Multi-armed Bandit Based Tariff Generation Strategy for Multi-Agent Smart Grid Systems

Sanjay Chandlekar[1], Easwar Subramanian[2], and Sujit Gujar[1]

[1] International Institute of Information Technology (IIIT), Hyderabad, India
sanjay.chandlekar@research.iiit.ac.in
sujit.gujar@iiit.ac.in
[2] TCS Innovation Labs, Hyderabad, India
easwar.subramanian@tcs.com

Abstract. The emergence of smart grid technology has opened the door for wide-scale automation in decision-making. A distribution company, an integral part of a smart grid system, has to procure electricity from the wholesale market and then sell it to customers in the retail market by publishing attractive tariff contracts. It can deploy autonomous agents to make decisions on its behalf. In this work, we describe the tariff contracts generation strategy of one such autonomous agent, which is based on a Contextual Multi-armed Bandit (ConMAB) based learning technique to generate tariff contracts for various types of customers in the retail market of smart grids. We particularly utilize the Exponential-weight algorithm for Exploration and Exploitation (EXP-3) for ConMAB-based learning. We call our proposed strategy GenerateTariffs-EXP3. Our previous work shows that maintaining an appropriate market share in the retail market yields high net revenue. Thus, we first present a game-theoretic analysis that determines an optimal level of market share. Then we train our proposed strategy to achieve and maintain the suggested level of market share by adapting to the market situation and revising the tariff contracts periodically. We validate our proposed strategy in PowerTAC, a close-to real-world smart grid simulator, and showcase that it is able to maintain the suggested market share.

Keywords: Contextual Multi-armed Bandit (ConMAB), EXP3, Smart Grids, Tariff Generation in Multi-agent Environment, PowerTAC

1 Introduction

Recent years have seen rapid growth in smart grid technology. Some developed nations have already adopted smart grid technology to replace the conventional grid system. Fundamentally, just like a conventional grid, a *smart grid* is also an electricity network that supplies electricity to customers; however smart grid enables two-way digital communication where customers can also communicate with electricity providers. It also allows for monitoring, analysis, control and

communication between participants to improve the efficiency, transparency, and reliability of the system [14].

The smart grid system comprises the wholesale and retail markets, transmission lines, and distribution company (DC) as the prominent players. The DCs play a significant role in smart grid operations and are responsible for the efficient functioning of the system. The major tasks of DC are to buy electricity from the wholesale market, sell electricity to retail customers by generating lucrative yet profitable tariff contracts, and manage the supply-demand balance in the smart grid system. The transmission lines are responsible for electricity transmission from GenCos to retail customers.

The retail market of a smart grid, which is the focus of this work, incorporates various types of customers like households, office spaces, villages, producers (customers having solar panels or wind turbines), electric vehicles, battery storage, and a few others. Some of these customers have the capability to change their electricity usage pattern based on the signals from the DC, commonly in the form of tariff contracts. To cater to the variety of customers, tariff contracts too can be of multiple types. For example, (i) Fixed Price Tariff (FPT) having the same rate values for all hours in a day/week, (ii) Time of Use (ToU) tariff having different rate values for different hours in a day/week, (iii) Tier tariffs having different rate values corresponding to different usage slots, (iv) variable tariffs where rate values can change dynamically, or (v) combination of any of the above tariff types. DCs decide the appropriate tariff types and tariff rate values for the customers in its portfolio.

The smart grid system is quite complex in nature, and it is practically impossible to test or validate the new strategies on the real-world smart grid system. Thus, in order to aid in smart grid research, Power Trading Agent Competition (PowerTAC) designed a close-to-real-world smart grid simulator [4]. PowerTAC simulates all the crucial elements of a smart grid system mentioned above. In PowerTAC, DC are commonly known as *electricity broker* or *broker* or *agent*. PowerTAC embodies a variety of customer models to represent the wide variety of customers as seen in the real world. It supports all kinds of tariff contracts mentioned earlier. Furthermore, PowerTAC introduces a balancing market that handles the real-time balancing of supply and demand. It penalizes agents in case of an imbalance in their portfolio.

The smart grid technology enables the use of adaptive autonomous agents to make crucial decisions on behalf of DC, and a simulator like PowerTAC helps analyze the effectiveness of such agents. To this end, PowerTAC organizes an annual tournament where participating teams design an autonomous agent that acts as DC and makes all the decisions in the simulated smart grid environment. The agents are required to design suitable strategies for the wholesale, retail and balancing markets. In this work, we specifically focus on the emphtariff contract generation problem in the retail market of the smart grid. To generate a new tariff contract, an agent needs to decide the tariff contract type and the tariff contract's rate values. The tariff contracts are public information; any agent and a customer in the simulation can see all the active tariffs in the retail market.

Thus, if an agent does not adapt to the changing market situation and does not update its tariff contracts periodically, any opponent agent can offer better tariff contracts and take away all the customers. Thus, it is paramount to update tariff contracts periodically, which can be done using either heuristic-based approaches or learning-based approaches.

In the PowerTAC literature, authors have proposed gradient-based MDP-based strategies [2], optimization strategies [15], and genetic algorithm based approaches [16] to publish tariff contracts in the retail market. The experimental evidence suggests that the seemingly optimal class of strategies of capturing all the market share may suffer from high grid balancing penalties as all the customers are subscribed to one agent, and that agent alone has to bear the total penalty for the grid imbalance. To remedy this, agent TUC_TAC proposed a strategy aimed at acquiring only half the retail market share [10]. However, all the above strategies except TUC_TAC sought to maximize the revenue/profit without explicitly controlling the agent's market share. Furthermore, the majority of the above retail strategies, including TUC_TAC, have been generic and are not effectively specialized for different player configurations and therefore fail to maintain performance across different player configurations.

To overcome the above problems, we, team *VidyutVanika*, designed an autonomous agent that emerged as the champion of the PowerTAC tournament in the year 2021 and 2022 [1]. The tariff strategy of our agent is inspired by the game theory literature that decides the optimal market share for various player configurations and uses heuristic-based techniques to achieve and maintain that market share during the simulation. In this work, we replace our heuristics-based strategy with a learning-based strategy to achieve similar performance. For that, we design a tariff strategy that learns to achieve and maintain the optimal market share. We model this problem by utilizing techniques derived from *Contextual Multi-armed Bandit (ConMAB)* and solve using the *Exponential-weight algorithm for Exploration and Exploitation (EXP-3)*. Our novelty lies in the formulation of the learning framework; as opposed to previous strategies that aim to maximize profit, we aim to maintain the optimal market share via a learning-based strategy which in turn reduces other costs and makes our agent profitable. We use ConMAB as its problem setting resembles the tariff generation problem in hand, where given a context, an agent has to pick an appropriate tariff (an optimal arm of ConMAB) that enables it to maintain the appropriate market share and, in turn, delivers higher returns. In summary, our contributions are as follows:

- We present game theoretical analysis to determine an optimal market share for various player configurations by modeling the PowerTAC games as two-player zero-sum games and calculate their mixed strategy Nash equilibrium.
- We propose a novel Contextual Multi-armed Bandit-based tariff contract generation strategy GenerateTariffs-EXP3, that learns to achieve and maintain the market share suggested by game theoretical analysis.
- We showcase the policies learned by the proposed strategy and its efficacy in maintaining the suggested market share during the PowerTAC games.

2 Related Work

Many approaches in the literature have been suggested to tackle the tariff generation problem, and a few have been implemented in PowerTAC as well. In the retail market of smart grids, techniques such as demand response, peak demand pricing, and learning-based approaches have been proposed to design competitive tariffs. Many multi-armed bandit-based strategies have been proposed to publish tariffs in the smart grid domain. Most of this work focuses on demand response in a smart grid where customers are incentivized via tariffs to curtail their usages in response to electricity supplier's signals [3, 13, 7, 6, 9, 5].

In the past PowerTAC tournaments, Markov Decision Process (MDP) based strategies were most popular in the retail market. The past brokers like COLD Energy and VidyutVanika18, as well as Reddy & Veloso, modeled the decision process in the retail market as an MDP to generate tariff contracts [11, 12, 2]. In fact, both COLD Energy and VidyutVanika18's tariff strategies were motivated by Reddy & Veloso. In these approaches, the state space is constituted by market parameters such as market rationality, agent's portfolio status etc. and action space was designed with actions to increase or decrease the rate value of tariffs by a certain amount. The reward function was profit in the market. TacTex'13 employed a gradient-based optimization method for tariff generation, and AgentUDE17 utilized a genetic algorithm-based tariff strategy [15, 16]. However, all the above strategies incur high grid imbalance costs as they do not focus on the market share of customers in their portfolio. Agent TUC_TAC proposed a strategy to acquire only half the retail market share [10] for each type of game configuration. Motivated by TUC_TAC's idea, we designed a heuristic-based tariff strategy backed by game theoretical analysis to determine the optimal market for various game configurations [1]. Furthermore, instead of focusing on revenue/profit, we aimed to maintain the appropriate market share, which helped us earn high returns. However, none of the previous works present an equilibrium-based strategy that can be learned online in the retail market. The novelty of this work lies in designing a game-theory-inspired ConMAB-based retail strategy that *learns* to achieve and maintain equilibrium market share in the retail market.

3 PowerTAC Simulator: An Overview

PowerTAC is a simulation platform that mimics essential components of a smart-grid ecosystem comprising retail, wholesale, balancing markets, and distribution companies (DCs). The wholesale market consists of GenCo, which sells electricity via auctions; the balancing market manages the real-time balance of supply and demand. The retail market consists of state-of-the-art customer models that simulate real-world smart grid users, including consumers, producers, and storage users such as households, offices, villages, hospitals, and renewable energy producers. Storage customers use electric vehicles or batteries to store electricity and supply it to the grid on demand. PowerTAC allows deploying an autonomous

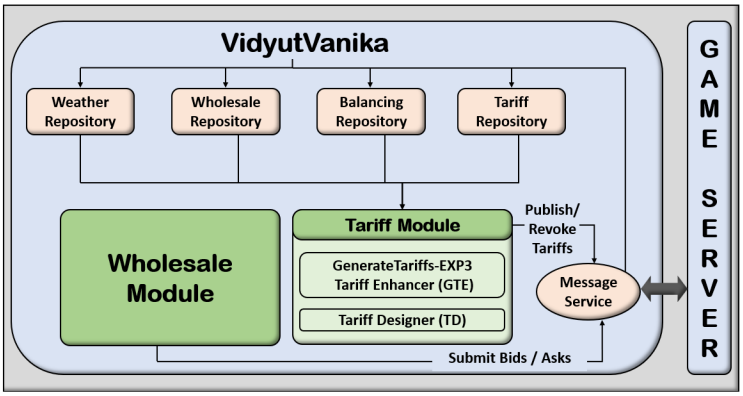

Fig. 1: System Architecture of VidyutVanika

agent to automate a DC's operations in retail, wholesale, and balancing markets to earn profits. PowerTAC also organizes an annual tournament in which numerous teams deploy autonomous brokers to compete in all three markets. The tournament consists of multiple games organized between agents in different player configurations and varying weather conditions, with each game lasting around 60 simulation days. During the game, an agent aims to develop a subscriber base in the retail market by offering competitive tariffs, such as FPT, tiered, variable or ToU, to sell energy bought in the wholesale market. Agents can also manage grid imbalances through subscriptions to storage customers. Agents update their tariffs periodically based on other available tariffs in the market, market and weather conditions, and customers' responses to previous tariffs. In the simulation environment, agents are provided with information that helps them make decisions. All agents in the retail market can see new and revoked tariffs and weather information. The final cash position of all brokers across games is aggregated to determine the tournament winner. A comprehensive simulator description is available in the 2020 PowerTAC specifications by Ketter et al. [4].

4 VidyutVanika (VV): Retail Module

In this section, we show the generic system architecture of our agent *VidyutVanika*, which emerged as the champion in the last two editions of the PowerTAC tournaments, namely PowerTAC'21 and PowerTAC'22. As shown in Fig. 1, VidyutVanika incorporates a wholesale module and a retail (tariff) module. It also has various repositories to store the important information received from the server. These repositories contain information about the weather, wholesale market procurement cost, all available tariffs in the market and customers' electricity usage patterns. In the current work, we only focus on the retail module; thus, we take our wholesale module as a black box that places bids in the wholesale mar-

Algorithm 1 TariffDesigner(avgPrice, powerType)

1: pattern ← DefineWeeklyTariffPattern().
2: $s[]$ ← DefineSurplusMultipliers(pattern)
3: find $normRate : \frac{\sum_{i=1}^{168} s_i * normRate}{168} =$ avgPrice
4: rate[i] ← $s_i * normRate, for i \in \{1, 2, ..., 168\}$
5: ToUTariff ← CreateTariff(rate, powerType)
6: **return** ToUTariff

ket auction and procures the required energy. We replaced the heuristic-based retail strategy used in the PowerTAC'21 and PowerTAC'22 tournaments with proposed ConMAB-based retail strategy, GenerateTariffs-EXP3.

As shown in the figure, the retail module consists of two submodules, namely, GenerateTariffs-EXP3 Tariff Enhancer (TE) and Tariff Designer (TD). The TE submodule comprises the proposed ConMAB-based tariff contract generation strategy, which is solved using the EXP-3 algorithm. This submodule observes the optimal market share for the ongoing game's player configuration by contacting the game theory module, then based on the ConMAB-based learning till that point, it picks the suitable action to enhance the current tariff. This TE sub-module calculates mean tariff rates that would maintain the appropriate market share. The TD sub-module designs weekly ToU tariffs by taking the mean rates suggested by TE as input. Below, we describe the details of the TD sub-module, while the details of the TE sub-module are deferred to the following sections.

Tariff Designer (TD): Algorithm 1 outlines TD, which is responsible for designing a weekly ToU tariff based on the average input price (*avgPrice*) received from TE. TD first generates a binary weekly tariff pattern using the *DefineWeeklyTariffPattern()* method, which identifies peak and non-peak hours by analyzing historical net market demand values retrieved from past PowerTAC tournaments. Peak hours are determined to be times of high demand, such as morning and evening hours. TD then uses the *DefineSurplusMultipliers()* method to set surplus multipliers s_i for each of the 168 hours in a week. These multipliers are greater than 1 for peak hours and 1 for non-peak hours. s_i depends on the peak magnitude observed from market demand data for peak hours. Thereafter, we calculate the *normRate*, which, after getting multiplied with s_i values of the week, results in *avgPrice* on an average. These *normRate* values with surplus s_i values are the rate values of the newly generated ToU tariff.

5 Game Theory to Determine Optimal Market-share

This section presents the game-theoretical analysis to decide an optimal market share for various player configurations of PowerTAC games, which is then used in the TE submodule to design suitable ToU tariffs. We show the analysis for three different player configurations of PowerTAC games, namely, 2-Player, 3-Player,

and 5-Player games. We construct a utility matrix for each player configuration by modeling the PowerTAC games as *two-player zero-sum games*, solving which results in an equilibrium market share. To assist the reader, we introduce a few definitions before proceeding further.

The below analysis is first presented in our previous work [1], where we presented the analysis briefly. Here, we include more details and present the complete analysis for all three player configurations under consideration, along with respective utility matrices. Furthermore, we utilized the below analysis to design a heuristics-based tariff strategy for our broker VidyutVanika during the PowerTAC'21 and PowerTAC'22 tournaments. The tariff strategy aimed to maintain the market share suggested by the game theory analysis using intelligent heuristics. In this work, too, we aim to maintain the suggested market share, albeit by following a more methodological way, that is, by incorporating the game theoretical analysis in the tariff strategy and framing the tariff contracts generation problem as a learning problem; and learning to improve tariffs *online* by looking at the market situation with the help of ConMAB-based techniques. A detailed description of the tariff strategy framework is included in Section 6.

Definition 1 (Mixed Strategy). *For player i, its mixed strategy σ_i is a probability distribution over the strategy set S_i, i.e., $\sigma_i(s_i), s_i \in S_i$ indicates the probability with which player i plays s_i.*

Definition 2 (Mixed Strategy Nash Equilibrium (MSNE)). *Given a N player game $\Gamma =< N, (S_i), (u_i) >$, a mixed strategy profile $(\sigma_1^*, ..., \sigma_n^*)$ is called a mixed strategy* Nash equilibrium *if, $\forall i \in N, \quad u_i(\sigma_i^*, \sigma_{-i}^*) \geq u_i(\sigma_i, \sigma_{-i}^*), \forall \sigma_i \in \Delta(S_i)$. σ_{-i}^* denotes mixed strategies of all players except i.*

The utility of the row player is defined in Equation 1, which is the difference between the average final cash positions of the row and column players. This way of modeling the utility matrix helps us to maximize the difference between VidyutVanika's average cash position and the opponent's average cash position, thereby helping VidyutVanika generate higher profits than opponents. As we formulate this as a zero-sum game, the column player gets negative of the utility calculated in Equation 1.

$$u_1(s_i, s_{-i}) = \frac{1}{T}\sum_{i=1}^{T} x_i - \frac{1}{n}\sum_{k=1}^{n}(\frac{1}{T}\sum_{i=1}^{T} y_{ik}) \tag{1}$$

In Eq. 1, x_i denotes the final cash balance of VidyutVanika in the game i, while y_{ik} denotes the final cash balance of opponent agent k in the game i and n denotes the number of opponent agents in the game. For our analysis, the average values are taken over $T = 5$ games.

In our modeling, we select VidyutVanika as the row player, and a subset of opponents, depending on the player configuration, act as a sole column player. The row player's (VidyutVanika's) strategy set is given by $S_1 = \{0\%, 15\%, 30\%,$

VV / Opp	(TT, VV18)	(TT, C)	(TT, VV20)	(TT, A)	(VV18, C)	(VV18, VV20)	(VV18, A)	(C, A)	(VV20, C)	(VV20, A)
0%	0.1103	-0.6891	-1.2093	-0.1409	-0.8984	-0.1235	0.3564	0.7303	-0.9107	0.608
15%	0.2327	-0.8492	-1.478	0.1224	-0.3161	0.8468	1.0907	0.6508	-0.4521	0.6338
30%	0.0913	-0.168	-0.4011	-0.208	-0.1865	0.9829	1.235	0.8528	-0.0197	0.8661
45%	0.8468	0.1589	0.1358	0.155	0.1415	1.3847	1.3087	0.8597	0.2026	0.8561
60%	1.069	0.0414	0.1811	0.2057	0.5651	1.3724	1.0288	0.8264	-0.2484	0.8409
75%	0.9615	-0.1678	-0.5645	0.1384	-0.0956	1.3274	1.0251	0.1766	-0.1712	0.3298
100%	0.7881	-0.5495	-1.0114	-0.5146	-0.6076	0.4744	-0.0569	-0.069	-1.3863	0.0263

Fig. 2: 3-Player Games Analysis (Utility Values in Millions)

45%, 60%, 75%, 100%}, where each element in the set $S1$ specifies the target market share that VidyutVanika has to maintain during the simulated games. We have five agents from past PowerTAC tournaments to act as opponents in our analysis, namely, TUC_TAC (TT) [10], VidyutVanika18 ($VV18$) [2], VidyutVanika20 ($VV20$), CrocodileAgent (C) and AgentUDE (A) [16]. The column player strategy set $S2$ depends on the player configuration. For example, in a 2-Player game configuration, we need only one opponent against VidyutVanika; thus, $S2 = \{TT, VV18, VV20, C, A\}$. Similarly, in a 3-Player game configuration, we need two opponents against VidyutVanika, which is to be selected from the available set of five agents; thus, total $5c2$ elements is the set $S2$ as shown in Figure 2.

VV / Opp	(TT, VV18, VV20,C)	(TT, VV18, VV20,A)	(TT, VV18, A,C)	(TT, A, VV20,C)	(A, VV18, VV20,C)
0%	-0.893	-0.298	-0.169	-0.156	1.737
15%	-0.199	-0.017	-0.205	-0.146	1.581
30%	0.112	-0.049	0.106	0.044	1.898
45%	-0.083	0.041	0.159	0.143	1.808
60%	-0.312	0.027	-0.288	-0.102	1.741
75%	-0.493	-0.228	-0.373	-0.409	1.025
100%	-0.498	-0.561	-0.188	-0.188	0.996

(a) 5-Player Games Analysis

VV / Opp	TT	VV18	VV20	C	A
0%	-1.115	-0.7268	-1.4139	-0.7265	-0.6337
15%	-2.9234	-1.7458	-2.028	0.3205	-1.6095
30%	-1.9241	-0.8775	-1.1025	0.3325	-0.3431
45%	-0.5181	-0.1804	0.0142	0.7307	0.2069
60%	-0.0596	0.5893	0.461	0.4369	1.3764
75%	-0.1405	1.3299	0.0818	-0.9286	1.8755
100%	-0.267	0.9995	-0.3381	-0.8462	0.7812

(b) 2-Player Games Analysis

Fig. 3: Games Analysis (Utility Values in Millions)

Equilibrium Calculation: Figures 3b, 2 and 3a show the utility matrices for 2-Player, 3-Player, and 5-Player configurations, respectively. Each cell describes the utility value, a cash difference in millions calculated by playing a set of T games. The same process is repeated for all the combinations of VidyutVanika's strategies ($S1$) and opponents' strategies ($S2$) to create the full utility matrix. Thereafter, we use Gambit [8] to solve the game and output the Nash Equilibrium. We found that each of the above three player configurations exhibits Mixed Strategy Nash Equilibrium (MSNE).

- **For 2-Player Configurations:** Based on Figure 3b, the utility matrix leads to Pure Strategy Nash Equilibrium of 60% market shares.
- **For 3-Player Configurations:** Based on Figure 2, the utility matrix leads to MSNE of randomizing between 45% and 60% market shares with probabilities 0.8 and 0.2, respectively, which results in equilibrium market share of 48% $(0.8 * 45 + 0.2 * 60)$.

– **For 5-Player Configurations:** Based on Figure 3a, the utilitys matrix leads to MSNE of randomizing between 30% and 45% market shares with probabilities 0.43 and 0.57, respectively, which translates to equilibrium market share of 38.55% $(0.43 * 30 + 0.57 * 45)$.

The same results can be seen visually as well; the green-shaded regions in the figures show the strategies having the higher utilities $u_1(\sigma_1^*, \sigma_{-1}^*)$ than the remaining strategies $u_1(\sigma_1, \sigma_{-1}^*)$ for row-player VidyutVanika, which leads to above-calculated MSNEs.

Adopting Equilibrium in PowerTAC Games: The above analysis suggests how we should randomize to achieve equilibrium market share. However, due to the stochasticity of the PowerTAC simulation and customer models, it is not easy to maintain one particular market share across different games. Hence, we aim to maintain market share within specific bounds ($[middle, high]$). Thus, in our experiments, we treat the above-calculated equilibrium market shares as the higher bounds ($high$) on the desired market share. We further define the middle bounds ($middle$), which is $0.7 * high$. We aim to maintain the market share between $middle$ and $high$, and thus, to train GenerateTariffs-EXP3, we give $0.85 * high$ $((1 + 0.7)/2 = 0.85)$ as the target optimal market share. So, for 2-Player, 3-Player, and 5-Player configurations, target optimal market shares for GenerateTariffs-EXP3 are 51%, 40.8%, and 32.3%, respectively.

6 Tariff Strategy: A Contextual MAB Approach

In the previous section, we showcase how we determine the optimal market for various player configurations. Based on our previous work, we also stated that maintaining a market share close to the optimal market share is sufficient to achieve effective profits in the market. Motivated by this, in this section, we showcase the formulation of the proposed GenerateTariffs-EXP3. The proposed strategy is modeled as a Markov Decision Process (MDP) consisting of a tuple $< S, A, P, R >$. S represents the state space of the MDP, A denotes the action space and R denotes the rewards of the MDP. P represents the transition probabilities of the MDP, that is, the probability with which MDP transition to the next state by taking action in the current state. However, the model does not know the transition probabilities. To learn the optimal action in each state (called a policy) in the absence of transition probabilities, we use ConMAB techniques along with the EXP-3 algorithm. Below we describe how the MDP is formulated and optimal policies are learned.

6.1 State Space

Here, we define the state space of the GenerateTariffs-EXP3 We construct state space depending on the difference between the current market share (CMS)

of the GenerateTariffs-EXP3 and the optimal market share (OMS) suggested by the game theory module in Section 5. Let us denote the difference between both the market shares by Δ, so

$$\Delta = (OMS - CMS)$$

. We categorize Δ into seven buckets, as shown below.

- State 0: $|\Delta| \leq OMS * 0.1$
- State 1: $\Delta > OMS * 0.1$ and $\Delta \leq OMS * 0.4$
- State 2: $\Delta > OMS * 0.4$ and $\Delta \leq OMS * 0.7$
- State 3: $\Delta > OMS * 0.7$
- State 4: $-\Delta > OMS * 0.1$ and $-\Delta \leq OMS * 0.4$
- State 5: $-\Delta > OMS * 0.4$ and $-\Delta \leq OMS * 0.7$
- State 6: $-\Delta > OMS * 0.7$

The above state space is designed in such a way that it gives the reflection of the GenerateTariffs-EXP3's current situation in the tariff market. For example, suppose the OMS for a game configuration is 50%, then the State 0 occurs when the broker's CMS is within ±5% difference of the OMS (i.e., between 45% to 55%). Similarly, State 1 happens when the broker's CMS is lower than the OMS, and the difference between OMS and CMS ($OMS - CMS$) is more than 5%, but less than 20% (between 30% to 45%). The states 1, 2 and 3 represent the situation when the broker's CMS is lower than the OMS. Replicating the similar logic for the other side as well, states 4, 5, and 6 represent the situation when the broker's CMS is higher than the OMS. The State 4 results in when the difference between CMS and OMS ($-OMS+CMS$) is more than 5%, but less than 20% (between 55% to 70%). The above seven states cover all possible differences between the broker's CMS and the OMS.

6.2 Action Space

The action space of the GenerateTariffs-EXP3 generates a new tariff contract in the tariff market. As discussed in Section 3, a broker needs to come up with rate values to design a new tariff contract. GenerateTariffs-EXP3's action space modifies the currently active tariff or suggests keeping the sane tariff active. Below is the action space,

- Action 0: $step = 0.0$ [$Maintain$]
- Action 1: $step = -0.02$ [$Lower1$]
- Action 2: $step = -0.04$ [$Lower2$]
- Action 3: $step = 0.02$ [$Higher1$]
- Action 4: $step = 0.04$ [$Higher2$]

As shown in the action space, GenerateTariffs-EXP3 can choose one of the five actions at any instance. The action selection problem is modeled as a MAB problem, which is solved using *EXP-3* algorithm in Section 6.4. At

any instance, GenerateTariffs-EXP3 can choose to maintain or modify the current tariff. If it chooses to modify the current tariff, it can either decrease the rate value of the currently active tariff or increase the rate value. The above action space provides two options for both scenarios; $Lower1$ or $Lower2$ to decrease the rate value and $Higher1$ or $Higher2$ to increase the rate value. After selecting an action, we decide the rate value of the new tariff by adding the *step* value of the selected action to the currently active tariff's average rate value. Thus generated new rate value is given to the TD sub-module that designs and publishes the new ToU tariff in the market. Note that, in PowerTAC sign convention, consumption tariffs are negatively valued as customers need to *pay* that amount; thus, actions such as *Lower1* and *Lower2* would make tariffs more negative (less attractive for customers), and actions such as *Higher1* and *Higher2* would make tariff less negative (more attractive for customers).

6.3 Reward

The reward function is defined in line with the state space, as shown below.

- reward = 1.00, if $|\Delta| \leq 5\%$
- reward = 0.50, if $|\Delta| \leq 20\%$
- reward = 0.25, if $|\Delta| \leq 35\%$
- reward = 0.00, otherwise

The above reward function awards the GenerateTariffs-EXP3 based on its ability to achieve market share close to the OMS. It gets the highest reward of 1 when the absolute difference between the broker's CMS and the OMS is less than 5%. Similarly, it gets a slightly worse reward when the difference is more than 5% (but less than 20%). The worst case happens when the market share achieved by GenerateTariffs-EXP3 is far away from the OMS (the difference is more than 35%); in that case, GenerateTariffs-EXP3 receives a zero reward.

6.4 EXP-3 Algorithm

The above contextual MAB-based tariff generation problem is solved using the *Exponential-weight algorithm for Exploration and Exploitation (EXP-3 algorithm)*. Generally, EXP-3 is used for non-contextual MAB problems but can also be extended for contextual MAB problems. For each state in the state space, It maintains a list of weights for each action in the action space. Using these weights, it stochastically decides which action to take next, and based on the reward received, it increases or decreases the relevant weights. Thus generated table resembles with Q-Table in Reinforcement Learning (RL). In RL Q-Table, the values of the state-action pairs denote how good it is to take that action in the given state in the long run, whereas, in ConMAB, the state-action pairs have the same interpretation albeit for an immediate future. Due to the similarity,

Algorithm 2 Contextual EXP-3(state s)

1: Initialize/Load table$[|S|][|A|]$
2: $prob(s,i,t) = (1-\gamma)\frac{table(s,i,t)}{\sum_{a=1}^{|A|} table(s,a,t)} + \frac{\gamma}{|A|}, \forall i \in \{1,2,...,|A|\}$
3: Sample next action act stochastically from $[prob(s,1,t), prob(s,2,t), ..., prob(s,|A|,t)]$
4: Observe reward $r(s,act,t)$ for taking action act in state s at t
5: Update the reward:
$\hat{r}(s,a,t) = r(s,a,t)/prob(s,a,t)$, if $a = act_t$
$\hat{r}(s,a,t) = 0$, otherwise
6: $table(s,i,t+1) = table(s,i,t) * e^{\gamma * \hat{r}(s,i,t)/|A|}, \forall i \in \{1,2,...,|A|\}$

we call the table generated by ConMAB as Q-Table. We introduce an egalitarianism factor $\gamma \in [0,1]$, tuning the desire to randomly pick an action. That is, if $\gamma = 1$, the weights do not affect the choices at any step. Algorithm 2 shows the modified EXP-3 algorithm for contextual MAB:

Algorithm 2 takes the current state s as the input. If the *table* is empty (at the start of the training), then initialize it with suitable values; otherwise, load the previously created *table* into memory. As described earlier, the dimensions of this table are $|S| * |A|$ (the size of state space S * the size of action space A). In the next step, we weigh the actions based on the corresponding values stored in the *table*. The probability of selecting an action i in state s at time t ($prob(s,i,t)$) is directly proportional to the corresponding state-action pair at time ($table(s,i,t)$). Here, an egalitarianism factor $\gamma \in [0,1]$ also plays a role in action selection; $\gamma = 0$ would calculate probabilities purely based on *table* values, while $\gamma = 1$ would assign the same probability to each of the actions. After calculating the probabilities for each action i in state s, in step 3, the algorithm stochastically picks one action based on the calculated probabilities. In step 4, the algorithm observes the reward $r(s,a,t)$ for taking action a in state s at time t. After that, in step 5, the algorithm updates the reward based on whether the action was selected or not; the new reward function $\hat{r}(s,a,t)$ is inversely proportional to the probability $prob(s,a,t)$. If the action was not selected, then the $\hat{r}(s,a,t)$ is set to zero, as expected. Finally, in step 6, the algorithm updates the *table*; only the state-action pair that got selected at time t gets updated, while other values in *table* remain unchanged. These updates are exponential in nature and proportional to the new reward $\hat{r}(s,a,t)$.

The EXP-3 algorithm deals with the explore-exploit dilemma by stochastically selecting an action based on the calculated probabilities in step 3. This step ensures picking the best-known action till now with higher probability while also occasionally selecting 'not so good' actions. After selecting any action and getting the corresponding reward in that state, it weighs the reward with respect to the probability. A reward for low-probability actions gets enhanced even further, allowing the algorithm to revisit those actions. Thus, the EXP-3 algorithm visits all the state-action pairs a sufficient number of times. In the next section, we

show how GenerateTariffs-EXP3 learns the policies to maintain the optimal market shares in each player configuration.

7 PowerTAC: Experiments and Results

In this section, we describe how the strategy described in Section 6 is deployed to the PowerTAC games. We further demonstrate how the learning process for EXP-3 is carried out in PowerTAC environment. We start by detailing the experimental setup, followed by the results and discussions.

7.1 Experimental Set-up

Q-Table Training: As the broker needs to adapt to various player configurations in PowerTAC, we deploy separate tariff MDP and EXP-3 algorithms in each configuration. In this experiment, we train three different models for three-player configurations, namely, 2-Player, 3-Player, and 5-Player. We chose these three configurations as the last PowerTAC tournament (PowerTAC22) had the same configurations. In each player configuration, we played 50 PowerTAC games, where each game simulates the smart grid operations for two months. At the start of the training, we initialize Q-Table with appropriate values and publish an initial tariff in the market. We keep the same tariff active for a day (24 hours) and then update the tariff at the start of the next day. While updating the tariff, we note the CMS and decide the reward to update the Q-Table as shown in Algorithm 2. This constitutes one epoch of training. After that, based on the CMS, we calculate the current state and choose an action following the EXP-3 algorithm, and publish a new ToU tariff in the market by using the TD sub-module. We continue this process and record Q-Table after every checkpoint (typically after every 100 epoch) as well as at the end of the game. While starting a new game, we read and update the previously stored Q-Table while training. We train GenerateTariffs-EXP3 for around 3000 epochs for each configuration and store the final Q-Tables.

Performance Testing: We conduct performance testing to verify whether GenerateTariffs-EXP3 is able to maintain the desired market share during the games after getting trained. As mentioned previously, we store intermediate Q-Tables after every checkpoint and test the effectiveness of GenerateTariffs-EXP3 at various stages of the training. For this, we take Q-Tables from seven different checkpoints, play 10 games with each Q-Table, and record the average market shares during the games. At the end of 10 games, we record the average and standard deviation of market shares after 10 games; we do this for all seven Q-Tables. In this paper, we present the performance testing for the 3-Player configuration. The following section showcases the result of this experiment.

7.2 Results and Discussion:

In this section, we present the results of the Q-Table training for the above-mentioned 2-Player, 3-Player, and 5-Player configurations. Furthermore, we also show the efficacy of the GenerateTariffs-EXP3 in maintaining the suggested market share during the games.

Q-Table Training: Figure 4, 5, and 6 are the final Q-Tables after training GenerateTariffs-EXP3 for 50 games (around 3000 epochs) for each player configuration. In Q-Tables, the higher the value (green-shaded region) for any state-action pair, the higher probability of that action getting selected in the given state.

First, focus on the 2-Player Q-Table in Figure 4. GenerateTariffs-EXP3 learns to maintain the currently active tariff if the current state is State 0, which is the best thing to do as the market share is already in the desired range. In State 1 as well, it chooses to continue with the current tariff. When the CMS is lower than OMS in State 2 and 3, it learns to select *Higher2* action to make tariff cheaper and very much attractive for customers to increase the CMS and go closer to OMS (*Higher2* would add a high positive step value in the negatively valued tariff, which makes tariff cheaper from customers' perspective). The same explanation is valid for the other side of the state space when the CMS is higher than OMS. It chooses to *Maintain* in State 4 and go for *Lower*2 for remaining states State 5 and 6 in order to make tariff less attractive for customers and decrease the CMS and reach closer to OMS.

Action / State	Maintain	Lower1	Lower2	Higher1	Higher2
0	33.64	16.99	10.35	28.90	14.85
1	361.41	30.35	11.11	18.30	167.86
2	18.07	4.04	3.59	22.95	32.24
3	2.02	1.74	1.32	3.66	7.94
4	40.02	27.96	19.82	20.95	10.69
5	4.33	7.35	15.81	4.31	2.36
6	1.17	2.46	4.47	1.13	1.32

Fig. 4: Q-Table for 2-Player Configuration [After 50 Games]

The other two player configurations too converge to similar Q-Tables; however, the values are very different from each other. For example, in State 2, 3-Player Q-Table would select the *Higher2* with high probability, while 5-Player Q-Table would pick *Higher1* or *Higher2* with almost equal probability. In summary, GenerateTariffs-EXP3 learns to decide the suitable action in each state for all three player configurations, which empirically looks like the correct

Action / State	Maintain	Lower1	Lower2	Higher1	Higher2
0	31.88	31.01	12.77	31.08	16.02
1	56.31	31.82	9.31	46.91	60.29
2	11.92	5.25	3.81	10.75	35.35
3	4.11	1.20	1.30	4.82	24.37
4	26.51	46.61	49.11	23.80	11.94
5	2.91	6.58	4.56	2.74	1.43
6	1.45	2.68	8.40	1.96	1.32

Fig. 5: Q-Table for 3-Player Configuration [After 50 Games]

action to pick given the state. To prove that the above Q-Tables actually learn the best actions in each state to achieve the goal of maintaining the desired market, we carried out a performance testing for the 3-Player configuration and report the results below.

Action / State	Maintain	Lower1	Lower2	Higher1	Higher2
0	2.30	1.54	1.41	2.00	1.36
1	4.83	2.00	1.47	4.23	2.19
2	2.90	1.45	1.07	6.24	6.40
3	1.72	1.00	1.04	5.85	33.89
4	2.14	5.61	1.78	1.62	1.51
5	1.66	3.60	1.81	1.22	1.10
6	1.36	12.27	10.62	1.44	1.69

Fig. 6: Q-Table for 5-Player Configuration [After 50 Games]

Performance Testing: Figure 7 shows the market share maintained by Q-Tables stored at various checkpoints (at the 0th epoch, 500th epoch etc.) for a 3-Player configuration. The light blue color strip in the graph shows the desired market share range for the 3-Player configuration. As seen from the graph, for the 0th epoch Q-Table which has an equal probability for each section getting selected, the market share maintained by GenerateTariffs-EXP3 is very far from the desired range. After 500 and 1000 epochs, too, it is not able to maintain market share in the desired range. However, after getting trained for 1500 epochs, it reaches closer to the desired range. After that, for the higher number of epochs, it maintains the market share within the desired range. The variance (shown as the bars around the dot) is also low after 2500 epochs of training. A similar result is achieved for the 2-Player and 5-Player configurations as well. This shows the

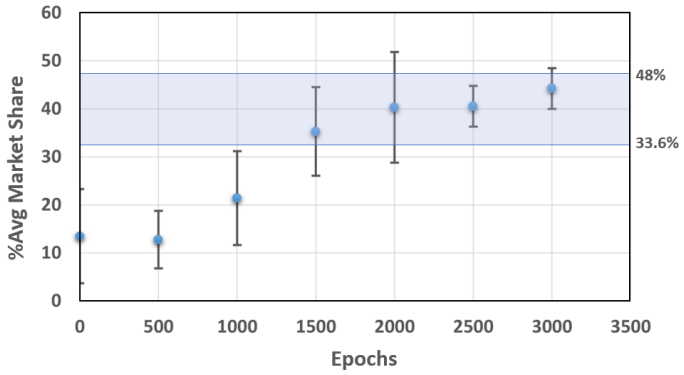

Fig. 7: Market Share Maintained by GenerateTariffs-EXP3 w.r.t Number of Epochs of Training for 3-Player Configuration

efficacy of GenerateTariffs-EXP3 that learns to update tariffs online and maintains the desired market share during the games.

8 Conclusion

Using the Contextual Multi-armed Bandit-based technique, we described the design of an adaptive tariff contract generation strategy, GenerateTariffs-EXP3, to sell electricity in the retail market. In particular, we demonstrated how tariff contracts could be adapted in real-time based on the market situation using the EXP-3 algorithm that efficiently managed the explore-exploit dilemma and visited all the states a sufficient number of times. In our strategy, we first determined the optimal market share and trained GenerateTariffs-EXP3 to achieve and maintain that market share during the game. We showcased that after training for an adequate number of games, GenerateTariffs-EXP3 learns the optimal action for a given state and learns to maintain the appropriate market share during the PowerTAC games.

References

1. Chandlekar, S., Pedasingu, B.S., Subramanian, E., Bhat, S., Paruchuri, P., Gujar, S.: Vidyutvanika21: An autonomous intelligent broker for smart-grids. In: Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22. pp. 158–164. International Joint Conferences on Artificial Intelligence Organization (7 2022). https://doi.org/10.24963/ijcai.2022/23, https://doi.org/10.24963/ijcai.2022/23
2. Ghosh, S., Subramanian, E., Bhat, S.P., Gujar, S., Paruchuri, P.: Vidyutvanika: A reinforcement learning based broker agent for a power trading competition. Proceedings of the AAAI Conference on Artificial Intelligence **33**, 914–921 (2019). https://doi.org/10.1609/aaai.v33i01.3301914

3. Jain, S., Narayanaswamy, B., Narahari, Y.: A multiarmed bandit incentive mechanism for crowdsourcing demand response in smart grids. In: AAAI Conference on Artificial Intelligence. Canada (2014)
4. Ketter, W., Collins, J., de Weerdt, M.: The 2020 power trading agent competition (march 30, 2020). ERIM Report Series Reference No. 2020-002, http://dx.doi.org/10.2139/ssrn.3564107 (2020)
5. Li, Y., Hu, Q., Li, N.: Learning and selecting the right customers for reliability: A multi-armed bandit approach. In: 2018 IEEE Conference on Decision and Control (CDC). pp. 4869–4874 (2018). https://doi.org/10.1109/CDC.2018.8619481
6. Ma, H., Parkes, D.C., Robu, V.: Generalizing demand response through reward bidding. In: Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems. pp. 60–68. AAMAS'17, Brazil (2017), http://dl.acm.org/citation.cfm?id=3091125.3091140
7. Ma, H., Robu, V., Li, N.L., Parkes, D.C.: Incentivizing reliability in demand-side response. In: the proceedings of The 25th International Joint Conference on Artificial Intelligence (IJCAI'16). pp. 352–358 (2016), http://www.ijcai.org/Abstract/16/057
8. McKelveya, R.D., McLennan, A.M., Turocy, T.L.: Gambit: Software Tools for Game Theory, Version 16.0.1. http://www.gambit-project.org (2014), [Online; accessed 27-December-2021]
9. Methenitis, G., Kaisers, M., La Poutré, H.: Forecast-based mechanisms for demand response. In: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems. pp. 1600–1608 (2019)
10. Orfanoudakis, S., Kontos, S., Akasiadis, C., Chalkiadakis, G.: Aiming for half gets you to the top: Winning powertac 2020. In: Multi-Agent Systems, 18th European Conference, EUMAS 2021. pp. 144–159 (07 2021)
11. Reddy, P.P., Veloso, M.M.: Strategy learning for autonomous agents in smart grid markets. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Two, IJCAI'11. pp. 1446—-1451. AAAI Press (2011)
12. Serrano, J., , González, A.Y.R., de Cote, M.: Fixed-price tariff generation using reinforcement learning. Fujita K. et al. (eds) Modern Approaches to Agent-based Complex Automated Negotiation. Studies in Computational Intelligence, Springer, Cham **674** (2017)
13. Shweta, J., Sujit, G.: A multiarmed bandit based incentive mechanism for a subset selection of customers for demand response in smart grids. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 2046–2053 (2020)
14. Techopedia.com: Smart Grid. https://www.techopedia.com/definition/692/smart-grid (2021), [Online; accessed 19-January-2023]
15. Urieli, D., Stone, P.: Autonomous electricity trading using time-of-use tariffs in a competitive market. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16). Association for the Advancement of Artificial Intelligence (2016)
16. Özdemir, S., Unland, R.: Agentude17: A genetic algorithm to optimize the parameters of an electricity tariff in a smart grid environment. Advances in Practical Applications of Agents, Multi-Agent Systems, and Complexity: The PAAMS Collection pp. 224–236 (06 2018). https://doi.org/10.1007/978-3-319-94580-4$_1$8