systems*, 17(4):1081–1088, Nov. 2002.

[5] J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with Drift Detection. In *Advances in Artificial Intelligence–SBIA*, pages 286–295, Brazil, Oct. 2004.

[6] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *Computing Surveys (CSUR*, 46(4), Apr. 2014.

[7] P. Hernandez-Leal, E. Munoz de Cote, and L. E. Sucar. A framework for learning and planning against switching strategies in repeated games. *Connection Science*, 26(2):103–122, Mar. 2014.

[8] W. Ketter, J. Collins, and P. Reddy. Power TAC: A competitive economic simulation of the smart grid. *Energy Economics*, 39:262–270, Sept. 2013.

[9] W. Ketter, J. Collins, P. P. Reddy, and M. D. Weerdt. The 2014 Power Trading Agent Competition. Technical report, 2014.

[10] E. Martinot, A. Chaurey, D. Lew, J. R. Moreira, and N. Wamukonya. Renewable energy markets in developing countries. *Annual Review of Energy and the Environment*, 27(1):309–348, Nov. 2002.

[11] E. Munoz de Cote, A. Chapman, A. M. Sykulski, and N. R. Jennings. Automated Planning in Repeated Adversarial Games. In *26th Conference on Uncertainty in Artificial Intelligence (UAI 2010)*, Catalina Island, California, 2010.

[12] S. Parsons, M. Marcinkiewicz, J. Niu, and S. Phelps. Everything you wanted to know about double auctions, but were afraid to (bid or) ask. Technical report.

[13] A. Perrault and C. Boutilier. Efficient coordinated power distribution on private infrastructure. In *AAMAS '14: Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, May 2014.

[14] M. Puterman. *Markov decision processes: Discrete stochastic dynamic programming.* John Wiley & Sons, Inc., 1994.

[15] P. Reddy and M. Veloso. Learned Behaviors of Multiple Autonomous Agents in Smart Grid Markets. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, May 2011.

[16] M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *The Journal of Machine Learning Research*, 10:1633–1685, 2009.

[17] G. Tesauro and J. L. Bredin. Strategic sequential bidding in auctions using dynamic programming. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 2*, July 2002.

[18] D. Urieli and P. Stone. TacTex'13: A Champion Adaptive Power Trading Agent. In *Association for the Advancement of Artificial Intelligence 2014*, Quebec, Canada, May 2014.

[19] M. Vinyals, V. Robu, A. Rogers, and N. R. Jennings. Prediction-of-use games: a cooperative game theoryapproach to sustainable energy tariffs. In *AAMAS '14: Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, May 2014.

[20] G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.

[21] E. B. Wilson. Probable Inference, the Law of Succesion, and Statistical Inference. *Journal of the American Statistical Association*, 22(158):209–212, Jan. 1927.

[22] P. R. Wurman, W. E. Walsh, and M. Wellman. Flexible double auctions for electronic commerce: theory and implementation. *Decision Support Systems*, 24(1):17–27, Dec. 1998.

[23] R. Zheng, Y. Xu, N. Chakraborty, and K. Sycara. Multiagent coordination for demand management with energy generation and storage. In *AAMAS '14: Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, May 2014.