

# Encrypted Value Iteration and Temporal Difference Learning over Leveled Homomorphic Encryption

Jihoon Suh

Takashi Tanaka

**Abstract**—We consider an architecture of confidential cloud-based control synthesis based on Homomorphic Encryption (HE). Our study is motivated by the recent surge of data-driven control such as deep reinforcement learning, whose heavy computational requirements often necessitate an outsourcing to the third party server. To achieve more flexibility than Partially Homomorphic Encryption (PHE) and less computational overhead than Fully Homomorphic Encryption (FHE), we consider a Reinforcement Learning (RL) architecture over Leveled Homomorphic Encryption (LHE). We first show that the impact of the encryption noise under the Cheon-Kim-Kim-Song (CKKS) encryption scheme on the convergence of the model-based tabular Value Iteration (VI) can be analytically bounded. We also consider secure implementations of TD(0), SARSA(0) and Z-learning algorithms over the CKKS scheme, where we numerically demonstrate that the effects of the encryption noise on these algorithms are also minimal.

## I. INTRODUCTION

The growing demand of applications such as smart grids [1] and smart cities [2] assures the important role of data usage and connectivity via advanced data-driven algorithms. However, the utility of advanced data-driven algorithms may be limited in many real-world control systems since components within the network are often resource-constrained. The cloud-based control can be an appealing solution in such scenarios, although a naive outsourcing comes with a steep cost of privacy. Recent literature in control has shown that the use of HE could mitigate the issue of privacy to a certain extent. However, there are many remaining challenges in encrypted control technologies. For instance, in the current encrypted control literature, the potential of FHE is not fully utilized, even though FHE would be necessary to encrypt advanced control algorithms, such as deep reinforcement learning [3]. RL framework has recently accomplished impressive feats with successful applications from AlphaGo Zero to traffic signal control, robot controls and many others, [4], [5], [6]. Its success is also bolstered by the availability of large-scale data and connectivity. Thus, we wish to study a cloud-based control synthesis architecture which can integrate two promising technologies, namely HE and RL. In particular, we examine the effects of using HE on RL both theoretically and numerically.

The first applications of HE to control systems utilized PHE since its relatively low computational overhead was suitable for real-time implementations. On the other hand,

FHE or LHE can support more general classes of computations in the ciphertext domain, although its computational overhead makes it less suitable for real-time systems. In this regard, we first make an observation that the encrypted control can be better suited to the control synthesis problems rather than control implementations. For instance, explicit model predictive control (MPC) or RL problems require heavy computations or a large set of data to synthesize the control policy but implementing such policy may be kept local as they are often relatively light computations. In addition, real-time requirement is less stringent in control law synthesis problems than control law implementation counterpart.

### A. Related Work

Various HE schemes were applied to the linear controller in [7], [8], [9]. Importance of quantization and using non-deterministic encryption scheme was identified in [7]. Necessary conditions on encryption parameters for closed-loop stability were shown by [8]. The feasibility of FHE in encrypted control was first shown in [9] by managing multiple controllers. Evaluation of affine control law in explicit MPC was shown by [10]. In [11] encrypted implicit MPC control was shown to be feasible but the cost of privacy as increased complexity was identified. Real-time proximal gradient method was used in [12] to treat encrypted implicit MPC control and it showed undesired computation loads of encrypted control system. Privacy and performance degradation was further highlighted in [13] through experiments on motion control systems. Dynamic controller was encrypted in [14] using FHE exploiting the stability of the system while not relying on bootstrapping. Despite significant progress, encrypted control has been limited to simple computations where the motivation for cloud computing is questionable.

### B. Contribution

To study the feasibility of RL over HE, this paper makes the following contributions. First, as a first step towards more advanced problems of private RL, we formally study the convergence of encrypted tabular VI. We show that the impact of the encryption-induced noise can be made negligible if the  $Q$ -factor is decrypted in each iteration. Second, we present implementation results of temporal difference (TD) learning (namely, TD(0), SARSA(0), and Z-learning [15]) over the CKKS encryption scheme and compare their performances with un-encrypted cases. Although the formal performance analysis for this class of algorithms are difficult

This work is supported by NSF Award 1944318.

Both authors are with the Department of Aerospace Engineering and Engineering Mechanics, University of Texas at Austin, TX 78712, USA. {jihoonsuh, ttanaka}@utexas.edu

with encryption noise, we show numerically that these RL schemes can be implemented accurately over LHE.

### C. Preview

In section II, we summarize the relevant essentials of RL and HE. In section III, we set up the encrypted control synthesis problem and specialize it to the encrypted model-based and model-free RL algorithms. We analyze how the encryption-induced noise influences the convergence of standard VI. In section IV-A, we perform the simulation studies to demonstrate the encrypted model-free RL over HE. Finally, in section V, we summarize our contributions and discuss the future research directions.

## II. PRELIMINARIES

### A. Reinforcement Learning

RL can be formalized through the finite state Markov decision process (MDP). We define a finite MDP as a tuple  $(\mathcal{S}, \mathcal{A}, P, R)$ , where  $\mathcal{S} = (1, 2, \dots, n)$  is a finite state space,  $\mathcal{A}$  is a finite action space,  $P = P(s_{t+1}|s_t, a_t)$  is a Markov state transition probability and  $R = R(s_t, a_t, s_{t+1})$  is a reward function for the state-transition. A policy can be formalized via a sequence of stochastic kernels  $\pi = (\pi_0, \pi_1, \dots)$ , where  $\pi_t(a_t|s_{0:t+1}, a_{0:t})$  is a mapping for each state  $s_t \in \mathcal{S}$  to the probability of selecting the action  $a_t \in \mathcal{A}$  given the history  $(s_{0:t-1}, a_{0:t-1})$ .

We consider a discounted problem with a discount factor  $\gamma \in [0, 1)$  throughout this paper. Under a stationary policy  $\pi$ , the Bellman's optimality equation can be defined through a recursive operator  $T$  applied on the initial value vector  $V_0^\pi$  of the form  $V^\pi = (V(1), \dots, V(n))^\pi$ :

$$(TV^\pi)(s_t) = \max_{a_t} \sum_{s_{t+1}} P[R + \gamma V^\pi(s_{t+1})]. \quad (1)$$

Throughout this paper, we adopt the conventional definition for the maximum norm  $\|\cdot\|_\infty$ . The following results are standard [16].

*Lemma 1:* (a) For any vectors  $V$  and  $\bar{V}$ , we have

$$\|TV - T\bar{V}\|_\infty \leq \gamma \|V - \bar{V}\|_\infty.$$

(b) The optimal value vector  $V^*$  is the unique solution to the equation  $V^* = TV^*$ ,

(c) We have

$$\lim_{k \rightarrow \infty} T^k V = V^*$$

for any vector  $V$ .

If some policy  $\pi^*$  attains  $V^*$ , we call  $\pi^*$  the optimal policy and the goal is to attain the optimal policy  $\pi^*$  for the given MDP environment.

Often the state values are written conveniently as a function of state-action pairs called  $Q$ -values:

$$Q^\pi(s_t, a_t) = \sum_{s_{t+1}} P[R + \gamma V^\pi(s_{t+1})], \quad (2)$$

and for all state  $s_t$ ,  $V_{k+1}^\pi(s_t) = \max_{a_t \in \mathcal{A}} Q^\pi(s_t, a_t)$ .

RL is a class of algorithms that solves MDPs when the model of the environment (e.g.,  $P$  and/or  $R$ ) is unknown.

RL can be further classified into two groups: model-based RL and model-free RL [17]. A simple model-based RL first estimates the transition probability and reward function empirically (to build an artificial model) then utilizes the VI to solve the MDP. Model-free RL on the other hand directly computes the value function by averaging over episodes (Monte Carlo methods) or by estimating the values iteratively like TD learning.

TD(0) is one of the simplest TD learning algorithm where the tuple  $(s, R, s')$  is sampled from the one-step ahead observation [18]. The on-policy iterative update rule for estimating the value is called TD(0) and is written as follows:

$$\hat{V}_{t+1}^\pi(s) = \hat{V}_t^\pi(s) + \alpha_t(s) \delta_t. \quad (3)$$

$\delta_t$  denotes the TD error and is computed with the sample as

$$\delta_t = R(s, a) + \gamma \hat{V}_t^\pi(s') - \hat{V}_t^\pi(s),$$

starting with an initial guess  $\hat{V}_0^\pi$ . We use  $\hat{V}$  to indicate that the values are estimated. With some standard assumptions on the step size  $\alpha_t(s)$  and exploring policies, the convergence of TD(0) is standard in the literature, see [19].

### B. Homomorphic Encryption

HE is a structure-preserving mapping between the plain-text domain  $\mathcal{P}$  and the cipher-text domain  $\mathcal{C}$ . Thus, encrypted data in  $\mathcal{C}$  with a function  $f(\mathcal{C})$  can be outsourced to the cloud for confidential computations.

PHE supports only the addition or the multiplication. On the other hand, LHE can support both the addition and the multiplication but the noise growth of ciphertext multiplication is significant without the bootstrapping [20]. The bootstrapping operation can promote a LHE scheme to FHE enabling unlimited number of multiplications but it requires a heavy computation resource on its own.

In order to use both addition and multiplication necessary in evaluating RL algorithms, we use an LHE. In particular, we employ CKKS encryption scheme proposed in [21] as it is well suited for engineering applications and also supports parallel computations. Also, there exists a widely available library optimized for implementations. We will only list and describe here several key operations and properties with more detailed information found in Appendix of the full version [22]. We are particularly concerned with the fact that the CKKS encryption is noisy (due to error injection for security) but if properly implemented, the noise is manageable.

The security of CKKS scheme assumes the difficulty of the Ring Learning With Errors (RLWE) problem. Security parameters are a power-of-two integer  $\mathcal{N}$ , a ciphertext modulus  $\mathcal{Q}$  and the variance  $\sigma$  used for drawing error from a distribution. The operation *KeyGen* uses security parameters  $(\mathcal{N}, \mathcal{Q}, \sigma)$  to create a  $\lambda$ -bit public key  $pk$ , a secret key  $sk$ , and an evaluation key  $evk$ .

## III. ENCRYPTED LEARNING OVER THE CLOUD

### A. Cloud-based reinforcement learning

Our control loop consists of the plant (environment) and the controller (client) and the cloud server. We are concerned

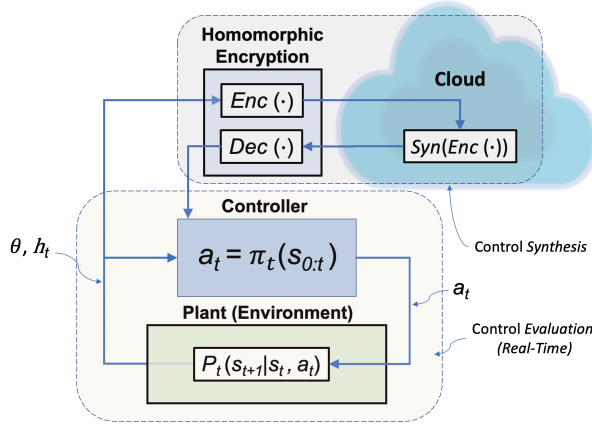


Fig. 1. Encrypted RL Control synthesis with cloud in the loop.

with a possible data breach at the cloud. In order to prevent the privacy compromise of our data such as training data set and other parameters at play, we add the homomorphic encryption module as seen in Fig. 1. In particular, we ignore the malleability risks [23] inherent to the considered HE scheme, which is beyond the scope of this paper. In our context,  $Syn(Enc(\cdot))$  can be thought of as a ciphertext domain implementation of RL algorithms. However,  $Syn(Enc(\cdot))$  is a general control synthesis that can be evaluated remotely. Ideal uses for  $Syn(Enc(\cdot))$  would include advanced data-driven control synthesis procedures such as MPC, or Deep RL. We emphasize that the encryption adds communication delays for control policy synthesis but does not add an extra computation cost to control implementation. For RL, the controller's task is to implement the most up-to-date state-action map  $\pi$  that is recently synthesized, which can be done locally.

We assume the tabular representation of the state and action pairs. The client explores its plant and samples the information set  $h_t = (s_{t+1}, a_t, r_t)$ . Other necessary parameters such as learning rate  $\alpha_t(s)$  and  $\gamma$  are grouped as  $\theta$  denoting the hyperparameters. This set of data is then encrypted by the CKKS encryption module to generate ciphertexts  $\mathbf{c}_v$ ,  $\mathbf{c}_{v'}$ ,  $\mathbf{c}_\alpha$ ,  $\mathbf{c}_\gamma$ , and  $\mathbf{c}_r$ , which are encrypted data for  $\hat{V}_t^\pi(s)$ ,  $\hat{V}_t^\pi(s')$ ,  $\alpha_t(s)$ ,  $\gamma$ , and  $R(s, a)$ , and will be used to implement (3). Note that the subscripts refer to the value when these ciphertexts are decrypted. The cloud is instructed on how to evaluate the requested algorithms in ciphertexts. One may concern about the cloud knowing the form of the algorithm, but this can be resolved by the circuit privacy. That is, we can hide the actual computation steps by adding zeros or multiplying ones in ciphertexts on the original algorithm. We also assume that the cloud is semi-honest so that the cloud faithfully performs the algorithm as instructed. The output of the cloud is a newly synthesized policy  $\pi_t(s_{0:t})$ , which, after a decryption, can be accessed on real-time by the client to produce the control action  $a_t(s_t)$ .

As a first step towards investigating more sophisticated RL algorithms, we specialize the proposed framework to solving RL problems in finite MDP with more elementary

methods. In particular, we consider the basic model-based RL and model-free RL using tabular algorithms. For the model-based, we theoretically analyze the convergence of VI under the presence of encryption-induced noise. For the model-free, we implement the encrypted TD algorithms to estimate the values and investigate how the encryption noise affects the output.

### B. Encrypted Model-based RL

The client is assumed to explore the plant, sample the information sets  $h_t$  and build the model first. A simple model of the state-transition probability can be in the form

$$P(s_{t+1}|s_t, a_t) = \frac{N(s, a, s')}{N(s, a)}, \quad (4)$$

where  $N(s, a, s')$  denotes the number of visits to the triplet  $(s, a, s')$  and  $N(s, a)$  to the pair  $(s, a)$  with  $s, s' \in \mathcal{S}$ ,  $a \in \mathcal{A}$ . Similarly, the reward function  $R(s_t, a_t, s_{t+1})$  can be quantified by the average rewards accumulated per the state-action pair.

Then, the client can encrypt the initial values of the state along with the model  $P$  and  $R$  and the discount rate  $\gamma$  and request the cloud to perform the computation of (2) over the ciphertext domain. Since comparison over HE is non-trivial, the max operation is difficult to be implemented homomorphically. Thus, the cloud computes the  $Q$  values and the controller will receive the decrypted  $Q$  values and completes the VI process for iteration index  $k$ :

$$\tilde{V}_{k+1}^\pi(s_t) = \max_{a_t \in \mathcal{A}} \tilde{Q}^\pi(s_t, a_t). \quad (5)$$

However, note that the  $Q$  values computed by the cloud will be noisy due to the encryption, hence we use  $\tilde{Q}$  and  $\tilde{V}$  to denote the noisy value. Let  $w(s_t, a_t)$  be the encryption-induced noise produced by computations over HE. Then, the noisy  $Q$  values can be written as

$$\tilde{Q}^\pi(s_t, a_t) = Q^\pi(s_t, a_t) + w(s_t, a_t), \quad (6)$$

where the noise term is bounded such that  $|w(s_t, a_t)| \leq \epsilon$   $\forall s_t, a_t$  for some  $\epsilon > 0$ . Appendix A shows how  $\epsilon$  depends on the encryption parameter and operations used to evaluate the given algorithm.

We now analyze the discrepancy between two sequences of vectors  $V_k^\pi$  and  $\tilde{V}_k^\pi$ . We separate the analysis into the synchronous and the asynchronous cases.

1) *Synchronous VI*: The synchronous VI means that the VI is applied to all state  $s$  simultaneously. First note that  $V_k^\pi$  is computed by the noiseless VI

$$\begin{aligned} V_{k+1}^\pi(s_t) &= \max_{a_t \in \mathcal{A}} \sum_{s_{t+1}} P[R + \gamma V_t^\pi(s_{t+1})] \\ &= (TV_k^\pi)(s_t) \quad \forall s_t \in \mathcal{S}, \end{aligned} \quad (7)$$

and  $\tilde{V}_k^\pi$  is computed by the noisy VI

$$\tilde{V}_{k+1}^\pi(s_t) = \max_{a_t \in \mathcal{A}} \sum_{s_{t+1}} P[R + \gamma V_k^\pi(s_{t+1}) + w(s_t, a_t)] \quad (8)$$

We will utilize the following simple lemma, whose proof is straightforward and hence omitted.

**Lemma 2:** For any arbitrary vectors  $x = (x(1), \dots, x(n))$  and  $w = (w(1), \dots, w(n))$  such that  $\|w\|_\infty \leq \epsilon$ ,

$$\max_i (x(i) + w(i)) = \max_i (x(i)) + \tilde{w} \quad (9)$$

for some constant  $\tilde{w}$  satisfying  $|\tilde{w}| \leq \epsilon$ .

By applying Lemma 2 on the noisy VI (8), we obtain

$$\begin{aligned} \tilde{V}_{k+1}^\pi(s_t) &= \max_{a_t \in \mathcal{A}} \sum_{s_{t+1}} P[R + \gamma \tilde{V}_k^\pi(s_{t+1}) + w_k(s_t, a_t)] \\ &= \max_{a_t \in \mathcal{A}} \sum_{s_{t+1}} P[R + \gamma \tilde{V}_k^\pi(s_{t+1})] + \tilde{w}_k(s_t) \\ &= (T\tilde{V}_k^\pi)(s_t) + \tilde{w}_k(s_t), \end{aligned} \quad (10)$$

where the vector  $\tilde{w}_k$  satisfies  $\|\tilde{w}_k\|_\infty \leq \epsilon$ . In the vector form, (10) can be written as

$$\begin{aligned} \tilde{V}_{k+1}^\pi &= T\tilde{V}_k^\pi + \tilde{w}_k, \\ &= \tilde{T}\tilde{V}_k^\pi. \end{aligned} \quad (11)$$

We can now quantify the worst-case performance degradation due to the encryption-induced noise. The result is summarized in the next Theorem.

**Theorem 1:** (Approximate VI, [16]) Let  $V^*$  be the optimal value function characterized by Lemma (1) (a). Suppose  $\tilde{V}_k^\pi$  is the sequence of vectors computed by the noisy VI (8). For an arbitrary initial condition  $\tilde{V}_0$ , we have

$$\limsup_{k \rightarrow \infty} \|V^* - \tilde{V}_k^\pi\|_\infty \leq \frac{\epsilon}{1 - \gamma}.$$

*Proof:* Let  $\tilde{V}_k^\pi$  be the sequence of vectors computed by the noiseless VI (7) with arbitrary initial conditions  $V_0^\pi$  and  $\tilde{V}_0^\pi$ . Then,

$$\|V_{k+1}^\pi - \tilde{V}_{k+1}^\pi\|_\infty = \|TV_k^\pi - T\tilde{V}_k^\pi - \tilde{w}_k\|_\infty \quad (12a)$$

$$\leq \|TV_k^\pi - T\tilde{V}_k^\pi\|_\infty + \|\tilde{w}_k\|_\infty \quad (12b)$$

$$\leq \|TV_k^\pi - T\tilde{V}_k^\pi\|_\infty + \epsilon \quad (12c)$$

$$\leq \gamma \|V_k^\pi - \tilde{V}_k^\pi\|_\infty + \epsilon, \quad (12d)$$

where the first inequality (12b) follows from (11), and (12c) follows from the triangular inequality, and the last inequality is due to Lemma (1). Now define a sequence  $e_k$  of positive numbers by

$$e_{k+1} = \gamma e_k + \epsilon \quad (13)$$

with  $e_0 = \|V_0^\pi - \tilde{V}_0^\pi\|_\infty$ . By geometric series,

$$\limsup_{k \rightarrow \infty} e_k = \frac{\epsilon}{1 - \gamma}.$$

Comparing (12) and (13), we have by induction

$$\|V_k^\pi - \tilde{V}_k^\pi\|_\infty \leq e_k, \forall k = 0, 1, 2, \dots$$

Therefore,

$$\limsup_{k \rightarrow \infty} \|V_k^\pi - \tilde{V}_k^\pi\|_\infty \leq \frac{\epsilon}{1 - \gamma}. \quad (14)$$

Since (14) holds for any  $V_0^\pi$ , we can pick  $V_0^\pi = V^*$  for which we have  $V_k^\pi = T^k V^* = V^*$  for  $k = 0, 1, 2, \dots$  by lemma (1), we obtain the desired result. ■

2) *Asynchronous VI:* It is often necessary or beneficial to run the VI algorithm asynchronously, or state-by-state, for simulations. We can define the asynchronous noiseless and noisy VI with the new mapping  $F$  and  $\tilde{F}$ , respectively.

$$FV_k^\pi(s_t) = \begin{cases} (TV_k^\pi)(s) & \text{if } s = s_t, \\ V_k^\pi(s_t), & \text{otherwise,} \end{cases} \quad (15)$$

$$\tilde{F}\tilde{V}_k^\pi(s_t) = \begin{cases} (\tilde{T}\tilde{V}_k^\pi)(s) & \text{if } s = s_t, \\ \tilde{V}_k^\pi(s_t), & \text{otherwise.} \end{cases} \quad (16)$$

In each case, we make the following assumption.

**Assumption 1:** (a) Each state is visited for updates infinitely often.

(b) There exists a finite constant  $M$ , which is greater than or equal to the number of updates to sweep through each state at least once.

Assumption 1 is essential for the following theorem as it ensures that the mapping  $F$  and  $\tilde{F}$  are contraction operators as long as all the states are visited at least once and that the time it takes for visiting all the states at least once is finite. A common approach to ensure this assumption would be to incorporate exploring actions as seen in  $\epsilon$ -greedy policy.

Now, define the iteration sub-sequence  $\{k_n\}_{n=0,1,2,\dots}$  such that  $k_0 = 0$  and each state is visited at least once between  $k_{n+1}$  and  $k_n$ . For instance, consider a finite MDP with 4 states denoted  $s_1, s_2, s_3, s_4$ . If the state trajectory is  $(s_1, s_2, s_4, s_3, s_1, s_1, s_2, s_4, s_3, s_1, s_1, \dots)$  for  $t = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, \dots)$ , then the sub-sequence  $k_n$  formed is  $k_0 = 0, k_1 = 4, k_2 = 9, \dots$

**Theorem 2:** Let  $V^*$  be the optimal value function as defined previously. Suppose  $\tilde{V}_k^\pi$  is the sequence of vectors computed by the asynchronous noisy VI (16) under Assumption 1. For an arbitrary initial condition  $\tilde{V}_0$ , we have

$$\limsup_{n \rightarrow \infty} \|V^* - \tilde{V}_{k_n}^\pi\|_\infty \leq \frac{M\epsilon}{1 - \gamma}.$$

*Proof:* By definition of mapping  $F$  and  $\tilde{F}$ , we can write:

$$\begin{aligned} \|V_{k_{n+1}}^\pi - \tilde{V}_{k_{n+1}}^\pi\|_\infty &= \|F^{k_{n+1}-k_n} V_{k_n}^\pi - \tilde{F}^{k_{n+1}-k_n} \tilde{V}_{k_n}^\pi\|_\infty \\ &\leq \|F^{k_{n+1}-k_n} V_{k_n}^\pi - \tilde{F}^{k_{n+1}-k_n} \tilde{V}_{k_n}^\pi\|_\infty \\ &\quad + (k_{n+1} - k_n)\epsilon \end{aligned} \quad (17)$$

Similar to the proof of Theorem 1, the inequality is due to the bound of the error vector and the triangle inequality. Now, we note that the asynchronous mapping is a contraction mapping with respect to a sequence index  $k_n$ . Thus, we have

$$\|V_{k_{n+1}}^\pi - \tilde{V}_{k_{n+1}}^\pi\|_\infty \leq \gamma \|V_{k_n}^\pi - \tilde{V}_{k_n}^\pi\|_\infty + (k_{n+1} - k_n)\epsilon.$$

By forming a sequence with  $n$  as it is done in the previous proof, and by the Assumption 1-(b), it yields

$$\limsup_{n \rightarrow \infty} \|V_{k_n}^\pi - \tilde{V}_{k_n}^\pi\|_\infty \leq \frac{M\epsilon}{1 - \gamma}. \quad (18)$$

By expanding the left hand side and applying the reverse triangle inequality

$$\begin{aligned} \|V_{k_n}^\pi - \tilde{V}_{k_n}^\pi\|_\infty &= \|V_{k_n}^\pi - V^* - (\tilde{V}_{k_n}^\pi - V^*)\|_\infty \\ &\geq \|\tilde{V}_{k_n}^\pi - V^*\|_\infty - \|V_{k_n}^\pi - V^*\|_\infty, \end{aligned} \quad (19)$$

for which we know the second term approaches zero in the limit. Since the left hand side is bounded in the limit with constants, we achieve the proposed result. ■

Theorem 2 assures that the encrypted VI outsourced to the cloud also guarantees a comparable performance asymptotically.

### C. Encrypted Model-free RL

Consider again the TD(0) update rule, (3). The cloud receives the set of ciphertexts  $\{\mathbf{c}_v, \mathbf{c}_{v'}, \mathbf{c}_\alpha, \mathbf{c}_\gamma, \mathbf{c}_r\}$ . Then, the update rule in the ciphertext domain becomes:

$$\mathbf{c}_v(t+1) = \mathbf{c}_v(t) + \mathbf{c}_\alpha \cdot \mathbf{c}_r + \mathbf{c}_\alpha \cdot \mathbf{c}_\gamma \cdot \mathbf{c}_{v'}(t) - \mathbf{c}_\alpha \cdot \mathbf{c}_v(t). \quad (20)$$

Upon decryption of  $\mathbf{c}_v(t+1)$ , we need to remember that the computed value is corrupted with the noise. A similar error analysis for the TD algorithms can be performed (perhaps using stochastic approximation theory). However, the formal analysis in this domain presents some difficulties. Whereas a conventional theory on stochastic approximation with an exogenous noise seen in [16] requires the noise to approach zero in the limit and bounded, the encryption-induced noise satisfies only the bounded condition. We thus only provide some implementation results at this time to gain some insight. We hope to rigorously prove this case as in the future work.

## IV. SIMULATION

We present implementation results of various model-free (TD(0), SARSA(0), and Z-learning) RL algorithms over the CKKS implementation scheme. The environment used is the grid world with the state size  $|\mathcal{S}| = 36$  for all three and the action size  $|\mathcal{A}| = 9$  for the first two. The available actions are up, up-right, right, down-right, down, down-left, left, up-left, and stay. The reward is fixed as randomly set real numbers to simulate unknown environment. The client starts with a policy  $\pi$  at the box coordinate (1,1), top-left and the grid world has three trap states and one goal state, marked by letters **T** and **G**, which terminate the current episode. In each episode, we set the maximum number of steps at which the current episode terminates as well. The learning parameter set  $\theta$  consists of the discount factor, learning rate, and exploration percentage. As soon as the new data set  $h_t$  containing the reward (or cost) and values of states  $s$  and  $s'$  become available, the client uploads the encrypted data to the cloud.

Choosing encryption parameters is not straightforward but there exists a standardization effort, [24]. Also, an open-source library SEAL [25] provides a practical tutorial and accessible tools along with encryption parameter generator. We use the default 128-bit security ( $\lambda = 128$ ) encryption parameters  $(\mathcal{N}, \mathcal{Q}, \sigma)$  generated by SEAL with the user input of  $\mathcal{N}$ . These are listed in Table I. The size of  $\mathcal{N}$  need not be this large as there is no parallel operation exploited for the particular example application considered in this paper. However, future applications such as multi agents RL or deep RL can find such capability useful as they contain many batch operations.

TABLE I  
ENCRYPTION PARAMETERS

| Param.        | TD(0)                   | SARSA(0)                | Z-learning              |
|---------------|-------------------------|-------------------------|-------------------------|
| $\mathcal{N}$ | 8192                    | 8192                    | 16384                   |
| $\mathcal{Q}$ | 219                     | 219                     | 441                     |
| $\sigma$      | $\frac{8}{\sqrt{2\pi}}$ | $\frac{8}{\sqrt{2\pi}}$ | $\frac{8}{\sqrt{2\pi}}$ |

The CKKS encryption without employing a bootstrapping allows a predetermined depth for multiplication. Thus, for interested users, it is important to note the largest depth of ciphertext multiplications needed to evaluate the algorithm at hand. For example, TD(0) update rule considered in equation (20) requires  $\mathbf{c}_\alpha \cdot \mathbf{c}_\gamma \cdot \mathbf{c}_{v'}(t)$  at the most. This is factored into the design your encryption parameters.

To examine the effect of encryption noise, we created two tables. One keeps track of the values of un-encrypted updates and the other keeps track of the values updated over HE. We recorded the error between two values through each iteration. At final stages of learning, these errors were confirmed to be bounded by some constant of very small magnitude. Although formal convergence analysis of model-free algorithms such as TD(0) are currently not available in this paper, simulation results suggest that they can be performed in the encrypted domain as equally well. Formal analysis of these algorithms based on the analysis already done is left as future research.

### A. Prediction: TD(0)

We implement a GLIE (greedy in the limit with infinite exploration) type learning policy seen in [26], where the client starts completely exploratory ( $\varepsilon = 1.00$ ) and slowly becomes greedy ( $\varepsilon = 0.00$ ) with more episodes. The learning rate is set to be 0 for non-visited states and for visited states, we set  $\alpha(s, t) = \frac{500}{500 + n(s, t)}$ , with  $n(s, t)$  counting the number of visits to the state  $s$  at time  $t$ , to satisfy the standard learning rate assumptions. The discount factor is  $\gamma = 0.9$ . \*RULE\* for TD(0) is the right hand side of equation (20).

### B. Control: SARSA(0)

The update rule for SARSA(0) is:

$$\hat{Q}_{t+1}(s, a) = \hat{Q}_t(s, a) + \alpha_t(s, a) \delta_t^S, \quad (21)$$

where  $\delta_t^S = (r(s, a) + \gamma \hat{Q}_t(s', a') - \hat{Q}_t(s, a))$ .

The policy for SARSA(0) is also a GLIE (greedy in the limit with infinite exploration) type learning policy used in TD(0). The learning rate  $\alpha(s, t)$  and the discount factor  $\gamma$  are unchanged. \*RULE\* for SARSA(0) is the right hand side of equation (20) after substituting  $\mathbf{c}_v$  and  $\mathbf{c}_{v'}$  with  $\mathbf{c}_Q$  and  $\mathbf{c}_{Q'}$ .

---

**Algorithm 1** Encrypted TD Learning
 

---

**Client (Start)**

- 1: Perform an action  $a$  and state transition  $s \rightarrow s'$  on a policy  $\pi_t$  to earn a reward  $r$ .
- 2: Collect the transition data set  $h_t$ . Index values from the current table using  $h_t$ .
- 3: Encode values (Q-values if SARSA(0); Z-values if Z-learning) and the set  $\theta$  into the encoded messages  $V_t$  and  $\Theta$ , respectively.
- 4: Encrypt  $V_t$  and  $\Theta$  to get  $\tilde{V}$  and  $\tilde{\Theta}$  and upload to the cloud.

**Cloud**

- 1: Extract ciphertexts from  $\tilde{V}$  and  $\tilde{\Theta}$ .

*Example:* extract  $\{c_z, c_{z'}, c_\alpha, c_l, c_{\frac{l}{2}}\}$  (Z learning).

- 2: Update: use the **\*RULE\*** with ciphertexts extracted.
- 3: Upload the result of **\*RULE\***, denoted by ciphertext  $c^*$ , back to the Client.

**Client**

- 1: Decrypt the ciphertext  $c^*$  to get the updated  $\tilde{H}$ .
  - 2: Decode  $\tilde{H}$  to get the newly synthesized table of values.
  - 3: Update the policy  $\pi_t$  if necessary.
  - 4: go to **Start**
- 

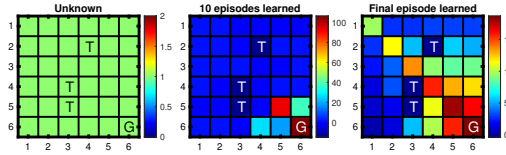


Fig. 2. State-value map learned over encrypted TD(0); states are numbered from 1-36 (from left-right and top-to-bottom) with G (goal state), and T (trap states) marked.

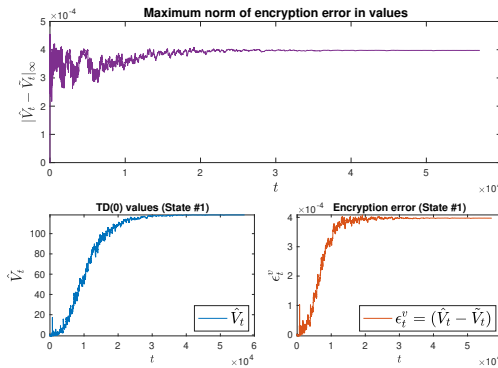


Fig. 3. Maximum norm of encryption errors in TD(0) over each iterate  $t$  (top). As seen in the un-encrypted value updates (bottom-left) and encryption error (bottom-right) plots of state 1, the value of state 1 gets increased as it propagates from reaching goal states more often once the policy becomes greedy (see Fig. 2) and its error dominates between iterates  $t = 30000$  and  $35000$ .

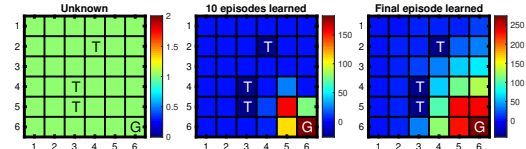


Fig. 4. Value map learned over encrypted SARSA(0)

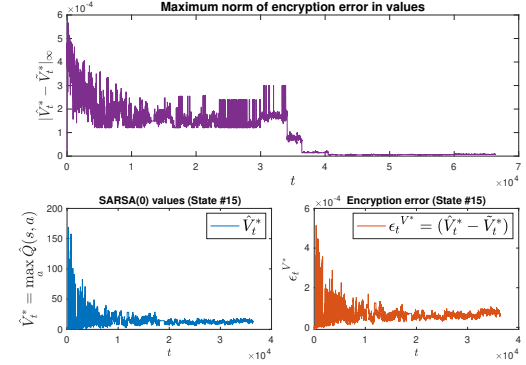


Fig. 5. Maximum norm of encryption errors in SARSA(0) over each iterate  $t$  (top). The un-encrypted and decrypted values for each state are computed using the equation  $V^* = \max_a Q(s, a)$ . Bottom-left shows the sample history of value  $V^*(s = 15)$  and the associated encryption error  $\hat{V}_t^\pi(s = 15) - \tilde{V}_t^\pi(s = 15)$

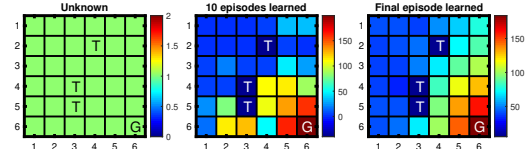


Fig. 6. Z value map learned over encrypted Z-learning

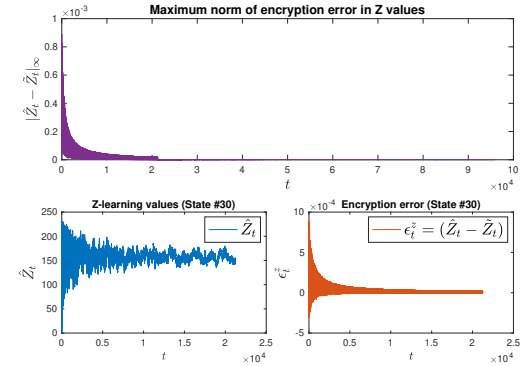


Fig. 7. Maximum norm of encryption errors in Z-learning over each iterate  $t$  (top). The state 30 converges to the highest Z value as seen in Fig. 4. Un-encrypted Z value updates show more variance than TD(0) counterparts due to constant exploration. Encryption error in state 30 is the highest error until about  $t = 22000$  which can be seen in maximum norm but it is bounded by a very small number.

### C. Control: Z-Learning

An off-policy learning control called Z-learning was proposed in [15]. It is formulated on the key observation that the control action can be regarded as an effort to change the passive state transition dynamics. The update rule for Z-learning is:

$$\hat{Z}_{t+1}(s) = \hat{Z}_t(s) + \alpha_t(s)\delta_t^Z, \quad (22)$$

where  $\delta_t^Z = \exp(-l_t) \hat{Z}_t(s') - \hat{Z}_t(s)$ . The estimate  $Z$  is named the desirability function and it uses the cost  $l_t$  associated with the unknown state-transitions, rather than the reward. Z-learning by default is exploratory. Thus, we fix the policy to be greedy ( $\varepsilon = 1.00$ ) but it will continuously explore, too. The learning rate  $\alpha(s, t)$  and the discount factor are unchanged. \*RULE\* for Z-learning is evaluating the right hand side of equation (22) after encrypting. Since evaluating  $\exp(-l_t)$  over a ciphertext is not straightforward, we approximate with Taylor series.

#### D. Results

We observed that encryption-induced noise over time approached some small numbers with minimal fluctuations. Moreover, the effects of encryption-induced noise were minimal regarding the convergence of values. This observation complements the analysis on VI in Section III as the TD-algorithms are sampling based value estimation algorithms.

Encryption and decryption are all done at the client's side and so significant computation time is expected for the client. For this reason, the ideal application will require less frequent needs for uploading and downloading and more advanced synthesis procedure that operate on a large set of data.

#### V. CONCLUSION AND FUTURE WORK

We considered an architecture of confidential cloud-based RL over LHE. For the model-based RL, we showed that the impact of the encryption noise on the convergence performance can be analytically bounded. For the model-free RL, we numerically tested implementations of TD(0), SARSA(0), and Z-learning and numerically confirmed that the impacts of the encryption noise on these algorithms are also minimal. Although the applications considered do not necessarily require the cloud, we can develop the framework to adopt to more advanced synthesis algorithms in the future.

There are numerous directions to extend this paper. First, the effort to derive analytical performance guarantees for encrypted RL (including the model-free schemes considered in this paper) is necessary to prove the utility of the encrypted RL concept. Second, an encrypted RL scheme that does not require periodic decryption (similar to the case in [27]) is highly desired as the periodic decryption and communication between the cloud and the controller is costly. Finally, more extensive numerical experiments are needed to fully understand the potential of advanced RL (e.g., deep Q learning) over HE. The interplay between computational overhead, delay, accuracy and security levels must be studied from both theoretical and experimental perspectives.

#### REFERENCES

- [1] E. Hossain, I. Khan, F. Un-Noor, S. S. Sikander, and M. S. H. Sunny, "Application of big data and machine learning in smart grid, and associated security concerns: A review," *IEEE Access*, vol. 7, pp. 13 960–13 988, 2019.
- [2] M. Mohammadi, A. Al-Fuqaha, M. Guizani, and J. Oh, "Semisupervised deep reinforcement learning in support of IoT and smart city services," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 624–635, April 2018.
- [3] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.
- [4] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, pp. 354–, Oct. 2017.
- [5] I. Arel, C. Liu, T. Urbanik, and A. G. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," *IET Intelligent Transport Systems*, vol. 4, no. 2, pp. 128–135, June 2010.
- [6] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *J. Mach. Learn. Res.*, vol. 17, no. 1, p. 1334–1373, Jan. 2016.
- [7] K. Kogiso and T. Fujita, "Cyber-security enhancement of networked control systems using homomorphic encryption," *2015 54th IEEE Conference on Decision and Control (CDC)*, pp. 6836–6843, 2015.
- [8] F. Farokhi, I. Shames, and N. Batterham, "Secure and private cloud-based control using semi-homomorphic encryption," *IFAC-PapersOnLine*, vol. 49, no. 22, pp. 163 – 168, 2016.
- [9] J. Kim, C. Lee, H. Shim, J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Encrypting controller using fully homomorphic encryption for security of cyber-physical systems," *IFAC-PapersOnLine*, vol. 49, no. 22, pp. 175 – 180, 2016.
- [10] M. Schulze Darup, A. Redder, I. Shames, F. Farokhi, and D. Quevedo, "Towards encrypted MPC for linear constrained systems," *IEEE Control Systems Letters*, vol. 2, no. 2, pp. 195–200, 2018.
- [11] A. B. Alexandru, M. Morari, and G. J. Pappas, "Cloud-based MPC with encrypted data," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 5014–5019.
- [12] M. S. Darup, A. Redder, and D. E. Quevedo, "Encrypted cloud-based MPC for linear systems with input constraints," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 535 – 542, 2018.
- [13] K. Teranishi, M. Kusaka, N. Shimada, J. Ueda, and K. Kogiso, "Secure observer-based motion control based on controller encryption," in *2019 American Control Conference (ACC)*, 2019, pp. 2978–2983.
- [14] J. Kim, H. Shim, and K. Han, "Comprehensive introduction to fully homomorphic encryption for dynamic feedback controller via lwe-based cryptosystem," *CoRR*, vol. abs/1904.08025, 2019.
- [15] E. Todorov, "Efficient computation of optimal actions," *Proceedings of the National Academy of Sciences*, vol. 106, no. 28, pp. 11 478–11 483, 2009.
- [16] D. P. Bertsekas, *Neuro-dynamic programming* *Neuro-Dynamic Programming*. Boston, MA: Springer US, 2009.
- [17] D. Silver, "Lecture 8: Integrating learning and planning." [Online]. Available: <https://www.davidsilver.uk/teaching/>
- [18] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*. The MIT Press, 2018.
- [19] J. N. Tsitsiklis, "Asynchronous stochastic approximation and Q-learning," in *Proceedings of 32nd IEEE Conference on Decision and Control*, Dec 1993, pp. 395–400 vol.1.
- [20] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford University, 2009, [crypto.stanford.edu/craig](https://crypto.stanford.edu/craig).
- [21] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," *Cryptology ePrint Archive*, Report 2016/421, 2016, <https://eprint.iacr.org/2016/421>.
- [22] J. Suh and T. Tanaka, "Encrypted value iteration and temporal difference learning over leveled homomorphic encryption," 2021, [http://sites.utexas.edu/tanaka/files/2021/03/Suh\\_Tanaka\\_ACC2021.pdf](http://sites.utexas.edu/tanaka/files/2021/03/Suh_Tanaka_ACC2021.pdf).
- [23] K. Teranishi and K. Kogiso, "Control-theoretic approach to malleability cancellation by attacked signal normalization," 09 2019.
- [24] M. Albrecht, M. Chase, H. Chen, J. Ding, S. Goldwasser, S. Gorbunov, S. Halevi, J. Hoffstein, K. Laine, K. Lauter, S. Lokam, D. Micciancio, D. Moody, T. Morrison, A. Sahai, and V. Vaikuntanathan, "Homomorphic encryption security standard," *HomomorphicEncryption.org*, Toronto, Canada, Tech. Rep., November 2018.
- [25] "Microsoft SEAL (release 3.4)," <https://github.com/Microsoft/SEAL>, Oct. 2019, microsoft Research, Redmond, WA.
- [26] S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári, "Convergence results for single-step on-policy reinforcement-learning algorithms," *Machine learning*, vol. 38, no. 3, pp. 287–308, 2000.
- [27] J. Kim, H. Shim, and K. Han, "Dynamic controller that operates over homomorphically encrypted data for infinite time horizon," *arXiv preprint arXiv:1912.07362*, 2019.