Feedback Capacity of Ising Channels With Large Alphabet via Reinforcement Learning

Ziv Aharoni, *Student Member, IEEE*, Oron Sabag, *Member, IEEE*, and Haim H. Permuter, *Senior Member, IEEE*

***Abstract*— We propose a new method to compute the feedback capacity of unifilar finite state channels (FSCs) with memory using reinforcement learning (RL). The feedback capacity was previously estimated using its formulation as a Markov decision process (MDP) with dynamic programming (DP) algorithms. However, their computational complexity grows exponentially with the channel alphabet size. Therefore, we use RL, and specifically, its ability to parameterize value functions and policies with neural networks, to evaluate numerically the feedback capacity of channels with a large alphabet size. The outcome of the RL algorithm is a numerical lower bound on the feedback capacity, which is used to reveal the structure of the optimal solution. The structure is modeled by a graph-based auxiliary random variable that is utilized to derive an analytic upper bound on the feedback capacity with the duality bound. The capacity computation is concluded by verifying the tightness of the upper bound by testing whether it is Bahl-Cocke-Jelinek-Raviv (BCJR) invariant. We demonstrate this method on the Ising channel with an arbitrary alphabet size. For an alphabet size smaller than or equal to 8, we derive the analytic solution of the capacity. Next, the structure of the numerical solution is used to deduce a simple coding scheme that achieves the feedback capacity and serves as a lower bound for larger alphabets. For an alphabet size greater than 8, we present an upper bound on the feedback capacity. For an asymptotically large alphabet size, we present an asymptotic optimal coding scheme.**

***Index Terms*— Feedback capacity, reinforcement learning (RL), Ising channels, Markov decision process (MDP), channel capacity.**

I. Introduction

COMPUTING the capacity of finite state channels (FSCs) is a difficult task that has been vigorously researched over the last few decades [2]–[4]. The capacity of FSCs is characterized by the directed information [5]–[8]. Despite the fact that the directed information is a multi-letter expression, it was shown that it can be computed using its formulation as an infinite horizon average reward Markov decision process (MDP) [6], [8]. The MDP formulation of the feedback capacity was introduced in [6], where they also presented a recursive formula for the computation of the MDP's state. However, the recursive formula can be computationally intractable, and consequently, the feedback capacity is difficult to solve.

Nevertheless, there are numerous channel settings in which the recursive formula of the MDP's state can be simplified. This fact motivated subsequent studies to explore more communication settings and channel models in which the feedback capacity can be computed numerically. In [9], the feedback capacity of FSCs with common state information (CSI) at the encoder and the decoder was computed by a recursive formula for the maximum directed information (the feedback capacity). In [10], a family of FSCs with intersymbol interference (ISI) was considered, and the authors applied the value iteration algorithm [11] to compute the feedback capacity of the dicode channel and the run-length-limited (RLL) input over binary symmetric channels. Another scenario, in which the encoder has a feedback link of both past outputs and channel states, was considered in [12]. In this case, the authors formulated the feedback capacity as a dynamic program and derived a single letter expression for the feedback capacity.

A large family of FSCs for which the recursive formula of the MDP's state simplifies is the family of unifilar FSCs. In this case, the MDP formulation of the feedback capacity can be computed using known dynamic programming (DP) algorithms, such as the value iteration algorithm [11]. Beyond the computation of the feedback capacity, this approach was used to produce analytic results of channels with binary alphabets (of the channel input, output and state) [8], [13]–[20]. However, a principal drawback of these algorithms originates in the continuous cardinality of the state and action spaces of the underlying MDP. Previous solutions used a binning technique to quantize these spaces; however, this is tractable only for channels with small input and state alphabets, since the number of bins grows exponentially with the alphabet size.

Another method to compute tractable bounds on the feedback capacity has been proposed in [21], [22]. Here, the authors introduced an auxiliary RV that is modeled by a directed graph called a *Q-graph*. For any fixed Q-graph, single-letter upper and lower bounds on the feedback capacity

Manuscript received 16 August 2020; revised 9 March 2022; accepted 1 April 2022. Date of publication 20 April 2022; date of current version 18 August 2022. This work was supported in part by the German Research Foundation (DFG) via the German–Israeli Project Cooperation (DIP), in part by the Israeli Science Foundation (ISF) under Research Grant 899/21, and in part by the Israeli Innovation Authority as part of the Worldwide Innovative Networking (WIN) Consortium. The work of Oron Sabag was supported in part by the Israeli Scholarship Education Foundation (ISEF) International Postdoctoral Fellowship. An earlier version of this paper was presented in part at the International Symposium on Information Theory (ISIT) 2019 [DOI: 10.48550/arXiv.2001.09685]. *(Corresponding author: Ziv Aharoni.)*

Ziv Aharoni is with the Department of Electrical Engineering, Ben-Gurion University of the Negev, Be'er Sheva 84105, Israel (e-mail: zivah@post.bgu.ac.il).

Oron Sabag is with the Department of Electrical and Computer Engineering, California Institute of Technology, Pasadena, CA 91125 USA.

Haim H. Permuter is with the Department of Computer Engineering, Ben-Gurion University of the Negev, Be'er Sheva 84105, Israel.

Communicated by R. Venkataramanan, Associate Editor for Machine Learning and Statistics, Communications, Signal Processing and Source Coding.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIT.2022.3168729.

Digital Object Identifier 10.1109/TIT.2022.3168729

are given by finite-dimensional optimization problems, where the upper bound is a convex optimization problem [23]. The authors demonstrated their method on binary channels by computing the upper and lower bounds for all possible graphs with up to four nodes. They showed that in all of these cases, the capacity-achieving Q-graph is small (maximum four nodes); however, this is not necessarily the case in general, and the number of all possible graphs grows exponentially with the graph size. Hence, it is of great interest to develop methods that can "reveal" the Q-graph that describes the optimal outputs process. Moreover, to obtain an analytic expression of the capacity, the method in [21] involves solving the Karush-Kuhn-Tucker (KKT) conditions; this can be quite complicated when the Q-graph size is large. Therefore, these algorithms' applicability is limited for the cases where the alphabet size is large.

To address the cardinality constraint of DP algorithms, we propose to use tools from the field of machine learning (ML). ML has been proven to be a strong numerical methodology, and has had a great impact in many research fields. One example in communications is [24], wherein a learning-based algorithm was applied to design a reliable code for the additive white Gaussian noise channel with feedback. Another example is [25], where a reinforcement learning (RL) algorithm was used to measure the information leakage when releasing time-series data in an online manner. The present work introduces a new role for ML in communications: an efficient computation of multi-letter capacity expressions using RL algorithms.

The main advantage of RL is the concept function approximation, which is the key to avoiding quantization of the action and state spaces. Instead, function approximation enables the optimization of policies without visiting the entire state and action spaces. This is the main bypass to the cardinality constraint described above, which makes the evaluation of channels with an alphabet size of ~ 100 tractable. The numerical evaluation provides a numerical lower bound on the feedback capacity. Moreover, for the purpose of deriving the capacity, the numerical results form the basis for conjecturing the structure (by a Q-graph) of the analytic solution.

The structure of the numerical solution is expressed using a directed graph, that is called a Q-graph [21]. The Q-graph nodes and its edges represent a finite subset of the MDP states and their transitions, respectively. This finite subset of states forms an auxiliary RV that is used to obtain an analytic upper bound, specifically, the duality bound for unifilar FSC with feedback [26]. The main advantage of the duality upper bound for unifialr FSCs is that for a fixed Q-graph it can be formulated as an MDP with finite state and action spaces. This allows its computation efficiently using DP algorithms; more importantly, converting the numerical results into analytic results is achieved by verifying the optimality of the Bellman equation. The upper bound is tight in the case where the Q-graph is Bahl-Cocke-Jelinek-Raviv (BCJR) invariant [21, Section III-A]. That is, there exists an input distribution that induce the Markov relation $Y_t - Q_{t-1} - Y^{t-1}$ with the same rate as the upper bound. Thus, the feedback capacity solution is derived.

In our work, the proposed methodology enabled us to compute the feedback capacity of the Ising channel with an alphabet size smaller than or equal to 8. Additionally, in this region, the conjectured structure enabled us to derive a capacity-achieving coding scheme. For an alphabet size greater than 8, we provide an upper bound on the capacity. To analyze the behaviour of the channel for an asymptotic alphabet size, we derive lower and upper bounds that are tight for an asymptotic alphabet size.

The remainder of the paper is organized as follows. Section II includes the necessary preliminaries, and contains notation and the problem definition. In Section III, we present our main results. In Section IV, we demonstrate the usage of RL on the Ising channel. Section V provides the RL algorithms applied in this work, their improvements for the feedback capacity formulation and their implementation. Section VI contains conclusions and a discussion of future work.

II. Preliminaries

This section includes the necessary preliminaries. First, we provide notations. Second, we present the problem definition, which includes the definition of unifilar FSCs, their feedback capacity, and their formulation as an MDP. Third, we present the Q-graph and the Ising channel, on which we demonstrate our methodology.

A. Notation

Calligraphic letters, $\mathcal{X}$, denote alphabet sets, upper-case letters, X, denote random variables, and lower-case letters, x, denote realizations. A superscript, x^t, denotes the vector $(x_1, \ldots, x_t)$. The probability distribution of a random variable, X, is denoted by p_X. We omit the subscript of the random variable when its argument has the same letter, e.g. $p(x|y) = p_{X|Y}(x|y)$. The binary entropy is denoted by $H_2(\cdot)$ and $\log(\cdot)$ refers to the logarithm with base 2.

B. Unifilar Finite State Channels

A FSC is defined by the triplet $(\mathcal{X} \times \mathcal{S}, p(s', y|x, s), \mathcal{S} \times \mathcal{Y})$, where X is the channel input, Y is the channel output, S is the channel state at the beginning of the transmission, and S' is the channel state at the end of the transmission. Also, the cardinalities $\mathcal{X}, \mathcal{Y}, \mathcal{S}$ are assumed to be finite. At each time t, the channel has the memory-less property, that is,

$$p(s_t, y_t|x^t, s^{t-1}, y^{t-1}) = p(s_t|x_t, s_{t-1}, y_t)p(y_t|x_t, s_{t-1}). \tag{1}$$

A FSC is called *unifilar* if the new channel state, s_t, is a time-invariant function $s_t = f(x_t, s_{t-1}, y_t)$.

C. Ising Channel

The Ising channel model was introduced as an information theory problem by Berger and Bonomi in 1990 [27], 70 years after Lenz and Ising first introduced a related Markov model in statistical physics that is now called the 1D Ising model [28].

TABLE I
MDP FORMULATION OF THE FEEDBACK CAPACITY

state	$p_{S_{t-1}\|Y^{t-1}}(\cdot\|y^{t-1})$
action	$p_{X_t\|S_{t-1},Y^{t-1}}(\cdot\|\cdot,y^{t-1})$
reward	$I(X_t,S_{t-1};Y_t\|Y^{t-1}=y^{t-1})$
disturbance	y_t

The Ising channel is a unifilar FSC and is defined by

$$Y=\begin{cases}X & \text{, w.p. } 0.5\\ S & \text{, w.p. } 0.5\end{cases}, \tag{2}$$

$$S'=X. \tag{3}$$

Hence, if $X=S$ then $Y=X=S$ w.p. 1. Otherwise, Y will be one of the last two channel inputs with equal probability. We denote the *channel cardinality* with $|\mathcal{X}|$ since, by definition, $|\mathcal{X}|=|\mathcal{Y}|=|\mathcal{S}|$. The feedback capacity of the binary Ising channel with a symmetric transition probability was derived previously in [13] along with a capacity-achieving coding scheme. In [16], the authors studied the generalized binary Ising channel, in which the channel output equals the input with probability p. The feedback capacity of this family of channels was derived for $p\in[0,0.39]$ along with upper and lower bounds on the capacity without feedback.

D. Feedback Capacity of Unifilar Finite State Channels

The feedback capacity of a unifilar FSC is presented in the following theorem.

Theorem 1: [8, Theorem 1] The feedback capacity of a connected unifilar FSC[1], where the initial state s_0 is available to both the encoder and the decoder, can be expressed by

$$C_{\mathsf{FB}}=\lim_{N\to\infty}\max_{\{p(x_t|s_{t-1},y^{t-1})\}_{t=1}^{N}}\frac{1}{N}\sum_{i=1}^{N}I(X_i,S_{i-1};Y_i|Y^{i-1}). \tag{4}$$

Note that the objective of Theorem 1 is a multi-letter expression and, therefore, its computation is not straightforward; however, it can be computed via an MDP formulation that is given next.

E. Feedback Capacity of Unifilar Finite State Channel as a Markov Decision Process

According to [8] the feedback capacity, as given in Theorem 1, can be formulated as an MDP. The state is the probability vector $z_{t-1}=p_{S_{t-1}|Y^{t-1}}(\cdot|y^{t-1})$, the action is the transition matrix $u_t=p_{X_t|S_{t-1},Y^{t-1}}(\cdot|\cdot,y^{t-1})$, and the reward is $r_t=I(X_t,S_{t-1};Y_t|Y^{t-1}=y^{t-1})$. The next state vector at coordinate s_t is given in Equation (5), shown at the bottom of the next page, where $\mathbb{1}$ denotes the indicator function, $z_{t-1}(s_{t-1})=p(s_{t-1}|y^{t-1})$ and $u_t(x_t,s_{t-1})=p(x_t|s_{t-1},y^{t-1})$. The MDP formulation is summarized in Table I.

[1]A connected unifilar FSC [8, Definition 2] satisfies the condition that for any $s\in\mathcal{S}$ there exists a positive integer $T(s)\in\mathbb{N}$ and an input distribution $\{p(x_i|s_{i-1})\}_{i=1}^{T(s)}$ such that there is a positive probability of reaching all other states with $T(s)$ time steps.

F. Q-Graph

The Q-graph [21] is defined as a directed graph with edges that are labelled with symbols from the channel outputs alphabet $\mathcal{Y}$. By restricting the outgoing edge labels from each node to be distinct, the Q-graph can be used as a mapping of (any-length) output sequences onto the graph nodes by walking along the labelled edges. For a fixed graph, we denote the induced mapping with $\phi:\mathcal{Q}\times\mathcal{Y}\to\mathcal{Q}$, where $\mathcal{Q}$ denotes the set of graph nodes. Given a sequence of channel outputs we denote $Q_i=\Phi_i(Y^i)$, where $\Phi_i=\phi\circ\phi\circ\cdots\circ\phi$ denotes the composition of ϕ, i times. The reader may refer to Figure 4 for an illustration of a Q-graph with 6 nodes. For instance, an initial node $Q_0=(1,0)$ and a sequence of output symbols $Y_1=0,Y_2=2,Y_3=2,Y_4=1$ yields the final node $Q_4=(1,1)$.

III. MAIN RESULTS

In this section, we present RL as a numerical tool used to estimate the feedback capacity. Thereafter, we present the application of RL on the Ising channel with a large alphabet to obtain the capacity, and a capacity-achieving coding scheme for $|\mathcal{X}|\leq 8$. We also show an analytic upper bound on the capacity for $|\mathcal{X}|>8$, and an additional coding scheme and upper bound in order to examine the channel behavior for very large alphabet sizes.

A. Feedback Capacity Estimation Using Reinforcement Learning

We present RL as a numerical tool to solve the feedback capacity of unifilar FSCs using their MDP formulation. Unlike DP algorithms, deep RL uses neural networks (NNs) to parameterize value functions and policies, which makes it a tractable numerical tool for channels with large alphabets[2]. A known issue with RL algorithms is their convergence properties. However, a main advantage of the feedback capacity formulation is that the underlying MDP is fully known. Off-the-shelf RL algorithms were developed under the setting in which the underlying MDP is not known, but here we embed this knowledge into existing algorithms. In Section V, we elaborate on how the knowledge of the underlying MDP is embedded in the RL algorithms used in this work.

The feedback capacity and the optimal input distribution of a unifilar FSC can be computed numerically using two RL algorithms[3]:

1) Deep deterministic policy gradient (DDPG).
2) Policy optimization by unfolding (POU).

The DDPG algorithm [29] is an RL algorithm for MDPs with continuous state and action spaces and deterministic actions. The POU algorithm is a variant of model-based RL algorithms [30], [31] that we developed for the purpose of

[2]The authors emphasizes that other numerical tools could be applied to solve the feedback capacity, such as approximate DP (ADP).

[3]To the best of our knowledge, there are no RL algorithms for deterministic actions and continuous action and state spaces except the DDPG algorithm. Since the dimension of the action and state spaces of the MDP considered here is large, quantization of these spaces is not tractable, and therefore, most of the existing RL algorithms cannot be applied here.

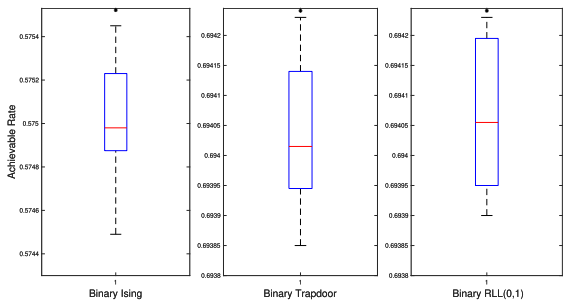


Fig. 1. Illustration of the application of RL on binary unifilar FSCs whose feedback capacity was derived in the past [8], [13], [14]. The achievable rate was computed using a Monte-Carlo simulation, as described in Section V. The black asterisk shows the true capacity and the box plot demonstrates the achievable rates obtained by 10 different seeds. The red central mark indicates the median, and the bottom and top edges of the box indicate the 25$^{\text{th}}$ and 75$^{\text{th}}$ percentiles, respectively. The whiskers extend to the most extreme achievable rates.

this work. In the DDPG algorithm both the value function and the policy are parameterized by NNs, while in POU only the policy is parameterized by a NN. Empirically, DDPG yielded higher numerical lower-bounds for $|\mathcal{X}| \leq 15$ and POU yielded higher numerical lower bounds for $15 < |\mathcal{X}| \leq 150$, and therefore both are presented. In Section V, we present both algorithms. We emphasize that RL is a numerical tool that can be used for any unifilar FSC. Its application on the binary channels investigated in [8], [13], [14] is illustrated in Figure 1.

The RL numerical results reveal bold insights into the structure of the optimal solution of the capacity problem. Specifically, examination of the learned input distribution showed that the visited MDP states are concentrated within a finite subset of states, and therefore can be represented by a Q-graph. The Q-graph is used subsequently to obtain analytic bounds on the feedback capacity, as we present next.

B. Ising Channel

In this section, we present the analytical results for the Ising channel that were deduced from RL numerical simulations, as summarized in Figure 2. The following theorem presents an application of RL to obtain the analytic feedback capacity of the Ising channel for $|\mathcal{X}| \leq 8$.

Theorem 2 (Feedback Capacity): The feedback capacity of the Ising channel with $|\mathcal{X}| \leq 8$ is given by

$$C_{\mathsf{FB}}(\mathcal{X}) = \max_{p\in[0,1]} 2\frac{H_2(p) + (1-p)\log\left(|\mathcal{X}|-1\right)}{p+3}. \quad (6)$$

Equivalently, the feedback capacity can be also expressed as

$$C_{\mathsf{FB}}(\mathcal{X}) = \frac{1}{2}\log\frac{1}{p}, \quad (7)$$

where p is the unique solution of $x^4 - ((|\mathcal{X}|-1)^4 + 4)x^3 + 6x^2 - 4x + 1 = 0$ on $[0, 1]$.

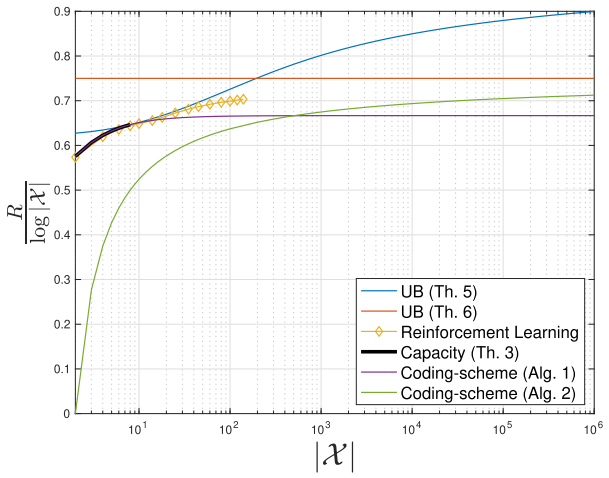


Fig. 2. Summary of the analytic bounds and the numerical results obtained by the RL simulations for varying alphabet sizes. The rates/bounds are normalized by $\log|\mathcal{X}|$. The RL curve was computed by a Monte-Carlo evaluation of the estimated policy. The experiment was run 10 times and the lowest achievable rate was plotted.

The proof of Theorem 2 involves computing the duality upper bound for unifilar FSCs [26] and verifying its tightness; it is given in Section IV-B.1. Algorithm 1 describes a simple coding scheme that achieves the feedback capacity in Theorem 2. Algorithm 1 is applicable for any alphabet size; however, it is optimal only for $|\mathcal{X}| \leq 8$ as stated in the following theorem.

Theorem 3 (Optimal coding scheme): The coding scheme in Algorithm 1 achieves the capacity in Theorem 2 for $|\mathcal{X}| \leq 8$.

In Section IV-B.3, we prove that the coding scheme in Algorithm 1 yields a zero-error code and that its maximum rate over the parameter p equals the feedback capacity as given in Theorem 2. The enumerative source encoding [32] procedure is described after the proof of Theorem 3 in Section IV-B.3.

For $|\mathcal{X}| > 8$, the structure of the analytic solution changes. Unlike the solution for $|\mathcal{X}| \leq 8$, the Q-graph induced by the numerical results cannot be described with a finite set of nodes. Nevertheless, the numerical results dictate a sub-optimal structure that induces an upper bound for $|\mathcal{X}| > 8$. The upper bound for $|\mathcal{X}| > 8$ is shown in the following theorem.

Theorem 4 (Upper Bound for $|\mathcal{X}| > 8$): The feedback capacity of the Ising channel with $|\mathcal{X}| > 8$ satisfies

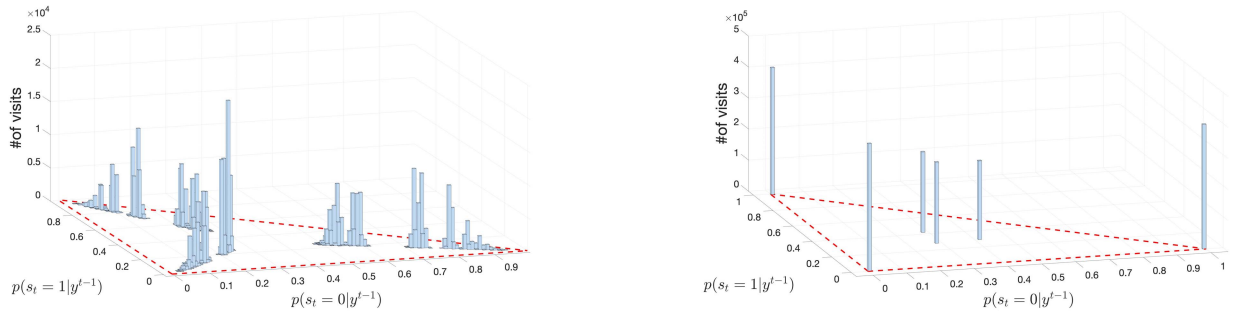
$$\mathsf{C}_{\mathsf{FB}}(\mathcal{X}) \leq \frac{1}{2}\log\frac{|\mathcal{X}|}{p}, \quad (8)$$

where p is the unique root of $x^2 - \left(2 + \frac{(|\mathcal{X}|-1)^2}{16|\mathcal{X}|}\right)x + 1 = 0$ in $[0, 1]$.

For any alphabet size, an upper bound on the capacity and a coding scheme are presented in the following theorem.

Theorem 5 (Upper Bound for $|\mathcal{X}| > 2$): For any alphabet size $|\mathcal{X}| > 2$, the feedback capacity of the Ising channel

$$z_t(s_t) = \frac{\sum_{x_t,s_{t-1}} z_{t-1}(s_{t-1})\, u_t(x_t, s_{t-1})\, p(y_t|x_t, s_{t-1})\mathbb{1}[s_t = f(x_t, s_{t-1}, y_t)]}{\sum_{x_t,s_{t-1},s'_t} z_{t-1}(s_{t-1})\, u_t(x_t, s_{t-1})\, p(y_t|x_t, s_{t-1})\mathbb{1}[s'_t = f(x_t, s_{t-1}, y_t)]}. \quad (5)$$



(a) State histogram in training

(b) Final state histogram

Fig. 3. State histogram of the policy as learnt by RL. The histogram is generated by a Monte-Carlo evaluation of the policy: (a) histogram of the policy after 1000 training iterations; (b) histogram of the policy after convergence.

satisfies

$$C_{\mathsf{FB}}(\mathcal{X}) \leq \frac{3}{4}\log|\mathcal{X}|. \tag{9}$$

Also, for any alphabet size $|\mathcal{X}|> 2$, there is a simple coding scheme with the following rate:

$$R(\mathcal{X}) = \frac{3}{4}\log\frac{|\mathcal{X}|}{2}. \tag{10}$$

Therefore, $\frac{3}{4}\log\frac{|\mathcal{X}|}{2} \leq C_{\mathsf{FB}}(\mathcal{X}) \leq \frac{3}{4}\log|\mathcal{X}|$.

The proof of Theorem 5 is given in Section IV-B.4.

The analytical results and the numerical results of the RL algorithms are summarized in Figure 2. The RL simulation is the yellow curve. We simulated the RL algorithms up to a size of 150 due to a computational memory constraint. The bold-black curve illustrates the analytical capacity that appears in Theorem 2 for $|\mathcal{X}| \leq 8$. One can see that the capacity-achieving coding scheme (in purple) coincides with the RL simulation for $|\mathcal{X}| \leq 8$. However, it converges to $\frac{2}{3}\log|\mathcal{X}|$ for large alphabets, while RL continues to improve. To back up this observation, our improved lower and upper bounds for the asymptotic case (green and orange curves, respectively) are shown. For large $|\mathcal{X}|$, both converge to $\frac{3}{4}\log|\mathcal{X}|$ with a constant difference of $\frac{3}{4}$. We also present the upper bound from Theorem 4, which outperforms the others for $8 < |\mathcal{X}| \leq 200$.

IV. The Ising Channel

This section demonstrates the usage of RL to obtain analytic results on the Ising channel. First, we describe a methodology to convert the numerical results into analytic results. Then, we demonstrate the implementation on the Ising channel.

A. Converting the Numerical Results Into Analytic Bounds

In this section, we describe the conversion of numerical results into analytic results; specifically, we demonstrate this method on the Ising channel with $|\mathcal{X}| = 3$. First, we describe how to extract the structure of the numerical solution. Afterwards, we present how to use the structure to obtain an analytic upper bound and how to verify whether this bound is tight.

Algorithm 1 Capacity-Achieving Coding Scheme for $|\mathcal{X}| \leq 8$

Code construction and initialization:

- Transform the n uniform bits of the message into a stream of symbols (from $\mathcal{X}$) with the following statistics:

$$\nu_i = \begin{cases} \nu_{i-1} & \text{, w.p. } p \\ \text{Unif}[\mathcal{X}\backslash\{\nu_{i-1}\}] & \text{, w.p. } 1-p, \end{cases} \tag{11}$$

  with $\nu_0 = 0$. The mapping can be done using enumerative source encoding [32]
- Transmit a symbol twice to set the initial state of the channel s_0

Encoder:

Transmit ν_t and observe y_t
if $y_t = s_{t-1}$ **then**
  Re-transmit ν_t
end if

Decoder:

Receive y_t
if $y_t \neq y_{t-1}$ **then**
  Store y_t as an information symbol
else
  Ignore y_t and store y_{t+1} as a new information symbol
end if

1) Extracting the Structure of the Optimal Solution: The output of the RL algorithm contains the actor, a parametric model of the input distribution of the channel. This network is used to obtain the structure of the solution by the following procedure. First, it is used for a Monte-Carlo evaluation of length n of the communication rate. During this evaluation, the MDP states and the channel outputs are recorded. These states are then clustered using common techniques, such as the k-means algorithm [33]. For instance, in Figure 3 the MDP state histogram of the Ising channel with $|\mathcal{X}| = 3$ is shown, and it is clear that the estimated solution has only six discrete states. Therefore, the sequence of MDP states $\{Z_i\}_{i=1}^n$ is converted into a sequence of auxiliary RVs $\{Q_i\}_{i=1}^n$ with a discrete alphabet $\mathcal{Q}$, where each value in $\mathcal{Q}$ forms a node

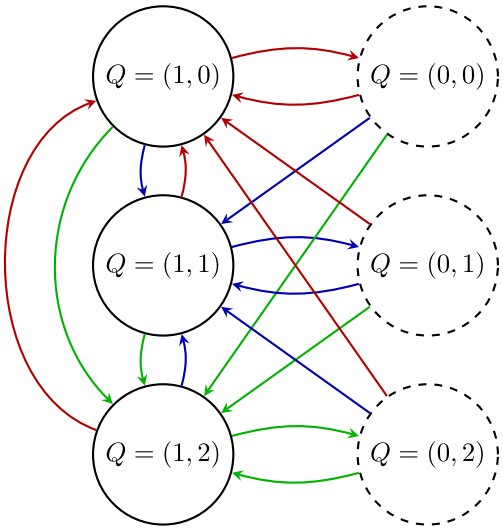


Fig. 4. Q-graph showing the transitions between states as a function of the channel's output. Red, blue and green lines correspond to $Y=0,1,2$, respectively. States with solid lines and dashed lines indicate whether the channel state is known or unknown to the decoder, respectively.

of the Q-graph. The transitions between nodes are determined uniquely[4] by the channel outputs, and the corresponding test distribution $P_{Y_i|Q_{i-1}} = T_{Y|Q}$ is estimated by counting the channel output frequency at every Q-graph node. The Q-graph for $|\mathcal{X}| = 3$ is shown in Figure 4. This completes the generation of the induced Q-graph and $T_{Y|Q}$.

2) Upper Bound Using the Extracted Structure: The upper bound is derived by using the Q-graph and $T_{Y|Q}$ in the duality bound for the unifilar FSC with feedback, as presented in [26]. The duality bound is given in the following theorem.

Theorem 6 [26, Theorem 4]: For any choice of Q-graph and test distribution $T_{Y|Q}$, the feedback capacity of a connected unifilar FSC is bounded by

$$C_{\mathsf{FB}} \leq \lim_{n\to\infty} \max_{f(x^n\|y^n)} \max_{s_0,q_0} \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[D_{KL}\left(P_{Y|X,S}\left(\cdot|x_i,S_{i-1}\right)\|T_{Y|Q}\left(\cdot|Q_{i-1}\right)\right)\right]. \tag{12}$$

The notation $f(x^n\|y^n)=\prod_i \mathbb{1}\left[x_i=f_i\left(x^{i-1},y^{i-1}\right)\right]$ stands for the causal conditioning of deterministic functions.

The Q-graph transition function is denoted by $\phi:\mathcal{Q}\times\mathcal{Y}\to\mathcal{Q}$, where $\phi(q,y)$ is the node followed by a transition from node $Q=q$ when the channel output is $Y=y$.

The upper bound defines an infinite horizon average reward MDP, as described in Table II, whose average reward is the upper bound on the feedback capacity in Theorem 6. Unlike the MDP of the feedback capacity of Theroem 1, this MDP has finite state and action spaces. Therefore, its evaluation is tractable with DP algorithms, such as the value iteration algorithm. For this purpose, the corresponding Bellman equation is

$$\rho + V(s,q) = \max_{x\in\mathcal{X}} D_{KL}\left(P_{Y|X=x,S=s}\|T_{Y|Q=q}\right) + \sum_{y\in\mathcal{Y}} p(y|x,s)V\left(x,\phi(q,y)\right), \tag{13}$$

[4]The disturbance is the only randomness of the transition between RL states.

TABLE II
MDP FORMULATION OF THE DUALITY UPPER BOUND

state	(s_{t-1},q_{t-1})
action	x_t
reward	$D_{KL}\left(P_{Y\|X=x_t,S=s_{t-1}}\|T_{Y\|Q=q_{t-1}}\right)$
disturbance	y_t

where the term $V(s,q)$ is the value function and ρ is the average reward.

Since the state and action spaces are finite, the Bellman equation defines a finite set of non-linear equations. Removing the non-linearity is achieved by solving the Bellman equation numerically using the value iteration algorithm. The solution includes an estimate of the value function and the average reward, but more importantly, it provides a conjectured optimal policy

$$x(s,q) = \arg\max_{x\in\mathcal{X}} D_{KL}\left(P_{Y|X=x,S=s}\|T_{Y|Q=q}\right) + \sum_{y\in\mathcal{Y}} p(y|x,s)V^*\left(x,\phi(q,y)\right), \tag{14}$$

where V^* is the estimated optimal value function. Substituting $x(s,q)$ in the Bellman equation converts it to a set of linear equations that are simple to solve and one obtains a conjecture of the optimal value function and average reward. Finally, the conjectured value function and average reward are verified as the optimal (fixed point) solution using the Bellman equation to complete the bound.

The bound is tested to be tight by verifying that the structure satisfies two conditions. The first is the Markov $Y^{i-1}-Q_{i-1}-Y_i$, which means that there exists an input distribution that visits only the MDP states that formed the Q-graph and yields an output distribution that satisfies $P_{Y_i|Y^{i-1}} = T_{Y|Q}$. The second condition is that the rate of this distribution equals the upper bound. In that case, the bound is tight, which completes the proof. In the next section, we demonstrate this methodology on the Ising channel with alphabet $|\mathcal{X}|\leq 8$, and $|\mathcal{X}|>8$.

B. Bounds on the Ising Channel

In this section we present our results on the Ising channel. First, we derive the feedback capacity of the Ising channel with $|\mathcal{X}|\leq 8$ by providing a tight upper bound. In this case, we also present a capacity-achieving coding scheme. Next, we provide an upper bound for $|\mathcal{X}|>8$. Finally, we provide an additional coding scheme and prove it is optimal for an asymptotic alphabet size.

1) Capacity for $|\mathcal{X}|\leq 8$: After applying the RL algorithm, we obtain a model of the input distribution. We use this model to conduct a Monte-Carlo evaluation of the communication rate using the MDP formulation. Then, the visited states are clustered using the k-means algorithm. Each cluster is a distinct value of $\mathcal{Q}$ that corresponds to an MDP state. Let us denote each node in the graph by the tuple (q_d,q_s),

$$q_d = \begin{cases} 1, & \text{decoder knows the channel state} \\ 0, & \text{decoder does not know the channel state} \end{cases}$$
$$q_s = \text{last known channel state}, \tag{15}$$

where the decoder knows the channel state if $P_{S_t|Y^t}(\cdot|y^t)$ contains a symbol w.p. 1. The Q-graph is defined by

$$(q'_d, q'_s) = \begin{cases} (1, y) & q_d = 0 \text{ or } q_d = 1, y \neq q_s \\ (0, y) & q_d = 1, y = q_s \end{cases}, \tag{16}$$

as given in Figure 4. Finally, $T_{Y|Q}$ is estimated by counting channel outputs at each node, and according to the transitions between nodes, edges are filled in the Q-graph. A parameterized version of $T_{Y|Q}$ is given in (53).

The Q-graph and $T_{Y|Q}$ are plugged into the duality bound as described in the previous section. Then, the value iteration algorithm is applied on the upper bound to obtain $x(s, q)$. This allows the conversion of the Bellman equation into a set of linear equations. Consequently, we conjecture the value function and average reward, that are proven as optimal, as given in Lemma 1.

Lemma 1: For a fixed Q-graph and $T_{Y|Q}$ the function

$$V(s,q) = \begin{cases} \rho & , q_d = 1, q_s = s \\ 1 + 1.5\rho & , q_d = 1, q_s \neq s \\ 2\rho - 1 + \log(1+p) & , q_d = 0, q_s = s \\ 1.5\rho + \log(1+p) & , q_d = 0, q_s \neq s \end{cases} \tag{17}$$

and the constant $\rho = \frac{1}{2}\log\frac{1}{p}$ satisfy the Bellman equation in Equation (13). The variable p is the only root of $x^4 - ((|\mathcal{X}|-1)^4 + 4)x^3 + 6x^2 - 4x + 1 = 0$ that lies in $[0, 1]$. Equivalently, the optimal average reward can be rewritten as

$$\rho = \max_{p\in[0,1]} 2\frac{H_2(p) + (1-p)\log(|\mathcal{X}|-1)}{p+3}.$$

Lemma 1 provides an upper bound on the feedback capacity, as given in Theorem 6; its proof is given in Appendix A.

The upper bound is verified to be tight by testing if $T_{Y|Q}$ is BCJR invariant. That is, there exists an input distribution whose corresponding output distribution satisfies $P_{Y_i|Y^{i-1}} = T_{Y|Q}$, with the same rate as the upper bound. For this purpose, we conjecture the input distribution by averaging the actions at every Q-graph node. This yields in the following input distribution:

$$p(x_t|s_{t-1}, q_d, q_s) = \begin{cases} p & , q_d = 1, q_s = s_{t-1}, x_t = s_{t-1} \\ \frac{1-p}{|\mathcal{X}|-1} & , q_d = 1, q_s = s_{t-1}, x_t \neq s_{t-1} \\ \text{arbitrary} & , q_d = 1, q_s \neq s_{t-1} \\ 1 & , q_d = 0, x_t = s_{t-1} \\ 0 & , q_d = 0, x_t \neq s_{t-1}, \end{cases} \tag{18}$$

which is a parameterized version of the numerical results. In words, when the decoder knows the channel state, the symbol repeats with probability p; otherwise, another symbol is chosen uniformly over all other symbols. When the decoder does not know the channel state, the state is transmitted again. With the input distribution, the tightness of the bound is verified; this is stated in the following lemma, which completes the derivation of the feedback capacity and the proof of Theorem 2.

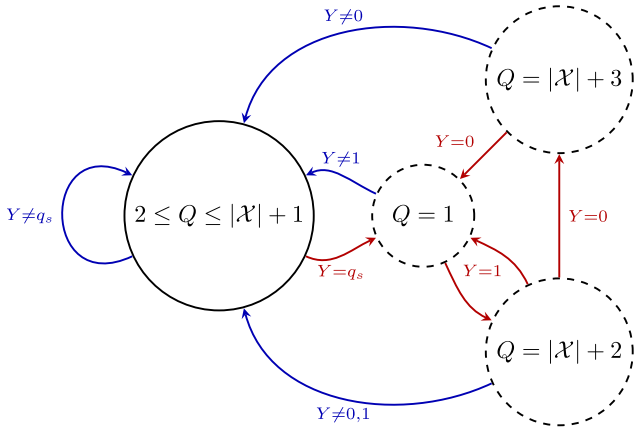


Fig. 5. Q-graph of the Ising channel with $|\mathcal{X}| = 9$. The node $Q = 1$ represents a state where the decoder knows the channel state. At other nodes the channel state is not known exactly to the decoder.

Lemma 2: The input distribution in (18) is BCJR-invariant and its achievable rate is

$$I(X, S; Y|Q) = \max_{p\in[0,1]} 2\frac{H_2(p) + (1-p)\log(|\mathcal{X}|-1)}{p+3}. \tag{19}$$

Therefore, it serves as a tight lower bound as $C_{\mathsf{FB}} \geq \rho$.

The proof of this lemma is given in Appendix A. The combination of Lemma 1 and Lemma 2 concludes the proof of Theorem 2.

2) Upper Bound for $|\mathcal{X}| > 8$: The Q-graph that was optimal for $|\mathcal{X}| \leq 8$ is not optimal for $|\mathcal{X}| > 8$ since the upper bound in this region is not tight. Therefore, we conducted an RL simulation on the Ising channel with an alphabet $|\mathcal{X}| = 9$ to obtain a new Q-graph. In this case, the structure of the solution has a complex histogram, and hence the structure cannot be fully recovered. Instead, we extract a subset of states where most of the transitions occur. This results in a graph with 12 nodes.

We generalize the structure for a general alphabet $|\mathcal{X}| > 2$, as depicted in Figure 5. For simplicity of the graph, all the nodes where the decoder knows the state of the channel are merged into node $2 \leq Q \leq |\mathcal{X}| + 1$. Next, we define the Q-graph with $|\mathcal{X}| + 3$ nodes. Every node where $Q \neq 1$ has a channel state with which it is associated. Therefore, all nodes except $Q = 1$ are denoted by a tuple (q, q_s); for $2 \leq q \leq |\mathcal{X}| + 1$, $q_s = q - 2$, for $q = |\mathcal{X}| + 2, |\mathcal{X}| + 3$ we set $q_s = 0, 1$ (arbitrary choice), respectively. Transitions between nodes as a function of the channel output are depicted in Figure 5. The test distribution $T_{Y|Q}$ is estimated subsequently and its parameterized version is given in (104).

Using the same methodology as in Section IV-A, we use the Q-graph and $T_{Y|Q}$ in the upper bound to obtain an upper bound whose Bellman solution is presented in the following lemma.

Lemma 3: For a fixed Q-graph and $T_{Y|Q}$ the function

$$V(s,q) = \begin{cases} \rho & , q = 1 \\ 2\rho - \log|\mathcal{X}| & , q \neq 1, q_s = s \\ 2\rho + 2 - \log|\mathcal{X}| & , q \neq 1, q_s \neq s \end{cases} \tag{20}$$

and the constant $\rho = \frac{1}{2}\log\frac{|\mathcal{X}|}{p}$ satisfy the Bellman equation. The variable p is the root of $x^2 - (2 + \frac{(|\mathcal{X}|-1)^2}{16|\mathcal{X}|})x + 1 = 0$ that lies in $[0, 1]$.

In Appendix B the proof of Lemma 3 is given, which completes the proof of the upper bound for $|\mathcal{X}| > 8$ as given in Theorem 4.

3) Coding Scheme for $|\mathcal{X}| \leq 8$: The insights from the numerical results led us to derive a capacity-achieving coding scheme for $|\mathcal{X}| \leq 8$, as stated in Theorem 3. This code is a generalization of the optimal coding scheme for $|\mathcal{X}| = 2$ that was presented in [13].

Proof of Theorem 3: The achievable rate is computed by dividing the entropy rate of input symbols by the expected channel uses per one symbol. The entropy rate is computed by the source statistics,

$$H(\nu_i|\nu_{i-1}) = H_2(p) + (1-p)\log\left(|\mathcal{X}|-1\right). \tag{21}$$

The expected channel uses per one symbol ν_i is

$$\mathbb{E}\left[L\right] = p \cdot 2 + (1-p)\cdot 1.5. \tag{22}$$

That is since when $\nu_i = \nu_{i-1}$, the symbol is sent twice, and when $\nu_i \neq \nu_{i-1}$, the symbol is sent once or twice with equal probability. The proof is completed by dividing (21) by (22) and taking a maximum over p. □

In Algorithm 1, it is assumed that the message bits are distributed as a first order Markov source. Since generally the message is uniformly distributed, one can use existing encoding and decoding mappings to convert the message into a sequence with first order Markov statistics. One such mapping is known as enumerative source encoding, which is described in [32, Section III]. Enumerative source encoding involves designing a function that counts all the sequences with an empirical distribution that matches the source statistics (first order Markov chain). This function is used to encode and decode the message into a stream of bits in a recursive calculation.

4) Asymptotic Coding Scheme: We present the proof of Theorem 5. First, we show the upper bound $C_{\mathsf{FB}}(\mathcal{X}) \leq \frac{3}{4}\log(|\mathcal{X}|)$. Then, we show a simple coding scheme with rate $R(\mathcal{X}) = \frac{3}{4}\log\frac{|\mathcal{X}|}{2}$ to complete the proof.

Proof of upper bound in Theorem 5: Let W^n be a sequence of RVs that is defined by

$$W_i = \begin{cases} 1 & , Y_i = X_i \\ 0 & , Y_i = X_{i-1} \end{cases}. \tag{23}$$

Equivalently, W_i indicates whether the output of the channel is the current input or the previous one. By the channel definition W^n is an i.i.d. sequence of RVs, independent of the message M, where $W_i \sim Ber(0.5)$. Now, consider a series of achievable codes $(n, 2^{nR})$ with rate R, an encoder $X_i = f_i(M, Y^{i-1})$ and a decoder $\hat{M}_i = g_i(Y^i)$ with $\mathbb{P}(M \neq \hat{M}_n) \overset{n\to\infty}{\longrightarrow} 0$. A converse for the feedback capacity is then obtained by the following steps:

$$nR = H(M) \tag{24}$$

$$\overset{(a)}{=} H(M|W^n) \tag{25}$$

$$= H(M|W^n) - H(M|Y^n, W^n) + H(M|Y^n, W^n) \tag{26}$$

$$= I(M; Y^n|W^n) + \epsilon_n \tag{27}$$

$$\overset{(b)}{\leq} H(Y^n|W^n) + \epsilon_n \tag{28}$$

$$= \sum_{i=1}^{n} H(Y_i|Y^{i-1}, W^n) + \epsilon_n \tag{29}$$

$$\overset{(c)}{\leq} \sum_{i=1}^{n} H(Y_i|Y_{i-1}, W_i, W_{i-1}) + \epsilon_n \tag{30}$$

$$\overset{(d)}{\leq} n\frac{3}{4}\log|\mathcal{X}| + \epsilon_n, \tag{31}$$

where (a) follows from the independence of M, W^n, (b) follows from the non-negativity of entropy, (c) follows from the fact that conditioning reduces entropy, and (d) is due to

$$\begin{aligned} &H(Y_i|Y_{i-1}, W_i, W_{i-1}) \\ &= \sum_{w_0, w_1 \in \{0,1\}^2} p(w_0, w_1) H(Y_i|Y_{i-1}, W_i = w_1, W_{i-1} = w_0) \\ &= \frac{1}{4}\sum_{w_0, w_1 \in \{0,1\}^2} H(Y_i|Y_{i-1}, W_i = w_1, W_{i-1} = w_0). \end{aligned} \tag{32}$$

By the channel definition, when $W_i = 0, W_{i-1} = 1$ it follows that $Y_i = Y_{i-1}$ and therefore $H(Y_i|Y_{i-1}, W_i = 0, W_{i-1} = 1) = 0$. For $w_0, w_1 \in \{0,1\}^2 \setminus \{0,1\}$, we bound $H(Y_i|Y_{i-1}, W_i = w_1, W_{i-1} = w_0) \leq \log|\mathcal{Y}| = \log|\mathcal{X}|$. Combining the results we obtain $\mathsf{C}_{fb} \leq \frac{3}{4}\log|\mathcal{X}| + \frac{\epsilon_n}{n}$. According to Fano's inequality, $\lim_{n\to\infty}\frac{\epsilon_n}{n} = 0$. Thus, by taking the limit we derive the desired upper bound. □

We show next a simple coding scheme that is asymptotically better than the capacity-achieving coding scheme from Theorem 3. The following proof describes the coding scheme and computes its rate.

Proof of Coding Scheme in Theorem 5: The coding scheme partitions the alphabet into two distinct sets that are assigned uniquely as the sources for odd and even transmission steps. Thus, the encoder sends interchangeably from both sets, which enables the decoder to distinguish whether the channel output was the input or the state of the channel.

a) Code analysis: The rate, $\mathsf{R}(\mathcal{X})$, of the code is computed by dividing the entropy rate of the source by the expected channel uses per one symbol. The entropy rate is $H(\mathcal{X}) = \log\frac{|\mathcal{X}|}{2}$. Let L denote the number of channel uses per one symbol. We compute $\mathbb{E}[L]$ by conditioning on W_0, W_1, RVs that indicate if the output of the channel is the input or the state, as defined in Section IV-B.4. Consequently, it follows that

$$\mathbb{E}\left[L\right] = \sum_{w_0, w_1 \in \{0,1\}^2} p(w_0, w_1)\mathbb{E}\left[L|W_0 = w_0, W_1 = w_1\right] \tag{34}$$

$$= \underbrace{0.5\cdot 1}_{w_0=1} + \underbrace{0.25\cdot 1}_{w_0=0, w_1=0} + \underbrace{0.25\mathbb{E}\left[L|W_0=0, W_1=1\right]}_{\text{symbol is transmitted again}} \tag{35}$$

$$= 0.75 + 0.25(1 + \mathbb{E}\left[L\right]). \tag{36}$$

Algorithm 2 Coding Scheme for Asymptotic $|\mathcal{X}|$

Code construction and initialization:

- Partition $\mathcal{X}$ into two equal-sized disjoint sets $\mathcal{X}_a, \mathcal{X}_b$ (up to one symbol)
- Transform a message of n bits into a stream of symbols from $\mathcal{X}$, denoted by $\nu_1\nu_2\dots$ with the following statistics:

$$\nu_i = \begin{cases} \text{Unif}[\mathcal{X}_b] & , i \text{ even} \\ \text{Unif}[\mathcal{X}_a] & , i \text{ odd} \end{cases} \tag{33}$$

  In words, symbols are drawn uniformly and interchangeably from $\mathcal{X}_a, \mathcal{X}_b$. Denote the source buffers for symbols from $\mathcal{X}_a, \mathcal{X}_b$ as X_a, X_b, respectively
- Generate two output buffers at the decoder Y_a, Y_b
- Transmit a symbol twice to set the initial state of the channel s_0

Encoder:

Transmit ν_t and observe y_t
if $y_t = \nu_t$ and $y_{t-1} = \nu_{t-2}$ **then**
  return ν_{t-1} to the top of its source buffer
end if

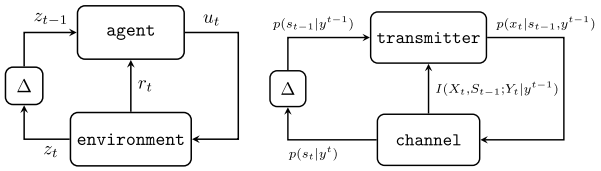
Decoder:

Receive y_t
if $y_t \neq y_{t-1}$ **then**
  if $y_t \in \mathcal{X}_a$ **then**
    store y_t in Y_a
  else
    store y_t in Y_b
  end if
end if

By rearranging we obtain $\mathbb{E}[L] = \frac{4}{3}$. Finally, by dividing the entropy rate by the expected channel uses per one symbol, the rate is obtained as $\mathsf{R}(\mathcal{X}) = \frac{3}{4}\log\frac{|\mathcal{X}|}{2}$. □

V. Computing the Feedback Capacity via Reinforcement Learning

In this section, we give a brief background on RL, based on [34]. Then, we formulate the feedback capacity of a unifilar FSC as an RL problem and provide the algorithms to compute the capacity. An important benefit of the formulation is that the RL environment is completely known, unlike the general assumption in classic RL. Therefore, we leverage the full knowledge of the channel equations and use two algorithms. The first is the DDPG algorithm with improvements; these are based on the knowledge of the environment, and on a prior assumption that the optimal solution has a structure. The second algorithm is POU, a variant of model-based RL algorithms [30], [31], that uses the complete knowledge of the environment to optimize the feedback capacity directly. The DDPG algorithm estimates both the value function and the policy, and therefore its results are easier to interpret. However, it yielded better lower bounds (compared with POU) only for $|\mathcal{X}| \leq 15$, and did not converge for alphabets beyond $|\mathcal{X}| = 15$. The POU algorithm, which only estimates the policy, performed better for $|\mathcal{X}| > 15$ empirically, but was less accurate for $|\mathcal{X}| \leq 15$.



(a) General RL setting (b) Feedback capacity formulated in the RL setting

Fig. 6. A description of (a) the general RL setting and (b) the feedback capacity problem formulated in the RL setting.

A. RL Setting

The RL setting comprises an agent that interacts with a state-dependent environment whose input is an action, and the output is a state and a reward. Formally, at time t, the environment state is z_{t-1}, and an action $u_t \in \mathcal{U}$ is chosen by the agent. Then, a reward $r_t \in \mathcal{R}$ and a new state $z_t \in \mathcal{Z}$ are generated by the environment, and are made available to the agent (Figure 6). The environment is assumed to satisfy the Markov property

$$p\left(r_t, z_t | z^{t-1}, u^t, r^{t-1}\right) = p\left(r_t, z_t | z_{t-1}, u_t\right), \tag{37}$$

and hence, it can be characterized by the time-invariant distribution $p(r_t, z_t|z_{t-1}, u_t)$ only. The agent's *policy* is defined as the sequence of actions $\pi = \{u_1, u_2, \dots\}$.

The objective of the agent is to choose a policy that yields maximal accumulated rewards across a predetermined horizon $h \in \mathbb{N}$. Here, we consider an *infinite-horizon average-reward* setting, where the agent-environment interaction lasts forever, and the goal of the agent is to maximize the average reward gained during the interaction. The average reward of the agent is defined by

$$\rho(\pi) = \lim_{h\to\infty} \frac{1}{h} \sum_{t=1}^{h} \mathbb{E}_\pi[R_t], \tag{38}$$

where the rewards depend on the actions taken according the policy π.

The *differential return* of the agent is defined by

$$G_t = R_t - \rho(\pi) + R_{t+1} - \rho(\pi) + R_{t+2} - \rho(\pi) + \cdots. \tag{39}$$

Accordingly, the state-action value function $Q_\pi(z, u)$ is defined as

$$Q_\pi(z, u) = \mathbb{E}_\pi\left[G_t | Z_{t-1} = z, U_t = u\right], \tag{40}$$

that is, the expected rewards for taking action u at state z and thereafter following policy π. Using the Markov property (37) of the environment, one can write (40) as the sum of the immediate and future rewards, i.e.,

$$\begin{aligned} Q_\pi(z, u) =& \mathbb{E}\left[R_t | Z_{t-1} = z, U_t = u\right] - \rho(\pi) + \\ & \mathbb{E}_\pi\left[Q_\pi(Z_t, U_{t+1}) | Z_{t-1} = z, U_t = u\right], \end{aligned} \tag{41}$$

that is the Bellman equation [34, Chapter 4], which is essential for estimating the function Q_π. Given an estimation of the

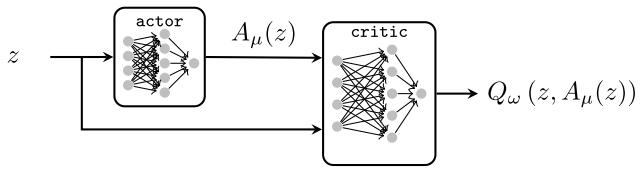


Fig. 7. Depiction of actor and critic networks. The actor network comprises a NN that maps the state z to an action $A_\mu(z)$. The critic NN maps the tuple (z, u) to an estimate of expected future cumulative rewards.

state-action value, it forms the basis for the improvement of a given policy. That is, for each state $z \in \mathcal{Z}$, the current action $\pi(z)$ can be improved to the action $\pi'(z)$ by choosing

$$\pi'(z) = \arg\max_u Q_\pi(z, u). \tag{42}$$

The *function approximators* in RL are parameterized models for $Q_\pi(z, u), \pi(z)$. The *actor* is defined by $A_\mu(z)$, a parametric model of $\pi(z)$, whose parameters are μ. The *critic* is defined by $Q_\omega(z, u)$, a parametric model with parameters ω of the state-action value function that corresponds to the policy $A_\mu(z)$. Generally, in deep RL, the actor and critic are modeled by NNs, as shown in Fig 7.

B. Formulation of the Capacity as an RL

The MDP formulation [8] of the feedback capacity is used to convert the multi-letter capacity formula in Theorem 1 into an RL setting. The formulation is depicted in Figure 6. Under this formulation the state $z_{t-1} = p_{S_{t-1}|Y^{t-1}}(\cdot|y^{t-1})$ is the probability vector of the channel state given the channel outputs feedback. The action $u_t = p_{X_t|S_{t-1}, Z_{t-1}=z_{t-1}}$ is the conditional probability of the channel input conditioned on the channel state. The reward is $r_t = I(X_t, S_{t-1}; Y_t|z_{t-1}, u_t)$ and the next state is given by the evolution of z_t and is described in (5). An equivalent notation denotes $r_t = g(z_{t-1}, u_t), z_t = f(z_{t-1}, u_t, w_t)$, where g, f are the reward and next state function, respectively. The disturbance, $w_t \sim p(w_t|z_{t-1}, u_t)$, is chosen as the channel output y_t, i.e. $w_t = y_t$.

C. Deep Deterministic Policy Gradient (DDPG) Algorithm

In this section we elaborate on the implementation of the DDPG [29], including the necessary adjustments to the feedback capacity formulation.

1) Algorithm: The DDPG algorithm [29] is a deep RL algorithm for deterministic policies and continuous state and action spaces, as needed by the feedback capacity underlying MDP. The DDPG is an extension of the family of deterministic policy gradient algorithms [35] for deep RL. In [35], the gradient theorem has been derived for MDPs with discounted rewards, and here we extend this result for average reward MDPs as well, as given in the following theorem.

Theorem 7: Suppose an ergodic MDP with deterministic actions under the average reward formulation satisfies Assumption 1 in Appendix C. Then,

$$\frac{\partial \rho(\pi)}{\partial \mu} = \int_z d^\pi(z) \frac{\partial A_\mu(z)}{\partial \mu} \frac{\partial}{\partial u} Q_\pi(z, u) \Big|_{u=A_\mu(z)}, \tag{43}$$

where $d^\pi(z) = \lim_{t\to\infty} \mathbb{P}_\pi(Z_t = z|Z_0 = z_0)$. The subscript π indicates the dependency of the stationary distribution on the policy.

The only modification of the policy gradient in Equation (43) from the policy gradient for discounted rewards in [35, Theorem 1] is the definition of $Q_\pi(z, u)$; in the average reward formulation $Q_\pi(z, u)$ is defined with the differential returns in Equation (39), while in [35] the authors used accumulated discounted rewards. The proof of Theorem 7 is given in Appendix C.

The feedback capacity is formulated as an infinite horizon average reward MDP; however, in order to explore the MDP's initial state, the training is divided artificially into episodes. The training procedure comprises M episodes. Each episode starts with a random initial state of the environment, and contains T sequential steps[5]. A single step of the algorithm comprises two parallel *operations*: (1) collecting experience from the environment, and (2) improving the actor and critic networks' performance by training them using the accumulated data.

In the first operation, the agent collects experience from the environment. Given the current state z_{t-1}, the agent chooses an action u_t according to an exploration policy. Here, the action is a probability distribution, and therefore exploration is applied by adding noise to the actor network's last hidden layer, and not by adding noise to the network output as was done in [29]. We denote a noisy action at state z_{t-1} by $A_\mu(z_{t-1}; N_t)$, where $\{N_t\}$ is an i.i.d. Gaussian process with $N_t \sim \mathcal{N}(0, \sigma^2)$. After taking the action $A_\mu(z_{t-1}; N_t)$, the agent observes the incurred reward r_t and the next state z_t. Subsequently, the *transition* tuple

$$\tau = (z_{t-1}, u_t, r_t, z_t)$$

is stored in a *replay buffer*, a bank of experience, that is used to improve the actor and critic networks in the second operation.

The second operation entails training the actor and critic networks. First, N transitions $\{\tau_i\}_{i=1}^N$ are drawn uniformly from the replay buffer. Second, for each transition, the target b_i is computed based on the right-hand-side of (41):

$$b_i = r_i - \rho_{MC} + Q'_\omega\left(z_i, A'_\mu(z_i)\right), \quad i = 1, \ldots, N. \tag{44}$$

The target is the sampled estimate of future rewards; for numerical reasons elaborated in [29], it is computed using a moving average of Q_ω, A_μ, which are the target networks, Q'_ω, A'_μ. The term ρ_{MC} is the estimate of the average reward, which is updated at the beginning of every episode by a Monte-Carlo evaluation of T_{MC} steps by $\frac{1}{T_{MC}} \sum_{t=0}^{T_{MC}-1} r_{t+1}$. Then, we minimize the following objective with respect to the parameters of the critic network ω as given by

$$L(\omega) = \frac{1}{N} \sum_{i=1}^{N} \left[Q_\omega\left(z_{i-1}, A_\mu(z_{i-1})\right) - b_i\right]^2. \tag{45}$$

[5]The episode length T was chosen to be long enough ($T = 500$) to resemble an infinite horizon, but not too long in order to make it possible to explore the MDP's initial state.

Algorithm 3 DDPG Algorithm for Feedback Capacity of Unifilar FSC

Initialize the critic Q_ω and actor A_μ networks with random weights ω and μ, respectively
Initialize target networks Q'_ω and A'_μ with weights $\omega' \leftarrow \omega$, $\mu' \leftarrow \mu$
Initialize an empty replay buffer R
Initialize moving average parameter α
for episode = 1:M **do**
  Initialize a random process $\{N_t\}$ for action exploration
  Set $\rho_{MC} = \frac{1}{T_{MC}} \sum_{t=0}^{T_{MC}-1} r_{t+1}$ by a Monte-Carlo evaluation of the average reward of A_μ
  Randomize initial state z_0 from the $|\mathcal{X}|$-simplex
  for step = 1:T **do**
    Select noisy action $u_t = A_\mu(z_{t-1}; N_t)$
    Execute action u_t and observe (r_t, z_t)
    Store transition (z_{t-1}, u_t, r_t, z_t) in R
    Sample a random batch of N transitions $\{(z_{i-1}, u_i, r_i, z_i)\}_{i=1}^N$ from R
    Set $b_i = r_i - \rho_{MC} + Q'_\omega\left(z_i, A'_\mu(z_i)\right)$
    Update critic by minimizing the loss: $L(\omega) = \frac{1}{N}\sum_{i=1}^N \left[Q_\omega\left(z_{i-1}, A_\mu(z_{i-1})\right) - b_i\right]^2$
    Update the actor policy using the sampled policy gradient:

$$\frac{1}{N}\sum_{i=1}^{N} \nabla_a Q_\omega\left(z_{i-1}, a\right)|_{a=A_\mu(z_{i-1})} \nabla_\mu A_\mu\left(z_{i-1}\right)$$

    Update the target networks:

$$\omega' \leftarrow \alpha\omega + (1-\alpha)\omega'$$
$$\mu' \leftarrow \alpha\mu + (1-\alpha)\mu'$$

  end for
end for
return $\rho_{MC} = \frac{1}{T_{MC}} \sum_{t=0}^{T_{MC}-1} r_{t+1}$

The aim of this update is to train the critic to comply with the Bellman equation (41). Afterwards, we train the actor to maximize the critic's estimation of future cumulative rewards. That is, we train the actor to choose actions that result in high cumulative rewards according to the critic's estimation. The formula for the actor update is given by

$$\frac{1}{N}\sum_{i=1}^{N} \nabla_a Q_\omega\left(z_{i-1}, a\right)|_{a=A_\mu(z_{i-1})} \nabla_\mu A_\mu\left(z_{i-1}\right). \quad (46)$$

Finally, the agent updates its current state to be z_t and moves to the next time step.

To conclude, the algorithm alternates between improving the critic's estimation of future cumulative rewards and training the actor to choose actions that maximize the critic's estimation. The algorithm is given in Algorithm 3 and its workflow is depicted in Figure 8.

2) Improvements: We propose two improvements for the DDPG algorithm. The first improvement uses the knowledge of the environment to reduce the variance of the estimation of Q_π by replacing samples with expectations. Instead of calculating the right-hand-side of (42), as done in (44), we compute the expectation explicitly over all possible next states, rather than using an unbiased estimate of it as was done in the original DDPG algorithm. The modified calculation is given by:

$$b_i = r_i - \rho_{MC} + \sum_{w\in\mathcal{Y}} p(w|z_{i-1}, u_i) Q'_\omega\left(z_i, A'_\mu(z_i)\right). \quad (45)$$

The DDPG algorithm has no theoretical convergence guarantees since it is applied on continuous action and state spaces with non-linear function approximation [29, Section 2]. However, empirically, this method works well due to advances presented in [36]. Since in the DDPG algorithm the Q-function and the policy are learnt using a stochastic approximation procedure, we emphasize that the modification in (45) does not change the expected gradient of the DDPG algorithm. Instead of sampling the disturbance to evaluate the expected gradient, its distribution is used to compute the expectation explicitly according to the law of total expectation. Therefore, the convergence analysis presented in [29] still holds.

The second improvement is a variant of importance sampling [37]. This is essential since there are states that are visited rarely, and in the current technique are rarely used to improve the policy. For this purpose, we modify the replay buffer to store transitions as clusters. Each time a new transition arrives at the buffer, its max-norm distance with all cluster centers is calculated. The distance from the closest cluster is compared with a threshold (typically ~ 0.1). In the case where the distance is smaller than the threshold, the transition is stored in the corresponding cluster; else, a new cluster is added with the new transition. For sampling, instead of drawing transitions uniformly over the entire buffer, we first sample uniformly from the clusters,

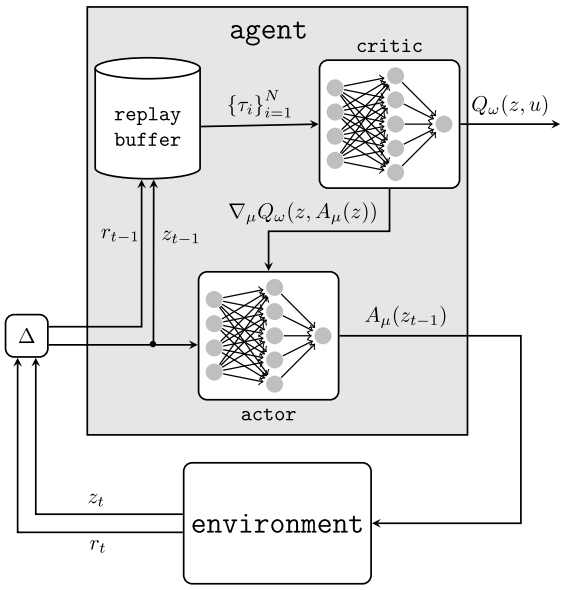


Fig. 8. Depiction of the work flow of the DDPG algorithm. At each time step t, the agent samples a transition from the environment using an ϵ-greedy policy and stores the transition in the replay buffer. Simultaneously, N past transitions $\{\tau_i\}_{i=1}^N$ are drawn from the replay buffer and used to update the critic and actor NN according to (45) and (46), respectively.

and then sample uniformly from within the sampled cluster. The aim of this modification is to draw transitions uniformly over the state space, a $(|\mathcal{X}|-1)$-simplex, and not over the replay buffer[6]. This corresponds to [29, Section 3], where the authors explain that having a large replay buffer increases the probability that uncorrelated examples are drawn from it. The sampling method proposed here encourages uniform sampling by construction.

3) Implementation: We model $Q_\omega(z,u)$, $A_\mu(z)$ with two NNs, each of which is composed of three fully connected hidden layers of 300 units separated by a batch normalization layer. The actor network input is the state z and its output is a matrix $A_\mu(z)\in\mathbb{R}^{|\mathcal{S}|\times|\mathcal{X}|}$ such that $A_\mu(z)\mathbf{1}=\mathbf{1}$. The critic network input is the tuple $(z,A_\mu(z))$ and its output is a scalar, which is the estimate for the cumulative future rewards. In our experiments, we trained the networks for $M=10^4$ episodes. Each episode length is $T=500$ steps. The Monte-Carlo evaluation length of average reward is $T_{MC}=10^8$. For the exploration, we added Gaussian noise with zero mean and variance $\sigma^2=0.05$ to the last layer of the actor network. The implementation details are published in Github[7].

D. Policy Optimization by Unfolding (POU) Algorithm

The POU algorithm utilizes the knowledge of the RL environment to optimize the policy without estimating the value function. That is, we optimize the expected average of consecutive rewards directly. This is done by using the reward function g, the next state function f and the disturbance distribution $P_{W|Z,U}$ to compute gradient of the expected average of n rewards.

1) Algorithm: Let us denote the policy-dependent reward function by

$$R_\mu(z)=g\big(z,\mu(z)\big), \tag{48}$$

which is determined exclusively by z since the policy μ is a deterministic policy. Consequently, we define the expected average reward over n consecutive time steps for an initial MDP state z_0 by

$$\begin{aligned} R^n_\mu(z_0)&=\frac{1}{n}\sum_{t=1}^{n}\mathbb{E}\big[R_\mu(Z_{t-1})\big]\\ &=\frac{1}{n}\left[R_\mu(z_0)+\sum_{t=2}^{n}\mathbb{E}\big[R_\mu(Z_{t-1})\big]\right], \end{aligned}\tag{49}$$

where $Z_t=f(Z_{t-1},\mu(Z_{t-1}),W_t)$ and hence, the expectation is implicitly taken with respect to $P_{W_t|Z_{t-1},\mu(Z_{t-1})}$. That is since the disturbance W_t is conditionally independent of the past given the previous MDP state and the action $\big(Z_{t-1},\mu(Z_{t-1})\big)$.

The choice of the interaction length parameter n affects directly the performance of the optimized policy. Specifically, as n increases the policy is optimized over more rewards in future steps rather than immediate rewards. For instance, choosing $n=1$ translates to optimizing the immediate reward, which consequently yields a greedy policy. As shown in Figure 2, an interaction over relatively small n, e.g., $n\sim 20$, is sufficient to achieve policies with long-term high performance. However, the number of possible MDP states over an interaction of n steps grows exponentially as $|\mathcal{Y}|^n$ (recall the disturbance in our case is the channel output Y).

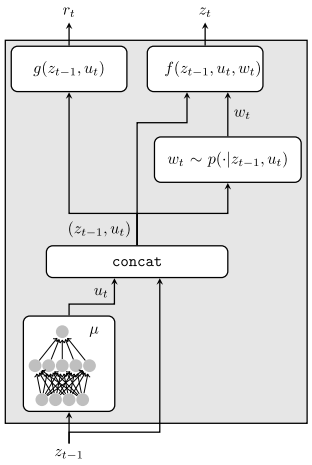


Fig. 9. A single step of the environment. The input of the block is the current RL state z_{t-1} and the outputs are the immediate reward r_t and the next sampled state z_t. Initially, the block uses the actor to construct the tuple (z_{t-1},u_t). Afterwards, it samples the disturbance from $w_t\sim p(\cdot|z_{t-1},u_t)$, and finally, uses g and f to compute the reward and the next state, respectively.

To resolve this practical issue, the POU algorithm proposes a simple, yet efficient, method to *unfold* the interaction with the environment. Given a policy μ and an initial state z_0, we sample n MDP states and rewards consecutively according the following law:

$$\begin{aligned} r_t&=R_\mu(z_{t-1}),\\ W_t&\sim P_{W|Z,U}(\cdot|Z=z_{t-1},U=\mu(z_{t-1})),\\ z_t&=f(z_{t-1},\mu(z_{t-1}),w_t), \end{aligned}\tag{51}$$

where the disturbance W_t is sampled conditioned on the previous MDP state and the action $\mu(Z_{t-1})$. Note that this law is dictated by the RL environment and the chosen policy and is not subject to the planning horizon n. For a single t, the law in (51) describes a single step where the agent interacts with the environment, as shown in Figure 9. The interaction with the environment for n consecutive steps is shown in Figure 10.

Upon differentiating Equation (49), it is evident that $R^n_\mu(z_0)$ depends on the policy parameters through a deterministic mapping from the policy parameters μ to the realized average reward; and through the dependency of $P_{W|Z,U}(w|z,\mu(z))$ on the policy parameters. To take into account these dependencies, we compute an estimated gradient using stochastic computation graphs, as presented in [39]. Figure 11 illustrates the computation graph that it is resulted from our MDP formulation. Circle nodes represent stochastic nodes, i.e. nodes whose output is distributed conditionally on its inputs. Square nodes indicate nodes whose output is a deterministic mapping of its inputs. Equation (50), shown at the bottom of the next

[6]The reader might raise the possibility of considering the idea of priority replay according to the Bellman error [38]. However, since the optimal policy visits only in a finite amount of states, prioritized replay led to drawing the same transition multiple times from the buffer, and thus did not make any positive contribution to the numerical evaluation.

[7]https://github.com/zivaharoni/capacity-rl

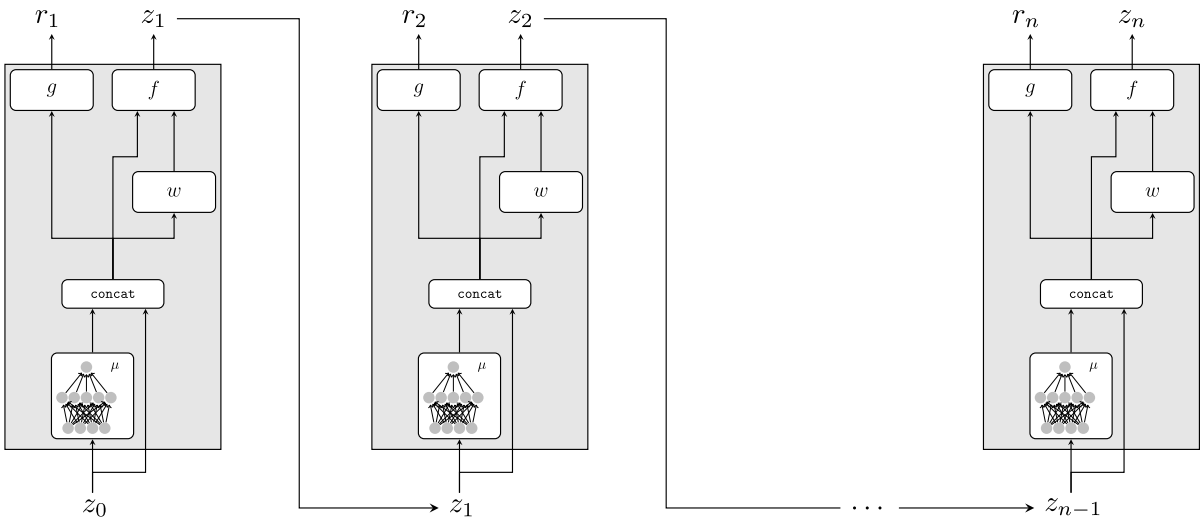


Fig. 10. The interaction with the channel unfolded across subsequent time steps. The weights of the actor network are shared across time steps.

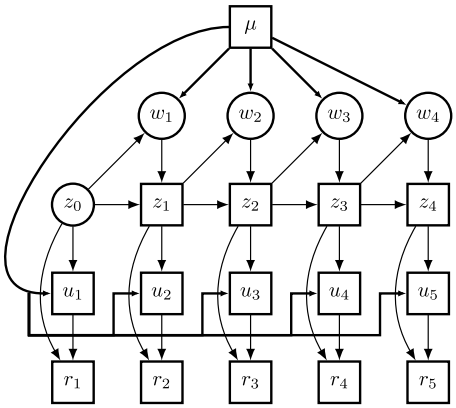


Fig. 11. Illustration of the stochastic computation graph of our MDP formulation for the estimation of the gradient of the POU algorithm. Square nodes indicate deterministic mappings and circle nodes are stochastic. Thick solid lines indicate nodes that depend on the parameters μ.

page, is an application of [39, Theorem 1] and it provides the policy gradient of the POU algorithm, where $c \in \mathbb{R}$ is a constant that is used as a baseline. The policy gradient is composed of two terms. The first term stems from all deterministic trajectories from $\{r_i\}_{i=1}^n$ to μ, i.e. trajectories that go through deterministic nodes only. The second term is obtained from stochastic nodes, e.g. $w_1, \ldots, w_4$ in Figure 11, whose distribution depends on the policy parameters. It aims to increase the likelihood of trajectories with high reward-to-go, i.e. sum of rewards obtained after visiting a stochastic node.

In order to reduce the variance of the algorithm, we subtract a baseline independent of the policy parameters from the realized rewards. As commonly used in RL algorithms [34], [40], we choose the baseline to be a Monte-Carlo evaluation of the average reward that is performed at the beginning of every episode. The update rule of the algorithm is given by using a single trajectory to compute an unbiased estimate of the policy gradient, and it is given by

$$\mu = \mu + \eta \nabla_\mu \left[\frac{1}{n} \sum_{t=1}^{n} \mathbb{E}\left[R_\mu(Z_{t-1}) - \rho_{MC} \right] \right], \tag{52}$$

Algorithm 4 POU Algorithm for Feedback Capacity of Unifilar FSC

Initialize actor A_μ with random weights μ
Initialize learning rate η.
for episode = 1:M **do**
  Evaluate $\rho_{MC} = \frac{1}{T_{MC}} \sum_{t=0}^{T_{MC}-1} r_{t+1}$
  Sample z_0 uniformly from $(|\mathcal{X}|-1)$-simplex
  for $t = 1 : T$ **do**
    Conditioned on z_0 and A_μ, sample $(w_1, \ldots, w_{n-1})$ according to (51)
    Update the actor parameters using gradient ascent

$$\mu = \mu + \eta \nabla_\mu \left[\frac{1}{n} \sum_{t=1}^{n} \mathbb{E}\left[R_\mu(Z_{t-1}) - \rho_{MC} \right] \right]$$

    Update the initial state $z_0 = z_n$
  end for
end for
return $\rho_{MC} = \frac{1}{T_{MC}} \sum_{t=0}^{T_{MC}-1} r_{t+1}$

$$\nabla_\mu \left[\frac{1}{n} \sum_{t=1}^{n} \mathbb{E}\left[R_\mu(Z_{t-1}) - c \right] \right] = \frac{1}{n} \sum_{t=1}^{n} \mathbb{E}\left[\frac{\partial}{\partial \mu} R_\mu(Z_{t-1}) \right] + \frac{1}{n} \sum_{t=2}^{n} \mathbb{E}\left[\frac{\partial}{\partial \mu} \log\left(P_{W|Z,U}(W_t | Z_{t-1}, \mu(Z_{t-1})) \right) \right] \left[\sum_{s=t}^{n} R_\mu(Z_s) - c \right], \tag{50}$$

where η is the step size and the gradient is computed according to Equation (50). This procedure is repeated using the last state z_n as the initial state of the next consecutive n steps. This is shown in Algorithm 4.

2) Implementation: The actor network is implemented exactly as described in Section V-C.3. For training, we trained the actor network for $M = 10^3$ episodes, each containing $T = 10^2$ consecutive n-blocks. Each block was chosen to have length $n = 20$. The Monte-Carlo evaluation length of average reward is $T_{MC} = 10^8$. For exploration, we used dropout [41] on the actor network throughout training. The implementation details are published in Github[8].

VI. Discussion and Conclusion

We proposed a new methodology to compute the feedback capacity of unifilar FSCs. RL is proposed instead of classic DP algorithms due its ability to evaluate MDPs with continuous state and action spaces. Two RL algorithms are proposed to evaluate numerically the feedback capacity. The numerical results form the basis for conjecturing the structure of the optimal solution via a Q-graph and a corresponding $T_{Y|Q}$. The structure is used in the duality bound to obtain an analytic expression of the upper bound, which is tested to be tight by verifying that $T_{Y|Q}$ is BCJR invariant.

We applied this methodology to obtain analytic results over the Ising channel with a general alphabet. For $|\mathcal{X}| \leq 8$, we found the analytic solution of the feedback capacity and derived a capacity-achieving coding scheme. For $|\mathcal{X}| > 8$, the structure in the numerical results enabled us to obtain an upper bound, but we did not manage to verify whether it is tight or not.

An interesting observation is the change of the structure of the solution as the alphabet size increases. Mathematically, the capacity-achieving coding scheme for $|\mathcal{X}| \leq 8$ is optimal for $\mathcal{X} : \mathsf{R}(\mathcal{X}) \leq 2$, as shown in Appendix A. This implies that the transmission policy might differ for the same channel with increasing alphabet size. We visualize this observation in Figure 2, where the efficiency of the coding scheme in Theorem 3, the numerical lower bound from the RL simulation, and the analytic upper bound in Theorem 4 are compared. It is apparent that the solution for $|\mathcal{X}| \leq 8$ saturates at $\frac{2}{3}$, while the numeric lower bound keeps improving when the alphabet increases. This phenomenon is observed when increasing the alphabet size, which emphasizes the importance of developing effective tools for channels with large alphabet sizes.

Appendix A
Upper Bound on the Feedback Capacity for $|\mathcal{X}| \leq 8$

Proof: [Proof of Lemma 1] The conditional distribution $T(y|q)$ that corresponds to the Q-graph extracted from the numerical results is given by

$$T(y|q) = \begin{cases} \frac{1+p}{2} & , q_d = 1, q_s = y \\ \frac{1-p}{2(|\mathcal{X}|-1)} & , q_d = 1, q_s \neq y \\ \frac{2p}{1+p} & , q_d = 0, q_s = y \\ \frac{1-p}{(|\mathcal{X}|-1)(1+p)} & , q_d = 0, q_s \neq y. \end{cases} \tag{53}$$

[8]https://github.com/zivaharoni/capacity-rl-po

Next, we verify that the Bellman equation holds for every (s, q)

$$\rho + V(s,q) = \max_x D_{KL}\left(P_{Y|X=x,S=s} \| T_{Y|Q=q}\right) + \sum_{y\in\mathcal{Y}} p(y|x,s) V\left(x, \phi(q,y)\right). \tag{54}$$

We will start by computing the Bellman equation operator for $q_d = 1, q_s = s$. For $x = s$,

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\left[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\right] \tag{55}$$

$$= -\rho + 1\left[\log\left(\frac{1}{\frac{1+p}{2}}\right) + V\left(s,(1,s)\right)\right] \tag{56}$$

$$= -\rho + 1 - \log(1+p) + 2\rho - 1 + \log(1+p) \tag{57}$$

$$= \rho. \tag{58}$$

Further, if $x \neq s$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\left[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\right] \tag{59}$$

$$= -\rho + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1+p}{2}}\right) + V(x,(0,s))\right]}_{y=s} + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{2(|\mathcal{X}|-1)}}\right) + V(x,(0,x))\right]}_{y=x} \tag{60}$$

$$= -\rho + \frac{1}{2}\Big[-\log(1+p) + 1.5\rho + \log(1+p) - \log\left(\frac{1-p}{(|\mathcal{X}|-1)}\right) + \rho\Big] \tag{61}$$

$$= -\rho + \frac{1}{2}\Big[-\log(1+p) + 1.5\rho + \log(1+p) + 1.5\rho + \rho\Big] \tag{62}$$

$$= \rho. \tag{63}$$

Since, for all x the operator is equal, the Bellman equation is satisfied.

For $q_d = 1, q_s \neq s$
$x = s$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{64}$$

$$= -\rho + 1\left[\log\left(\frac{1}{\frac{1-p}{2(|\mathcal{X}|-1)}}\right) + V\left(s,(1,s)\right)\right] \tag{65}$$

$$= -\rho + 1 - \log\left(\frac{1-p}{(|\mathcal{X}|-1)}\right) + \rho = 1 + 1.5\rho. \tag{66}$$

$x \neq s$, $q_s = x$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{67}$$

$$= -\rho + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{2(|\mathcal{X}|-1)}}\right) + V(x,(1,s))\right]}_{y=s} + \underbrace{\frac{1}{2}\left[\log\left(\frac{\frac{1}{2}}{\frac{1+p}{2}}\right) + V(x,(0,x))\right]}_{y=x} \tag{68}$$

$$= -\rho + \frac{1}{2}\left[-\log\left(\frac{1-p}{(|\mathcal{X}|-1)}\right) + 1 + 1.5\rho)\right] + \frac{1}{2}\left[-\log(1+p) + 2\rho - 1 + \log(1+p)\right] \tag{69}$$

$$= -\rho + \frac{1}{2}\left[1.5\rho + 1 + 1.5\rho)\right] + \frac{1}{2}\left[2\rho - 1\right] = 1.5\rho. \tag{70}$$

$x \neq s$, $q_s \neq x$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{71}$$

$$= -\rho + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{2(|\mathcal{X}|-1)}}\right) + V(x,(1,s))\right]}_{y=s} + \underbrace{\frac{1}{2}\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{2(|\mathcal{X}|-1)}}\right) + V(x,(0,x))\right]}_{y=x} \tag{72}$$

$$= -\rho + \frac{1}{2}\left[-\log\left(\frac{1-p}{(|\mathcal{X}|-1)}\right) + 1 + 1.5\rho)\right] + \frac{1}{2}\left[-\log\left(\frac{1-p}{(|\mathcal{X}|-1)}\right) + \rho\right] \tag{73}$$

$$= -\rho + \frac{1}{2}\left[1.5\rho + 1 + 1.5\rho)\right] + \frac{1}{2}\left[1.5\rho + \rho\right] = 1.75\rho + 0.5. \tag{74}$$

Hence, for any cardinality that satisfies $\rho < 2$ the Bellman equation holds ($1.5\rho + 1 > 1.75\rho + 0.5$).

$q_d = 0, q_s = s$

$x = s$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{75}$$

$$= -\rho + 1\left[\log\left(\frac{1}{\frac{2p}{1+p}}\right) + V\left(s,(1,s)\right)\right] \tag{76}$$

$$= -\rho + \log(1+p) - 1 - \log(p) + \rho = 2\rho - 1 + \log(1+p). \tag{77}$$

$x \neq s$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{78}$$

$$= -\rho + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{2p}{1+p}}\right) + V(x,(1,s))\right]}_{y=s} + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{(|\mathcal{X}|-1)(1+p)}}\right) + V(x,(1,x))\right]}_{y=x} \tag{79}$$

$$= -\rho + \frac{1}{2}\left[\log(1+p) - 2 - \log(p) + 1 + 1.5\rho\right] + \frac{1}{2}\left[\log(1+p) - 1 - \log\left(\frac{1-p}{(|\mathcal{X}|-1)}\right) + \rho\right] \tag{80}$$

$$= -\rho + \log(1+p) + \frac{1}{2}\left[-2 + 2\rho + 1 + 1.5\rho\right] + \frac{1}{2}\left[-1 + 1.5\rho + \rho\right] = 2\rho - 1 + \log(1+p). \tag{81}$$

In that case the Bellman equation is satisfied.

$q_d = 0, q_s \neq s$

$x = s$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{82}$$

$$= -\rho + 1\left[\log\left(\frac{1}{\frac{1-p}{(|\mathcal{X}|-1)(1+p)}}\right) + V\left(s,(1,s)\right)\right] \tag{83}$$

$$= -\rho + \log(1+p) - \log\left(\frac{1-p}{(|\mathcal{X}|-1)}\right) + \rho = 1.5\rho + \log(1+p). \tag{84}$$

$x \neq s, q_s = x$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{85}$$

$$= -\rho + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{(|\mathcal{X}|-1)(1+p)}}\right) + V(x,(1,s))\right]}_{y=s} + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{2p}{1+p}}\right) + V(x,(1,x))\right]}_{y=x} \tag{86}$$

$$= -\rho + \frac{1}{2}\Big[\log(1+p) - 1 - \log\left(\frac{1-p}{(|\mathcal{X}|-1)}\right) + 1 + 1.5\rho\Big] + \frac{1}{2}\left[\log(1+p) - 2 - \log(p) + \rho\right] \tag{87}$$

$$= -\rho + \log(1+p) + \frac{1}{2}\left[-1 + 1.5\rho + 1 + 1.5\rho\right] + \frac{1}{2}\left[-2 + 2\rho + \rho\right]$$
$$= 2\rho - 1 + \log(1+p). \tag{88}$$

$x \neq s,\ q_s \neq x$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{89}$$
$$= -\rho + \underbrace{\frac{1}{2}\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{(|\mathcal{X}|-1)(1+p)}}\right) + V(x,(1,s))\right]}_{y=s} + \underbrace{\frac{1}{2}\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{(|\mathcal{X}|-1)(1+p)}}\right) + V(x,(1,x))\right]}_{y=x} \tag{90}$$
$$= -\rho + \frac{1}{2}\Big[\log(1+p) - 1 - \log\left(\frac{1-p}{(|\mathcal{X}|-1)}\right) + 1 + 1.5\rho\Big] + \frac{1}{2}\left[\log(1+p) - 1 - \log\left(\frac{1-p}{(|\mathcal{X}|-1)}\right) + \rho\right] \tag{91}$$
$$= -\rho + \log(1+p) - 1 + 1.5\rho + 0.5 + \frac{5}{4}\rho$$
$$= \frac{7}{4}\rho - 0.5 + \log(1+p). \tag{92}$$

Here, too, the Bellman equation is satisfied for $\rho < 2$. □

Proof: [Proof of Lemma 2] We show that the Markov $Y^{i-1} - Q_{i-1} - Y_i$ holds, and since $Q_i = \Phi(Y^{i-1})$ it is enough to show that $T_{Y|Q} = P_{Y_i|Y^{i-1}}$. Using the MDP state for every node of the Q-graph we obtain the following relation:

$$z_{i-1} = \begin{cases} [0,\ldots,1,\ldots,0] & ,q_d = 1 \\ [\frac{1-p}{(1+p)(|\mathcal{X}|-1)},\ldots,\frac{2p}{1+p},\ldots,\frac{1-p}{(1+p)(|\mathcal{X}|-1)}] & ,q_d = 0, \end{cases} \tag{93}$$

where the unique index corresponds to index q_s. Now we plug the input distribution into (18) to get the following representation:

$$p(y_i|y^{i-1}) = \sum_{x_i,s_{i-1}} z_{i-1} u_i p(y_i|x_i,s_{i-1}), \tag{94}$$

to verify the desired relation.

For $q_d = 1, q_s = y$

$$\sum_{x,s} z_{i-1}(s)u_i(x,s)p(y|x,s) = \sum_x u_i(x,y)p(y|x,y) \tag{95}$$
$$= \underbrace{p}_{x=y} + \underbrace{(|\mathcal{X}|-1)\frac{1-p}{|\mathcal{X}|-1}\frac{1}{2}}_{x\neq y}$$
$$= \frac{1+p}{2}. \tag{96}$$

For $q_d = 1, q_s \neq y$

$$\sum_{x,s} z_{i-1}(s)u_i(x,s)p(y|x,s) = \sum_x u_i(x,y)p(y|x,y) \tag{97}$$
$$= \underbrace{\frac{1-p}{|\mathcal{X}|-1}\frac{1}{2}}_{x=y}. \tag{98}$$

For $q_d = 0, q_s = y$

$$\sum_{x,s} z_{i-1}(s)u_i(x,s)p(y|x,s) = \frac{2p}{1+p}\underbrace{\sum_x u_i(x,y)p(y|x,y)}_{=1} + \frac{1-p}{(1+p)(|\mathcal{X}|-1)}\underbrace{\sum_{x,s\neq y} u_i(x,s)p(y|x,s)}_{=0} \tag{99}$$
$$= \frac{2p}{1+p}. \tag{100}$$

For $q_d = 0, q_s \neq y$

$$\sum_{x,s} z_{i-1}(s)u_i(x,s)p(y|x,s) = \frac{2p}{1+p}\underbrace{\sum_x u_i(x,y)p(y|x,y)}_{=0} + \frac{1-p}{(1+p)(|\mathcal{X}|-1)}\underbrace{\sum_{x,s\neq y} u_i(x,s)p(y|x,s)}_{=1} \tag{101}$$
$$= \frac{1-p}{(1+p)(|\mathcal{X}|-1)}. \tag{102}$$

Since the Markov $Y^{i-1} - Q_{i-1} - Y_i$ holds, the feedback capacity is converted into a single-letter expression $I(X,S;Y|Q)$ as shown in [21]. First, we use $p(x|s,q), p(y|x,s)\mathbb{1}(s' = x)\mathbb{1}(q' = \phi(q,s))$ to compute the transition matrix of the Markov $S_0, Q_0 - S_1, Q_1 - \ldots$ and compute its stationary distribution. It is given by

$$\pi(s,q) = \begin{cases} \frac{2}{|\mathcal{X}|(p+3)} & ,q_d = 1, q_s = s \\ 0 & ,q_d = 1, q_s \neq s \\ \frac{2p}{|\mathcal{X}|(p+3)} & ,q_d = 0, q_s = s \\ \frac{1-p}{|\mathcal{X}|(|\mathcal{X}|-1)(p+3)} & ,q_d = 0, q_s \neq s \end{cases}. \tag{103}$$

Next, we compute the rate

$$H(Y|Q) = \sum_{q=(q_d,q_s)} \pi(q)H(Y|Q=q) = \sum_{q=(1,q_s)} \pi(q)H(Y|Q=q) + \sum_{q=(0,q_s)} \pi(q)H(Y|Q=q)$$
$$= \frac{2}{p+3}H(Y|Q=(1,q_s)) + \frac{p+1}{p+3}H(Y|Q=(0,q_s))$$
$$= \frac{2}{p+3}\left(H_2(p) + (1-p)\log(|\mathcal{X}|-1)\right) + 2\frac{1-p}{p+3},$$

$$
\begin{aligned}
&H(Y|X,S)\\
&= -\sum_{x,s,y,q} \pi(s,q)p(x|s,q)p(y|x,s)\log p(y|x,s)\\
&= -|\mathcal{X}| \sum_{q=(1,q_s),x,s,y} \pi(s,q)p(x|s,q)p(y|x,s)\log p(y|x,s)+\\
&\quad -|\mathcal{X}| \sum_{q=(0,q_s),x,s,y} \pi(s,q)p(x|s,q)p(y|x,s)\log p(y|x,s)\\
&= -\frac{2}{p+3} \sum_{q=(1,q_s),x,y} p(x|q_s,q)p(y|x,q_s)\log p(y|x,q_s)-\\
&\quad |\mathcal{X}| \Bigg[\sum_{q=(0,q_s),x,y} \pi(q_s,q)p(x|q_s,q)p(y|x,q_s)\log p(y|x,q_s)+\\
&\qquad \sum_{q=(0,q_s),x,y,s} \pi(s,q)p(x|s,q)p(y|x,s)\log p(y|x,s)\Bigg]\\
&= -\frac{2}{p+3} \sum_{q=(1,q_s),x\neq q_s,y} p(x|q_s,q)p(y|x,q_s)\log p(y|x,q_s)-\\
&\quad \frac{2p}{p+3}\Bigg[\sum_{q=(0,q_s),x,y} \mathbb{1}[x=q_s]p(y|x,q_s)\log p(y|x,q_s)+\\
&\quad \frac{1-p}{(|\mathcal{X}|-1)(p+3)} \sum_{q=(0,q_s),x,y,s} \mathbb{1}[x=s]p(y|x,s)\log p(y|x,s)\Bigg]\\
&= -\frac{2}{p+3} \sum_{q=(1,q_s),x\neq q_s,y} p(x|q_s,q)p(y|x,q_s)\log p(y|x,q_s)-\\
&\quad \frac{2p}{p+3}\Bigg[\sum_{y} p(y|q_s,q_s)\log p(y|q_s,q_s)+\\
&\quad \frac{1-p}{(|\mathcal{X}|-1)(p+3)} \sum_{x,y} p(y|x,x)\log p(y|x,x)\Bigg]\\
&= -\frac{2}{p+3} \sum_{x\neq q_s,y} \frac{1-p}{|\mathcal{X}|-1} p(y|x,q_s)\log p(y|x,q_s)\\
&= \frac{2(1-p)}{p+3}\frac{1}{|\mathcal{X}|-1}\sum_{x\neq q_s} 1\\
&= \frac{2(1-p)}{p+3}.
\end{aligned}
$$

Thus, there exists an input distribution with $P_{Y_i|Y^{i-1}} = T_{Y|Q}$ and the same rate as the upper bound in Lemma 1. This concludes the proof. □

APPENDIX B
UPPER BOUND ON THE FEEDBACK CAPACITY FOR $|\mathcal{X}|> 8$

Proof: [Proof of Theorem 4] The Q-graph obtained from the numerical results of the RL algorithm applied on the Ising channel with $|\mathcal{X}| = 9$ has 12 nodes, where node $Q = 1$ denotes that the decoder does not know the channel states. Nodes $Q = 2,\ldots,10$ correspond to complete knowledge of the channel state and nodes $Q = 11, 12$ correspond to partial knowledge of the channel state. We denote each node by the tuple (q, q_s) where q indicates the node number and q_s indicates the channel state that this node contains information about. The corresponding conditional distribution $T(y|q)$ of the Q-graph is given by

$$
T(y|q) = \begin{cases} \frac{1}{|\mathcal{X}|} & , q = 1\\ p & , q \neq 1, q_s = y\\ \frac{1-p}{(|\mathcal{X}|-1)} & , q \neq 1, q_s \neq y \end{cases}. \tag{104}
$$

Next, we verify that the Bellman equation holds for every (s,q). That is, we verify that the right-hand-side maximum of (13) equals $V(s,q)$. For this purpose, we use the following identity:

$$
\rho = \frac{1}{3}\log\left(\frac{|\mathcal{X}|(|\mathcal{X}|-1)}{4p(1-p)}\right). \tag{105}
$$

For $q = 1, s \neq 6$
$x = s$

$$
\begin{aligned}
V(s,q) &= -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right)\\
&\quad + V(x,\phi(q,y))\Big] &(106)\\
&= -\rho + 1\left[\log\left(\frac{1}{\frac{1}{|\mathcal{X}|}}\right) + V(s,(1,s))\right] &(107)\\
&= -\rho + \log(|\mathcal{X}|) + 2\rho - \log(|\mathcal{X}|) = \rho. &(108)
\end{aligned}
$$

$x \neq s$

$$
\begin{aligned}
V(s,q) &= -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right)\\
&\quad + V(x,\phi(q,y))\Big] &(109)\\
&= -\rho + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1}{|\mathcal{X}|}}\right) + V(x,(1,s))\right]}_{y=s}\\
&\quad + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1}{|\mathcal{X}|}}\right) + V(x,(1,x))\right]}_{y=x} &(110)\\
&= -\rho + \frac{1}{2}\Big[2\log(\frac{|\mathcal{X}|}{2}) + 2\rho\\
&\quad + 2 - \log(|\mathcal{X}|) + 2\rho - \log(|\mathcal{X}|)\Big]\\
&= \rho. &(111)
\end{aligned}
$$

For $q = 1, s = 6$
$x = s$

$$
\begin{aligned}
V(s,q) &= -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right)\\
&\quad + V(x,\phi(q,y))\Big] &(112)\\
&= -\rho + 1\left[\log\left(\frac{1}{\frac{1}{|\mathcal{X}|}}\right) + V(6,(1,6))\right] &(113)\\
&= -\rho + \log(|\mathcal{X}|) + 2\rho - \log(|\mathcal{X}|) = \rho. &(114)
\end{aligned}
$$

$x \neq s$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{115}$$

$$= -\rho + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1}{|\mathcal{X}|}}\right) + V(x,(1,6))\right]}_{y=s} + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1}{|\mathcal{X}|}}\right) + V(x,(1,x))\right]}_{y=x} \tag{116}$$

$$= -\rho + \frac{1}{2}\Big[2\log(\frac{|\mathcal{X}|}{2}) + 2\rho - \log(|\mathcal{X}|) + 2\rho + 2 - \log(|\mathcal{X}|)\Big]$$

$$= \rho. \tag{117}$$

In that case the Bellman equation is satisfied.
For $q \neq 1, q_s = s$:
$x = s$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{118}$$

$$= -\rho + 1\left[\log\left(\frac{1}{p}\right) + V\left(s,(1,s)\right)\right] \tag{119}$$

$$= -\rho + \log(\frac{1}{p}) + \rho \tag{120}$$

$$= -\rho + \underbrace{\log(\frac{|\mathcal{X}|}{p})}_{2\rho} - \log(|\mathcal{X}|) + \rho = 2\rho - \log(|\mathcal{X}|). \tag{121}$$

$x \neq s$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{122}$$

$$= -\rho + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{p}\right) + V(x,(q,s))\right]}_{y=s} + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{|\mathcal{X}|-1}}\right) + V(x,(q,x))\right]}_{y=x} \tag{123}$$

$$= -\rho + \frac{1}{2}\Big[\log\left(\frac{(|\mathcal{X}|-1)}{4p(1-p)}\right)2\rho - \log(|\mathcal{X}|) + \rho\Big] \tag{124}$$

$$= -\rho + \frac{1}{2}\Big[\underbrace{\log\left(\frac{|\mathcal{X}|(|\mathcal{X}|-1)}{4p(1-p)}\right)}_{3\rho} + 2\rho - 2\log(|\mathcal{X}|) + \rho\Big]$$

$$= 2\rho - \log(|\mathcal{X}|). \tag{125}$$

In that case the Bellman equation is satisfied.
For $q \neq 1, q_s \neq s$
$x = s$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{126}$$

$$= -\rho + 1\left[\log\left(\frac{1}{p}\right) + V\left(s,(q,s)\right)\right] \tag{127}$$

$$= -\rho + \underbrace{\log(\frac{1}{p})}_{2\rho-\log(|\mathcal{X}|)} + \rho = 2\rho - \log(|\mathcal{X}|). \tag{128}$$

$x \neq s, x = q_s$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{129}$$

$$= -\rho + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{p}\right) + V(x,(q,s))\right]}_{y=s} + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{|\mathcal{X}|-1}}\right) + V(x,(q,x))\right]}_{y=x} \tag{130}$$

$$= -\rho + \frac{1}{2}\Big[\underbrace{\log\left(\frac{(|\mathcal{X}|-1)}{4p(1-p)}\right)}_{3\rho-\log(|\mathcal{X}|)} + 2\rho + 2 - \log(|\mathcal{X}|) + \rho\Big]$$

$$= 2\rho - \log(|\mathcal{X}|) + 1. \tag{131}$$

$x \neq s, x \neq q_s$

$$V(s,q) = -\rho + \sum_{y=x,s} p(y|x,s)\Big[\log\left(\frac{p(y|x,s)}{T(y|q)}\right) + V(x,\phi(q,y))\Big] \tag{132}$$

$$= -\rho + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{|\mathcal{X}|-1}}\right) + V(x,(q,s))\right]}_{y=s} + \frac{1}{2}\underbrace{\left[\log\left(\frac{\frac{1}{2}}{\frac{1-p}{|\mathcal{X}|-1}}\right) + V(x,(q,x))\right]}_{y=x} \tag{133}$$

$$= -\rho + \frac{1}{2}\Big[\underbrace{2\log\left(\frac{(|\mathcal{X}|-1)}{1-p)}\right)}_{\rho+1} + 2\rho + 2 - \log(|\mathcal{X}|) + 2\rho - \log(|\mathcal{X}|)\Big]$$

$$= 2\rho - \log(|\mathcal{X}|) + 2. \tag{134}$$

Here, also, the Bellman equation is satisfied. This concludes the proof. □

APPENDIX C
DETERMINISTIC POLICY GRADIENT FOR THE INFINITE HORIZON AVERAGE REWARD MDPS

We begin with two definition on the setting and ergodic MDPs.

Definition 1: An MDP is defined by $\mathcal{M} = \left(\mathcal{Z},\mathcal{U},\mathcal{W},f,g,P_{W|Z,U},P_0\right)$, where $\mathcal{Z}$ is the state space, $\mathcal{U}$ is the action space, $\mathcal{W}$ is the disturbance space, $f : \mathcal{Z}\times\mathcal{U}\times\mathcal{W} \to \mathcal{Z}$ is the next state function, $g : \mathcal{Z}\times\mathcal{U}\times\mathcal{W} \to \mathbb{R}$ is the reward function, $P_{W|Z,U}$ is the disturbance distribution given previous state and action, and P_0 is the distribution of the initial state.

The following definition restricts the result in Theorem 7 of average reward MDPs to ergodic MDPs, similarly to [34, Chapter 10, Section 3].

Definition 2: An MDP $\mathcal{M} = \left(\mathcal{Z},\mathcal{U},\mathcal{W},f,g,P_{W|Z,U},P_0\right)$ is said to be ergodic if for any policy π the induced Markov process is ergodic. This implies that for any $z_0, z \in \mathcal{Z}$ the steady state distribution exists and is independent of the initial state, i.e. $d^\pi(z) = \lim_{n\to\infty}\mathbb{P}_\pi(Z_n = z|Z_0 = z_0)$.

The following assumption, as given in [35, Supplementary Material], is technically to ensure that the limits, integrations and derivatives may be interchanged in the subsequent proof.

Assumption 1: An MDP $\mathcal{M} = \left(\mathcal{Z},\mathcal{U},\mathcal{W},f,g,P_{W|Z,U},P_0\right)$ satisfies:

- $\mathcal{Z},\mathcal{U}$ are compact sets.
- f, g are measureable and bounded functions.
- $f, g, P_{W|Z,U}, P_0$ are continuous with continuous derivatives.

The proof of Theorem 7 is given next.

Proof: [Proof of Theorem 7] The value function V_μ depends on the policy parameters μ through all actions $\pi = \{u_i\}_{i\in\mathbb{N}}$, where $u_i = A_\mu(z_{i-1})$. Hence, the derivative w.r.t. μ is given by $\frac{\partial}{\partial\mu} = \sum_{i\in\mathbb{N}}\frac{\partial}{\partial u_i}\frac{\partial u_i}{\partial\mu}$ according to the chain rule of derivatives. For fixed $z_0 \in \mathcal{Z}$,

$$\begin{aligned}
&\frac{\partial}{\partial\mu}V_\mu(z_0)\\
&= \frac{\partial}{\partial\mu}Q_\mu(z_0,A_\mu(z_0))\\
&= \frac{\partial}{\partial\mu}\Big[g(z_0,A_\mu(z_0)) - \rho(\pi)\\
&\quad + \int_{\mathcal{Z}}\mathrm{d}P_{Z'|Z,U}(z_1|z_0,A_\mu(z_0))V_\mu(z_1)\Big]\\
&\overset{(a)}{=} -\frac{\partial\rho(\pi)}{\partial\mu} + \frac{\partial A_\mu(z_0)}{\partial\mu}\frac{\partial}{\partial u_1}\Big[g(z_0,u_1)\\
&\quad + \int_{\mathcal{Z}}\mathrm{d}P_{Z'|Z,U}(z_1|z_0,u_1)V_\mu(z_1)\Big]\Big|_{u_1=A_\mu(z_0)}\\
&\quad + \int_{\mathcal{Z}}\mathrm{d}P_{Z'|Z,U}(z_1|z_0,A_\mu(z_0))\frac{\partial}{\partial\mu}V_\mu(z_1)\\
&\overset{(b)}{=} -\frac{\partial\rho(\pi)}{\partial\mu} + \frac{\partial A_\mu(z_0)}{\partial\mu}\frac{\partial}{\partial u_1}\Big[g(z_0,u_1) - \rho(\pi)\\
&\quad + \int_{\mathcal{Z}}\mathrm{d}P_{Z'|Z,U}(z_1|z_0,u_1)V_\mu(z_1)\Big]\Big|_{u_1=A_\mu(z_0)}\\
&\quad + \int_{\mathcal{Z}}\mathrm{d}P_{Z'|Z,U}(z_1|z_0,A_\mu(z_0))\frac{\partial}{\partial\mu}V_\mu(z_1)\\
&= -\frac{\partial\rho(\pi)}{\partial\mu} + \frac{\partial A_\mu(z_0)}{\partial\mu}\frac{\partial}{\partial u_1}Q_\mu(z_0,u_1)\Big|_{u_1=A_\mu(z_0)}\\
&\quad + \int_{\mathcal{Z}}\mathrm{d}P_{Z'|Z,U}(z_1|z_0,A_\mu(z_0))\frac{\partial}{\partial\mu}V_\mu(z_1),
\end{aligned}$$

where (a) follows from the chain rule, and (b) follows from the assumption on the ergodicity of the MDP. In partiular, the average reward of an ergodic MDP is independent from the initial state z_0 and the first action u_1. Now, rearranging and integrating w.r.t. the stationary distribution

$$\begin{aligned}
&\int_{\mathcal{Z}} d^\pi(z_0)\frac{\partial\rho(\pi)}{\partial\mu}\\
&= \int_{\mathcal{Z}} d^\pi(z_0)\frac{\partial A_\mu(z_0)}{\partial\mu}\frac{\partial}{\partial u_1}Q_\mu(z_0,u_1)\Big|_{u_1=A_\mu(z_0)}\\
&\quad + \int_{\mathcal{Z}} d^\pi(z_0)\int_{\mathcal{Z}}\mathrm{d}P_{Z'|Z,U}(z_1|z_0,A_\mu(z_0))\frac{\partial}{\partial\mu}V_\mu(z_1)\\
&\quad - \int_{\mathcal{Z}} d^\pi(z_0)\frac{\partial}{\partial\mu}V_\mu(z_0)
\end{aligned}$$

Using the fact that the stationary distribution satisfies $\int_{\mathcal{Z}} d^\pi(z_0)\,\mathrm{d}P_{Z'|Z,U}(z_1|z_0,A_\mu(z_0)) = d^\pi(z_1)$

$$\begin{aligned}
\frac{\partial\rho(\pi)}{\partial\mu} &= \int_{\mathcal{Z}} d^\pi(z_0)\frac{\partial A_\mu(z_0)}{\partial\mu}\frac{\partial}{\partial u_1}Q_\mu(z_0,u_1)\Big|_{u_1=A_\mu(z_0)}\\
&\quad + \int_{\mathcal{Z}} d^\pi(z_1)\frac{\partial}{\partial\mu}V_\mu(z_1) - \int_{\mathcal{Z}} d^\pi(z_0)\frac{\partial}{\partial\mu}V_\mu(z_0)\\
&= \int_{\mathcal{Z}} d^\pi(z_0)\frac{\partial A_\mu(z_0)}{\partial\mu}\frac{\partial}{\partial u_1}Q_\mu(z_0,u_1)\Big|_{u_1=A_\mu(z_0)}.
\end{aligned}$$

□

ACKNOWLEDGMENT

The authors would like to thank the associate editor and the anonymous reviewers for their valuable and constructive comments, which helped to improve the paper and its presentation considerably.

REFERENCES

[1] Z. Aharoni, O. Sabag, and H. H. Permuter, "Computing the feedback capacity of finite state channels using reinforcement learning," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2019, pp. 837–841.

[2] R. G. Gallager, *Information Theory and Reliable Communication*. New York, NY, USA: Wiley, 1968.

[3] D. Blackwell, L. Breiman, and A. J. Thomasian, "Proof of Shannon's transmission theorem for finite-state indecomposable channels," *Ann. Math. Statist.*, vol. 29, no. 4, pp. 1209–1220, Dec. 1958.

[4] R. Gray, M. Dunham, and R. Gobbi, "Ergodicity of Markov channels," *IEEE Trans. Inf. Theory*, vol. IT-33, no. 5, pp. 656–664, Sep. 1987.

[5] J. L. Massey, "Causality, feedback and directed information," in *Proc. Int. Symp. Inf. Theory Appl. (ISITA)*, Nov. 1990, pp. 303–305.

[6] S. Tatikonda and S. Mitter, "The capacity of channels with feedback," *IEEE Trans. Inf. Theory*, vol. 55, no. 1, pp. 323–349, Jan. 2009.

[7] H. H. Permuter, T. Weissman, and A. J. Goldsmith, "Finite state channels with time-invariant deterministic feedback," *IEEE Trans. Inf. Theory*, vol. 55, no. 2, pp. 644–662, Feb. 2009.

[8] H. H. Permuter, P. Cuff, B. V. Roy, and T. Weissman, "Capacity of the trapdoor channel with feedback," *IEEE Trans. Inf. Theory*, vol. 54, no. 7, pp. 3150–3165, Jul. 2009.

[9] J. Chen and T. Berger, "The capacity of finite-state Markov channels with feedback," *IEEE Trans. Inf. Theory*, vol. 51, no. 3, pp. 780–798, Mar. 2005.

[10] S. Yang, A. Kavčić, and S. Tatikonda, "Feedback capacity of finite-state machine channels," *IEEE Trans. Inf. Theory*, vol. 51, no. 3, pp. 799–810, Mar. 2005.
[11] R. Bellman, "A Markovian decision process," *Indiana Univ. Math. J.*, vol. 6, no. 4, pp. 679–684, Apr. 1957.
[12] J. H. Bae and A. Anastasopoulos, "The capacity of Markov channels with noiseless output and state feedback," in *Proc. Inf. Theory Appl. Workshop (ITA)*, Apr. 2010, pp. 1–5.
[13] O. Elishco and H. Permuter, "Capacity and coding for the Ising channel with feedback," *IEEE Trans. Inf. Theory*, vol. 60, no. 9, pp. 5138–5149, Sep. 2014.
[14] O. Sabag, H. H. Permuter, and N. Kashyap, "The feedback capacity of the binary erasure channel with a no-consecutive-ones input constraint," *IEEE Trans. Inf. Theory*, vol. 62, no. 1, pp. 8–22, Jan. 2016.
[15] H. H. Permuter, H. Asnani, and T. Weissman, "Capacity of a post channel with and without feedback," *IEEE Trans. Inf. Theory*, vol. 60, no. 10, pp. 6041–6057, Oct. 2014.
[16] A. Sharov and R. M. Roth, "On the capacity of generalized ising channels," *IEEE Trans. Inf. Theory*, vol. 63, no. 4, pp. 2338–2356, Apr. 2017.
[17] O. Sabag, H. H. Permuter, and N. Kashyap, "Feedback capacity and coding for the BIBO channel with a no-repeated-ones input constraint," *IEEE Trans. Inf. Theory*, vol. 64, no. 7, pp. 4940–4961, Jul. 2018.
[18] J. Wu and A. Anastasopoulos, "On the capacity of the chemical channel with feedback," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2016, pp. 295–299.
[19] O. Peled, O. Sabag, and H. H. Permuter, "Feedback capacity and coding for the $(0, k)$-RLL input-constrained BEC," *IEEE Trans. Inf. Theory*, vol. 65, no. 7, pp. 4097–4114, Mar. 2019.
[20] E. Shemuel, O. Sabag, and H. Permuter, "The feedback capacity of noisy output is the STate (NOST) channels," 2021, *arXiv:2107.07164*.
[21] O. Sabag, H. H. Permuter, and H. D. Pfister, "A single-letter upper bound on the feedback capacity of unifilar finite-state channels," *IEEE Trans. Inf. Theory*, vol. 63, no. 3, pp. 1392–1409, Mar. 2017.
[22] O. Sabag, B. Huleihel, and H. H. Permuter, "Graph-based encoders and their performance for finite-state channels with feedback," *IEEE Trans. Commun.*, vol. 68, no. 4, pp. 2106–2117, Apr. 2020.
[23] B. Huleihel, O. Sabag, H. H. Permuter, N. Kashyap, and S. Shamai, "Computable upper bounds on the capacity of finite-state channels," *IEEE Trans. Inf. Theory*, vol. 67, no. 9, pp. 5674–5692, Sep. 2021.
[24] H. Kim, Y. Jiang, S. Kannan, S. Oh, and P. Viswanath, "Deepcode: Feedback codes via deep learning," 2018, *arXiv:1807.00801*.
[25] E. Erdemir, P. L. Dragotti, and D. Gunduz, "Privacy-aware time-series data sharing with deep reinforcement learning," 2020.
[26] O. Sabag and H. H. Permuter, "The duality upper bound for unifilar finite-state channels with feedback," in *Proc. Int. Zurich Seminar Inf. Commun. (IZS)*, ETH Zurich, 2020, pp. 68–72.
[27] T. Berger and F. Bonomi, "Capacity and zero-error capacity of Ising channels," *IEEE Trans. Inf. Theory*, vol. 36, no. 1, pp. 173–180, Jan. 1990.
[28] E. Ising, "Beitrag zur theorie des ferromagnetismus," *Zeitschrift für Physik*, vol. 31, no. 1, pp. 253–258, 1925.
[29] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
[30] T. Kurutach, I. Clavera, Y. Duan, A. Tamar, and P. Abbeel, "Model-ensemble trust-region policy optimization," 2018, *arXiv:1802.10592*.
[31] M. Deisenroth and C. E. Rasmussen, "PILCO: A model-based and data-efficient approach to policy search," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, 2011, pp. 465–472.
[32] T. M. Cover, "Enumerative source encoding," *IEEE Trans. Inf. Theory*, vol. IT-19, no. 1, pp. 73–77, Jan. 1973.
[33] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 2, pp. 129–137, Mar. 1982.
[34] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[35] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 387–395.
[36] V. Mnih *et al.*, "Playing atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.
[37] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, Apr. 1970.
[38] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," 2015, *arXiv:1511.05952*.
[39] J. Schulman, N. Heess, T. Weber, and P. Abbeel, "Gradient estimation using stochastic computation graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–9.
[40] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 229–256, 1992.
[41] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, Jun. 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

Ziv Aharoni (Student Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical and computer engineering from the Ben-Gurion University of the Negev, Israel, in 2017 and 2020, respectively, where he is currently pursuing the Ph.D. degree in electrical and computer engineering. His research interests include information theory and machine learning.

Oron Sabag (Member, IEEE) was born in Tel Aviv-Yafo, Israel, in 1986. He received the B.Sc. *(cum laude)*, M.Sc. *(summa cum laude)*, and Ph.D. degrees in electrical and computer engineering from the Ben-Gurion University of the Negev, Israel, in 2013, 2016, and 2019, respectively.

He is currently a Post-Doctoral Fellow with the Department of Electrical Engineering, California Institute of Technology (Caltech). His research interests include control theory, information theory, and reinforcement learning. He was a recipient of several awards, among them are the ISEF Postdoctoral Fellowship, the Lachish Fellowship, the ISIT-2017 Best Student Paper Award, the SPCOM-2016 Best Student Paper Award, the Feder Family Award for outstanding research in communications, and the Kaufman Award.

Haim H. Permuter (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees *(summa cum laude)* in electrical and computer engineering from the Ben-Gurion University of the Negev, Israel, in 1997 and 2003, respectively, and the Ph.D. degree in electrical engineering from Stanford University, California, in 2008. Between 1997 and 2004, he was an Officer at a research and development unit of the Israeli Defense Forces. Since 2009, he has been with the Department of Electrical and Computer Engineering, Ben-Gurion University of the Negev, where he is currently a Professor and the Luck-Hille Chair in electrical engineering and also serves as the Head for communication, cyber, and information track in his department. He was a recipient of several awards, among them are the Fullbright Fellowship, the Stanford Graduate Fellowship (SGF), Allon Fellowship, and the U.S.–Israel Binational Science Foundation Bergmann Memorial Award. He served on the Editorial Board of the IEEE TRANSACTIONS ON INFORMATION THEORY from 2013 to 2016.