Investigation of Learning Strategies for the SPOT Broker in Power TAC

Moinul Morshed Porag Chowdhury[†], Russell Y. Folk[‡], Ferdinando Fioretto[‡], Christopher Kiekintveld[†], and William Yeoh[‡]

[†]Department of Computer Science, The University of Texas at El Paso
mchowdhury4@miners.utep.edu cdkiekintveld@utep.edu
[‡]Department of Computer Science, New Mexico State University
{rfolk,ffiorett,wyeoh}@nmsu.edu

Abstract. The Power TAC simulation emphasizes the strategic problems that broker agents face in managing the economics of a smart grid. The brokers must make trades in multiple markets and, to be successful, brokers must make many good predictions about future supply, demand, and prices in the wholesale and tariff markets. In this paper, we investigate the feasibility of using learning strategies to improve the performance of our broker, SPOT. Specifically, we investigate the use of decision trees and neural networks to predict the clearing price in the wholesale market and the use of reinforcement learning to learn good strategies for pricing our tariffs in the tariff market. Our preliminary results show that our learning strategies are promising ways to improve the performance of the agent for future competitions.

Keywords: Smart Grid, Artificial Intelligence, Game Theory, Multi Agent System, Machine Learning

1 Introduction

The traditional energy grid lacks several important features such as effective use of pricing and demand response of energy, customer participation, and proper distribution management for variable-output renewable energy sources [1]. The smart grid has the potential to address many of these issues by providing a more intelligent energy infrastructure [2]. Researchers rely on rich simulations such as the *Power Trading Agent Competition* (Power TAC) [1] to explore the characteristics of future smart grids. In the Power TAC smart grid simulation, brokers participate in several markets including the wholesale market, the tariff market, and the load balancing market to purchase energy and sell it to customers. This game was designed as a scenario for the annual Trading Agent Competition, a research competition with over a decade of history [3].

The wholesale and tariff markets attempt to simulate existing energy markets such as the European or North American wholesale energy markets. The wholesale market is a "day ahead market," where the energy is a perishable good and it allows brokers to buy and sell quantities of energy for future delivery. Market structures like this exist across many different types of perishable goods. So, finding effective, robust, automated bidding strategies for these markets is an important research challenge.

The tariff market is where the major portion of energy purchased from the wholesale market is sold to consumers (e.g., households, offices, etc.). Energy is sold through tariffs offered by the brokers and a goal for the broker is to offer competitive tariffs that attract a large pool of consumers. The overall goal of each broker is to maximize its profit (e.g., by selling energy in the tariff market at a higher price than the purchase price of the energy in the wholesale market).

In this paper, we investigate the feasibility of using learning strategies to improve the performance of our broker, called *Southwest Portfolio Optimizing Trader* (SPOT), in Power TAC. We present our initial work on using decision trees to predict the clearing prices in the wholesale market and the use of an unsupervised reinforcement learning algorithm to learn good strategies for pricing our tariffs in the tariff market. Preliminary results show that these learning strategies hold promise, though we plan to investigate additional improvements to increase the competitiveness of the agent further.

2 Background: Power TAC

Power TAC models a competitive retail power market where the simulation runs for approximately 60 simulated days, and takes about two hours. Broker agents compete with each other by acting in three markets: wholesale market, tariff market and balancing market. It also includes a regulated distribution utility and a real location based population of energy customers during a specific period. Customer models include several entities such as households, electric vehicles, and various commercial and industrial models. Brokers participating in the simulation try to make profit by balancing the energy supply and demand as accurately as possible. By efficiently managing stochastic customer behaviors, weather-dependent renewable energy sources, the broker with highest bank balance wins the competition [4]. SPOT participated in the 2015 Power TAC competition. The table below shows results of the 11 participating agents in 2015 across games with varying numbers of competing brokers.

Name	3 Brokers	9 Brokers	11 Brokers	Total	Total (Normalized)
Maxon15 (1st)	186159	3667524	80687243	84540925	3.402
TacTex15 (2nd)	488341	5196258	38755591	44440191	2.221
CUHKTac (3rd)	556792	4000749	35070699	39628240	1.927
AgentUDE	-14748	1162481	52098550	53246283	1.597
Sharpy	-6459	2586534	45130820	47710895	1.564
COLDPower	307197	1334765	14309076	15951038	0.371
cwiBroker	-461511	-1650580	41663592	41663592	0.343
Mertacor	-23099	-139344	32199	-130244	-0.786
NTUTacAgent	-1533793	-10416019	43469971	31520159	-2.202
SPOT	-1570860	-2361785	7521196	3588551	-2.327
CrocodileAgent	-2981460	-13915197	-3318695	-20215352	-6.111

Table 1. Power TAC 2015 Final Round Results

We only had a couple of months of development before the 2015 tournament, so the main goal was to participate competently without major errors. Overall, our agent achieved this objective, but was not yet competitive with the top agents in the competition. The 2015 agent had preliminary implementation of some of the ideas we describe here, but we have since worked to improve the performance of the agent by updating the learning strategies and decision-making components of the agent.

2.1 Wholesale Market

The wholesale market functions as a short-term spot market for buying and selling energy commitments in specific timeslots, where each timeslot represents a simulated hour. At any point in the simulation, agents can participate in auctions to trade energy for the next 24 hours, so there are always 24 active auctions. These auctions are periodic double auctions, similar to those used in European or North American wholesale energy markets [5]. Each simulation begins with 14 days pregame data (bootstrap data), which includes data on customers, the wholesale market, and weather data based on the default broker. Brokers can submit bids (orders to buy energy) and asks (orders to sell energy), represented by a quantity and an optional limit price. In addition to the bids of the brokers, several large gencos also sell energy on the wholesale market. The simulation clears the bids by matching buy and sell orders, and determines the clearing price for each auction every day. If the minimum ask price has a higher value than the maximum bid price, then the market does not clear.

The main problem we consider here is learning to predict the clearing prices of these auctions, which can be used by the agent to implement an effective bidding strategy. Previous agents in both Power TAC and earlier TAC competitions have considered similar price prediction problems. AstonTAC is a Power TAC agent that uses a Non-Homogeneous Hidden Markov Model (NHHMM) to forecast energy demand and price [6]. This was the only agent in that competition that was able to buy energy at a low price in the wholesale market and keep energy imbalance low. TacTex13, winner of 2013 Power TAC competition, uses a modified version of Tesauro's bidding algorithm, where they modeled the sequential bidding process as a Markov Decision Process (MDP) for the wholesale market [7]. In the TAC/SCM game, Deep Maize used a Bayesian model of the stochastic demand process to estimate the underlying demand and trend. It employs a k-Nearest-Neighbors technique to predict the effective demand curves from historical data, self-play games data, and the current game data [8].

2.2 Tariff Market

The Power TAC environment offers the ability for brokers to issue several different types of tariffs three times per simulation day. Each tariff may be as complex or simple as the broker desires though each tariff can only target a single power type such as consumption or production. The most simple type of tariff one may issue is a flat rate tariff that offers a single price per kWh to subscribers. From there, the tariff may be augmented with signup bonuses, or a minimum subscription duration and early termination fee. Tariffs may also be customized to offer tiered usage pricing, time of use pricing, or a daily fee in addition to usage pricing. Tariffs may be issued, revoked, or

modified at any time, though the new tariffs will only become available for subscription at the designated 6 hour intervals. A tariff is modified by publishing a new tariff with the superseding flag set to the old tariff and then revoking the old tariff. For the purpose of the experiments outlined in this paper, our broker issued a single, simplified flat rate tariff at the beginning of the game and modified it throughout the simulation by superseding the past tariff and revoking the past tariff.

In order to publish the optimal tariff so that we gain both the most subscribers, henceforth referred to as market-share, and the greatest net balance, we utilized an unsupervised reinforced learning technique. This technique is chosen because we want the agent to be able to learn to react in such a way that gains the best possible reward with little interaction from the researchers. To achieve this goal we modeled this problem as a Markov Decision Process (MDP) [9] and utilize the Q-Learning algorithm [10] to discover the optimal policy. Q-Learning involves an iterative process whereby the SPOT agent plays many simulations constantly updating the Q-Value for a given state, action pair. Q-Learning will continue to improve the Q-Values until a convergence is obtained where the Q-Values for each state, action pair change very little per iteration. In order to expedite the convergence of the Q-Learning algorithm, we implemented a distributed system that allowed many simulations to be run simultaneously.

3 Learning in the Wholesale Market

Our baseline broker used a moving average price prediction based on the price history of the agent. To predict a new price for a week ahead specific hour price, the baseline agent uses a weighted sum of the current hour's clearing price, yesterday's predicted clearing price for that specific hour and 6 day ahead same hour predicted price. We have experimented with three different machine learning methods to predict clearing prices in the wholesale market: (*i*) REPTree (a type of decision tree) [11], (*ii*) Linear Regression, and (*iii*) Multilayer Perceptron (a type of neural network). We have also investigated a variety of different features for training the predictors. These include 8 price features that capture information about the recent trading history, such as the clearing price for the previous hour and the prices for the equivalent time slot in the previous day and week. We also include the weather forecast and time of day because the energy production of renewable energy producers (e.g., solar) depends on these factors. The number of participants in the game is included because the amount of competition affects the market clearing price. Finally, we include a moving average of the prices as a convenient way to capture an aggregate price history.

To generate training data, we use simulations with a variety of agent binaries from previous tournaments, as well as a variety of different bootstrap initialization files. We train our models using Weka [12], and evaluate their ability to predict market clearing prices based on the mean absolute prediction error only for auctions that clear (we do not include auctions that do not clear in the error calculations). In the following experiments, we investigate the performance of the models in several areas, including how well they generalize to new agents, different numbers of agents, and how important the different features are to the performance of the predictors.

3.1 Prediction Accuracy Comparisons

We begin with a basic evaluation of the prediction accuracy of the learned models. One of the most significant factors we discovered that influences the accuracy of the models is how we handle auctions that do not clear. In many cases, an auction will have no clearing price due to a spread between the bid and ask prices, which results in the simulation returning null values for these prices. This causes significant problems with the price features we use, as well as the final error calculations. To improve this, we calculate an estimated clearing price for auctions that do not clear by taking the average of lowest ask price and the highest bid price. Figure 1 shows the prediction errors during the course of a single simulation for two different REPTree models trained on 20 games, one with estimated clearing prices and the other without. We also include the errors for a simple moving average price predictor as a baseline for comparison. Each data point shows the average error for all auctions in a window of five timeslots. The data show that both REPTree models outperform the moving average predictor, but the version with estimated clearing prices is dramatically better, and produces much more consistent predictions throughout the entire game. Next, we compare the performance of the three

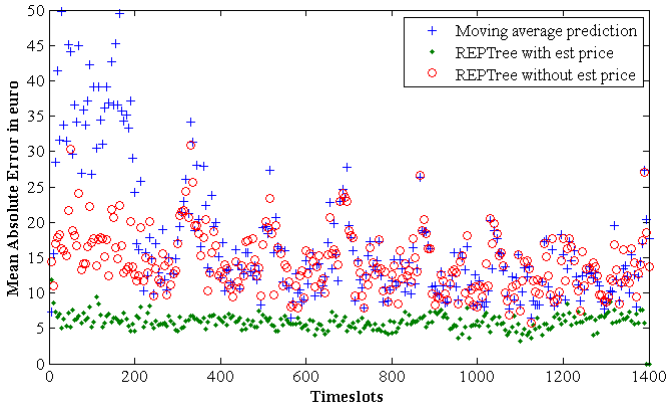


Fig. 1. Effect of Clearing Price Estimation

different learning methods with different amounts of training data ranging from 5 to 20 games. We evaluated a variety of different configurations of hidden layers for the Multilayer Perceptron model; only the best one is shown here (MP-20-20, i.e., 2 layer neural network with 20 nodes in each layer). Figure 2 shows the average mean absolute error for the different models based on 5 games of test data. The results show that the decision tree model makes good predictions compared to other models. The decision tree model slowly improves according to the number of games while other models do not show this trend. The default Multilayer Perceptron (1 layer with 18 nodes) with estimated prices shows some improvement in the initial number of games than REPTree but finally looses to REPTree in the 20 game model. In all cases, the models with estimated clearing prices are much better than models without estimated prices.

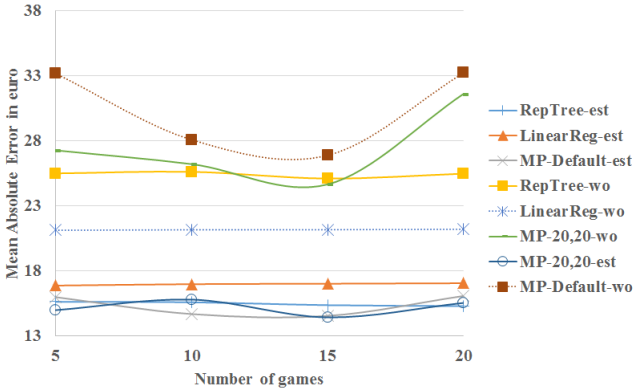


Fig. 2. Comparison of Several Prediction Models by Number of Games

3.2 Evaluation with different agents

In the Power TAC competition, broker agents play many games against different opponents with varying strategies. Here, we test how well our predictors generalize to playing new agents that are not in the training data. We test our models on games of the same size, but varying one of the agents in the game between AgentUDE15, cwiBroker15 and TacTex14. All the predictor model are generated from the training dataset where AgentUDE is used. Figure 3 shows the average results for each of the learning methods in the three different agent environments. The REPTree predictor consistently does better

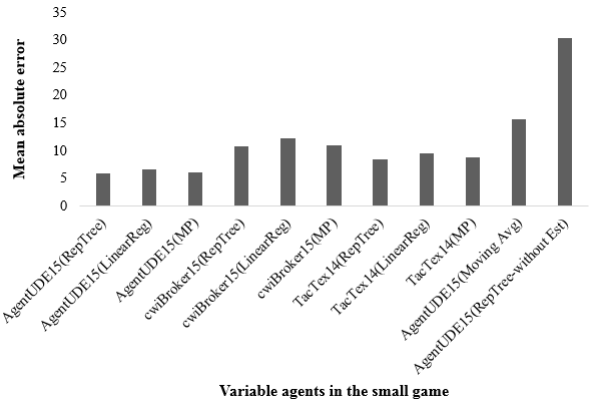


Fig. 3. Comparison of Several Prediction Models

than others, though there are differences depending on the pool of opponents. We can also see that the models do best against AgentUDE (which was in the training set), and there is a significant decrease in accuracy when playing either cwiBroker or TacTex. Further work is needed to help the models generalize better to new opponents.

3.3 Different number of agents

In the competition, broker agents must play in games with varying numbers of opponents. We experiment with different number of brokers in the games, ranging from 3 to 7 brokers. We focus here on the REPTree predictor since it performs better than the others consistently in previous experiments. The 5 agent predictor models trained on data generated from SPOT(Baseline), AgentUDE15, cwiBroker15, SampleBroker, Maxon14 and the 7 agent predictor models use data from SPOT(Baseline), AgentUDE15, cwiBroker15, SampleBroker, Maxon14, Maxon15, COLDPower and CrocodileAgent15. The test data uses the same agents. We also trained a predictor based on a mixed dataset that included the same number of training games, but with a combination of 3, 5, and 7 agent games. The data in Figure 4 shows that, in each case, the model trained on

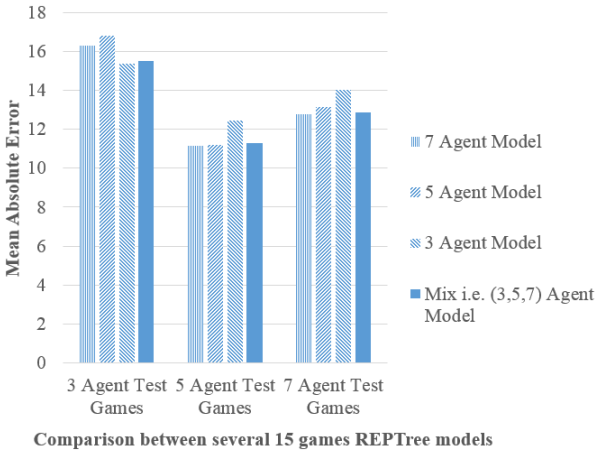


Fig. 4. Comparison for Different Number of Agents

the correct number of agents has the best performance. However, we also note that the mixed model performs very well in all three cases. Table 2 shows the average error of the predictor models over the 3 different test game data, and demonstrates that the average error for the mixed model is better than any of the other three models.

7 Agent	5 Agent	3 Agent	Mixed Agent
13.406	13.714	13.958	13.225

Table 2. Average Error for the Various Agent Models

3.4 Using Price Predictions for Bidding

We took the best performing predictor from our experiments (REPTree) and tested whether using these predictions could improve performance for a basic bidding strategy. This strategy attempts to target auctions where the clearing price is predicted to be low, and to buy a higher volume of the needed energy in those specific auctions. Figure 5 shows that using the new predictions and bidding strategy the agent is able to buy a high volume of the needed energy when the average clearing price is lowest against the champion agent Maxon15.

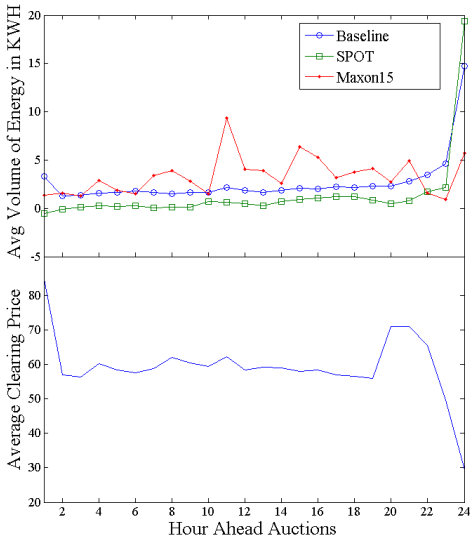


Fig. 5. Comparison for Wholesale Bidding Strategies

3.5 Feature evaluation

To evaluate which features are the most important for the predictions we used ReliefFAttributeEvaluation [13] and the Ranker method in Weka to rank our 18 features. We also used the ClassifierSubsetEval method and best-first search to get the best subset of features from all the features. Table 3 shows the top 7 features using the ranker algorithm and the best subset of features using the ClassifierSubsetEval method. We ran the subset evaluation on 5, 10, 15, and 20 games and, for all cases, we found a consistent subset of seven features. The features such as temperature, day of a month, month of a year, number of participants are ranked low and also out of the best subset. We could potentially discard these types of features while training a predictor model. From the ranked feature column, we see that price features are very important for the REPTree predictor model. So, adding additional features of this type may improve performance.

Ranked Features	Subset Evaluation
PreviousHourN_1Price	YesterdayClrPrice
PrevHourClrPrice	PreviousHourN_1Price
PredictedClrPrice	PredictedClrPrice
YesterdayClrPrice	PrevHourClrPrice
AWeekAgoN_1Price	Day
YesterdayN_1Price	HourAhead
PrevOneWeekClrPrice	CloudCoverage

Table 3. Feature Evaluation

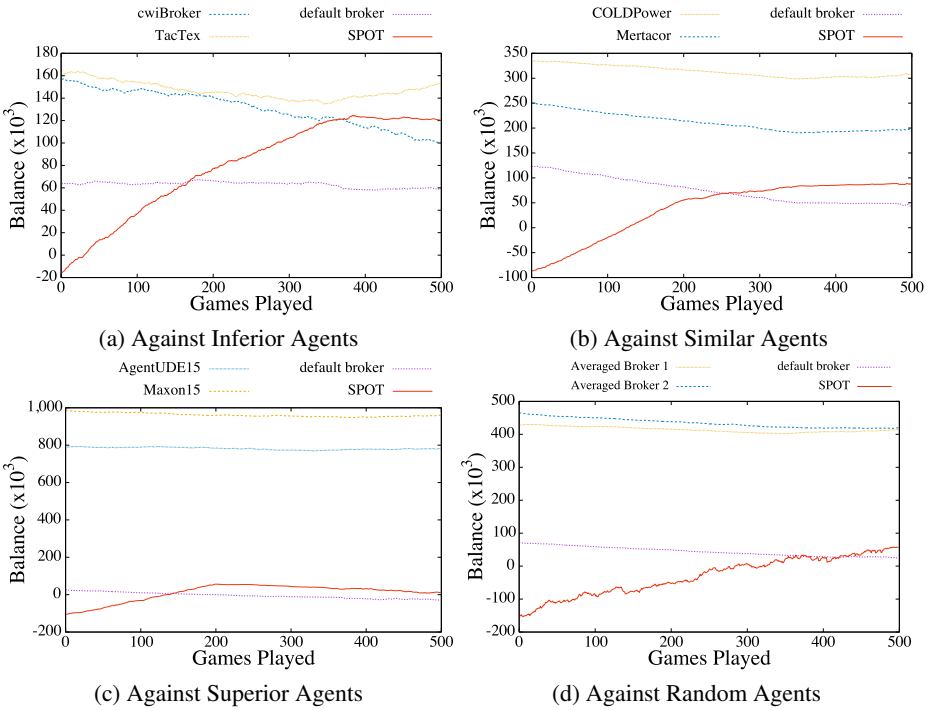
4 Learning in the Tariff Market

We describe how we formulate our problem in the tariff market as a *Markov Decision Process* (MDP) [9] and use Q-learning [10] to learn the optimal policy.

4.1 Formulating the Problem as an MDP

Recall that in the tariff market, the goal is to design tariffs that will result in the largest profit for the broker agent. In this paper, we investigate a restricted version of the problem, where we assume that the broker can only offer flat-rate tariffs, i.e., the price per kWh is uniform across all time steps. However, the broker can vary the price of the flat-rate tariff and the objective is still to maximize the profit of the broker. This problem can be formulated as a *Markov Decision Process* (MDP) [9], defined by the tuple $\langle \mathbf{S}, s_0, \mathbf{A}, \mathbf{T}, \mathbf{R} \rangle$:

- A set of states $\mathbf{S}$. In our problem, we define the set of states to be all possible pairs of $\langle MS, Bal \rangle$, where MS is the percentage of market share controlled by our agent (i.e., the percentage of customers that are subscribing to our agent) and Bal is the overall profit or loss since the start of the simulation (i.e., the amount of money in the "bank"). We discretized MS from 0% to 100% in increments of 5% and Bal from $-$€2,000,000 to €8,000,000 in increments of €20,000.
- A start state $s_0 \in \mathbf{S}$. In our problem, the start state is always $\langle 0\%, e0 \rangle$ since the agent does not have any subscribers to its tariff and starts with no initial profit or loss.
- A set of actions $\mathbf{A}$. In our problem, the first action of the agent is to publish a new flat-rate tariff at €15 per kWh. Subsequent actions are from the following set of actions:
  - $\uparrow$: Increase the price of the tariff by €2.00 per kWh. This is implemented by publishing a new tariff at the higher price and revoking the previous lower-priced tariff.
  - $\leftrightarrow$: Keep the price of the tariff. This is implemented by not publishing or revoking any tariffs.
  - $\downarrow$: Decrease the price of the tariff by €2.25 per kWh. This is implemented by publishing a new tariff at the lower price and revoking the previous higher-priced tariff.
- A transition function $\mathbf{T} : \mathbf{S} \times \mathbf{A} \times \mathbf{S} \rightarrow [0, 1]$ that gives the probability $T(s, a, s')$ of transitioning from state s to s' when action a is executed. In our problem, the transition function is not explicitly defined and transitions are executed by the Power TAC simulator.



(a) Against Inferior Agents

(b) Against Similar Agents

(c) Against Superior Agents

(d) Against Random Agents

Fig. 6. Convergence Rates

- A reward function $\mathbf{R} : \mathbf{S} \times \mathbf{A} \times \mathbf{S} \rightarrow \mathbb{R}^+$ that gives the reward $R(s, a, s')$ of executing action a in state s and arriving in state s'. In our problem, the reward is the gain or loss in profits of the agent, determined by the Power TAC simulator.

A "solution" to an MDP is a policy π, which maps states to actions. Solving an MDP is to find an optimal policy, that is, a policy with the largest expected reward.

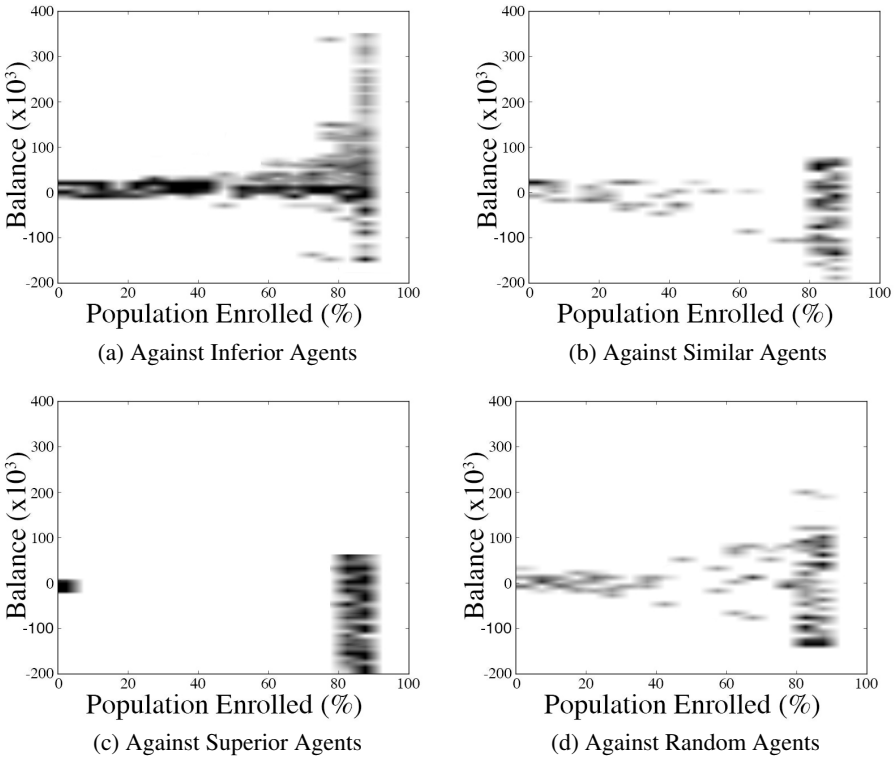
4.2 Learning Optimal Tariff Prices

We now describe how to learn the optimal policy of the MDP using Q-learning [10]. We initialize the Q-values of all state-action pairs $Q(s, a)$ to 1,000,000 in order to better encourage exploration [10] and use the following update rule to update the Q-values after executing action a from state s and transitioning to state s':

$$Q(s, a) \leftarrow \alpha \left\{ R(s, a, s') + \gamma \cdot \max_{a' \in \mathbf{A}} Q'(s', a') \right\} \tag{1}$$

where $\alpha = 0.9$ is the learning rate and $\gamma = 1.0$ is the discount factor.

Parallelizing the learning process: In order to increase the robustness of the resulting learned policy, we executed the learning algorithm with 10 different simulation bootstrap files [14]. The different bootstrap files may contain different combinations of types of users, with different energy consumption profiles, energy generation capabilities, etc.



(a) Against Inferior Agents

(b) Against Similar Agents

(c) Against Superior Agents

(d) Against Random Agents

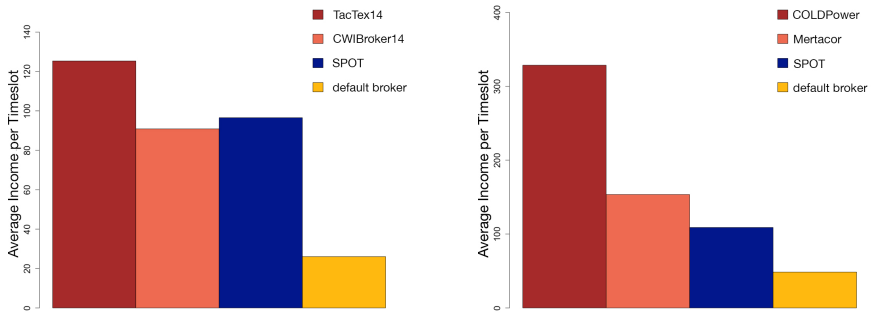
Fig. 7. Explored States

In order to speed up the learning process, we parallelize the Q-learning algorithm by running multiple instances of the simulation. We run the simulations in groups of 10 instances, where each instance in the group uses one of the 10 unique bootstrap files. Instead of using and updating their local Q-values, all these instances will use and update the same set of Q-values stored on a central database. Once the simulation of one of the instances ends (the Power TAC simulation can end any time between 1440 to 1800 simulated hours), it will restart with the same bootstrap file from the first time step again.

4.3 Experimental Results

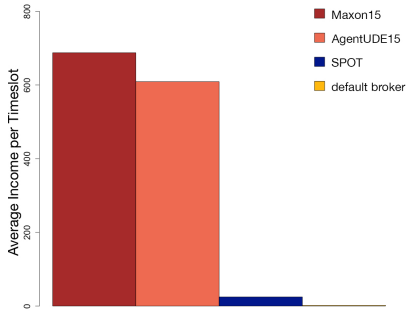
In our experiments, we learn policies against two opposing agents; this scenario corresponds to the 3-agent scenario in the previous Power TAC competition. We characterized possible opposing agents according to their relative competitiveness in the previous years' Power TAC competitions. We learned four different sets of Q-values and, equivalently, four different sets of policies against four different types (in terms of their competitive level) of opposing agents:

- SUPERIOR AGENTS: AgentUDE15 and Maxon15.
- SIMILAR AGENTS: Mertacor and COLDPower.

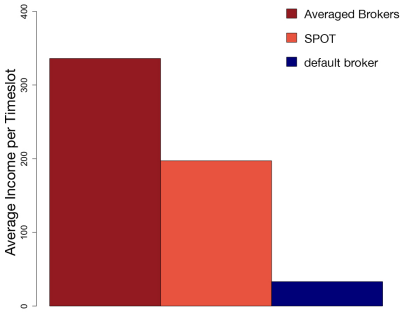


(a) Against Inferior Agents

(b) Against Similar Agents



(c) Against Superior Agents



(d) Against Random Agents

Fig. 8. Comparison against Opposing Agents

- INFERIOR AGENTS: TacTex14 and CWIBroker14.
- RANDOM AGENTS: Two randomly chosen agents from the set of 6 agents above.

Figure 6 shows the convergence rates for all of the scenarios, where the y-axis shows the final balance at the last time step for each iteration. SPOT is able to learn better policies and improve its final balance with more iterations. To reach convergence, SPOT takes various numbers of iterations according to opponents. SPOT sees the most variance in games where the opponents are randomized. Against a set list of opponents, policy convergence is reached in a limited number of iterations. For example, after approximately 200 iterations convergence is reached against superior brokers.

Figure 7 illustrates the explored states in the same scenarios. The color of each state represents the number of times the actions for each state were explored, ranging from black, where all three actions were explored the most, to white, where no actions were explored. The figure shows that more states and actions were explored against inferior agents than against superior agents. Additionally, these results also explain the performance of the agent; against superior agents, our agent was very limited in the states it was able to explore, most times being unable to gain more than 5% of the market share, and when it did get a significant amount it was often at a loss. Thus, it took its best actions and maintained a balance of approximately €50,000.

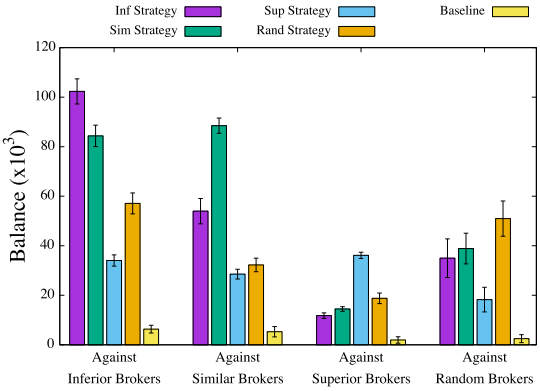


Fig. 9. Profit in the Tariff Market per Timeslot

We evaluated the learned policies against the same set of opposing agents. Figure 8 shows the profit in the tariff market alone of each agent over the various time steps. These results are averaged over 5 different bootstrap files (different from those used in the learning process) and 3 runs per bootstrap file. These results are consistent with the final converged results shown in Figure 6, where our agent does better against inferior agents than against superior agents.

Figure 9 plots the performance of our agent with each of the four learned policies in addition to an agent with no learned strategy against each pair of opposing agents. Not surprisingly, the results show that the agent with the policy learned through playing against a specific pair of opponents does best when playing against the same pair of opponents (e.g., the agent with the policy learned through playing against superior agents does better than other policies when playing against superior agents). The policy learned through playing against random agents is more robust towards different opponent types, especially compared against the policy learned through playing against superior agents.

5 Conclusions and Future Work

A forward-looking policy is needed between the tariff and wholesale strategy to make consistent profit. The preliminary results in this paper show that the application of learning strategies to broker agents within Power TAC have immediate benefits in both the wholesale and tariff markets separately. However, a more comprehensive study is needed to better harness the strength of these learning approaches. Currently, the evaluations in the wholesale and tariff markets are conducted independently of each other. Therefore, future work includes learning good bidding strategies such as Monte Carlo Tree Search in the wholesale market by taking into account the predicted clearing prices as well as empirically evaluating the coupling effects of the learning strategies between the wholesale and tariff markets.

6 Acknowledgment

This research is partially supported by NSF grant 1345232. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the sponsoring organizations, agencies, or the U.S. government.

References

[1] Ketter, W., Peters, M., Collins, J.: Autonomous agents in future energy markets: the 2012 power trading agent competition. In: Proceedings of the AAAI Conference on Artificial Intelligence. (2013) 1298–1304

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

[3] Wellman, M.P., Greenwald, A., Stone, P., Wurman, P.R.: The 2001 trading agent competition. Electronic Markets **13**(1) (2003) 4–12

[4] Ketter, W., Collins, J., Reddy, P.P., Weerdt, M.D.: The 2015 power trading agent competition. ERIM Report Series Reference No. ERS-2015-001-LIS (2015)

[5] Ketter, W., Collins, J., Reddy, P.: The 2012 power trading agent competition. ERIM Report Series Research in Management (No. ERS-2012-010-LIS) (2012)

[6] Kuate, R.T., He, M., Chli, M., Wang, H.H.: An intelligent broker agent for energy trading: An MDP approach. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI). (2013) 234–240

[7] Urieli, D., Stone, P.: Tactex'13: a champion adaptive power trading agent. In: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2014) 1447–1448

[8] Kiekintveld, C., Miller, J., Jordan, P.R., Wellman, M.P.: Controlling a supply chain agent using value-based decomposition. In: Proceedings of the ACM Conference on Electronic Commerce (EC). (2006) 208–217

[9] Mausam, Kolobov, A.: Planning with Markov Decision Processes: An AI Perspective. Morgan & Claypool (2012)

[10] Abounadi, D.B., Borkar, V.S.: Learning algorithms for markov decision processes with average cost. SIAM Journal on Control and Optimization **40**(3) (2001) 681–698

[11] Elomaa, T., Kaariainen, M.: An analysis of reduced error pruning. Journal of Artificial Intelligence Research (2001) 163–187

[12] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update; sigkdd explorations, volume 11, issue 1. `http://www.cs.waikato.ac.nz/ml/weka/` (2009–2015)

[13] Robnik-Sikonja, M., Kononenko, I.: An adaptation of relief for attribute estimation in regression. In: Proceedings of the International Conference on Machine Learning (ICML). (1997) 296–304

[14] Lauer, M., Riedmiller, M.A.: An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In: Proceedings of the International Conference on Machine Learning (ICML). (2000) 535–542