Pragmatic Communication for Remote Control of Finite-State Markov Processes

Pietro Talli, Edoardo David Santi, Federico Chiariotti, Touraj Soleymani,
Federico Mason, Andrea Zanella, and Deniz Gündüz

***Abstract*—Pragmatic or goal-oriented communication can optimize communication decisions beyond the reliable transmission of data, instead aiming at directly affecting application performance with the minimum channel utilization. In this paper, we develop a general theoretical framework for the remote control of finite-state Markov processes, using pragmatic communication over a costly zero-delay communication channel. To that end, we model a cyber-physical system composed of an encoder, which observes and transmits the states of a process in real-time, and a decoder, which receives that information and controls the behavior of the process. The encoder and the decoder should cooperatively optimize the trade-off between the control performance (i.e., reward) and the communication cost (i.e., channel use). This scenario underscores a pragmatic (i.e., goal-oriented) communication problem, where the purpose is to convey only the data that is most valuable for the underlying task, taking into account the state of the decoder (hence, the pragmatic aspect). We investigate two different decision-making architectures: in pull-based remote control, the decoder is the only decision-maker, while in push-based remote control, the encoder and the decoder constitute two independent decision-makers, leading to a multi-agent scenario. We propose three algorithms to optimize our system (i.e., design the encoder and the decoder policies), discuss the optimality guarantees ofs the algorithms, and shed light on their computational complexity and fundamental limits.**

***Index Terms*—Effective communication, goal-oriented communication, cyber-physical systems, implicit information, Markov decision processes.**

I. Introduction

THE PARADIGM of cyber-physical systems (CPSs) has recently enabled the creation of complex system architectures that tightly integrate communication, computation, and control [3], [4]. CPSs often exhibit a dynamic and distributed nature [5], which necessitates persistent status updating of local components to reflect changes in the environment [6]. This steady influx of real-time data empowers a CPS to swiftly adapt to evolving conditions, ensuring that control decisions are made based on the most pertinent and up-to-date information [7]. The performance of a CPS is, therefore, dramatically affected by both control and communication policies [8]. However, modeling, design, and optimization of CPSs in terms of control and communication policies can be quite challenging when control quality indices and communication constraints are simultaneously taken into account [9], [10].

Prior research has attempted to address these challenges by optimizing the age of information (AoI), a metric that captures the freshness of information in real-time networked systems [11]. Unfortunately, policies based on the AoI may result in abrupt changes in the process states being unreported for a relatively long time, as well as wasting resources for communication of irrelevant data [12]. This has led to the development of the notion of the value of information (VoI) in the context of CPSs [13], [14], which captures the significance of information and is intimately related to pragmatic communication[1] approaches [15]. The recent developments in communications focus on approaches that integrate semantic and pragmatic aspects [10] into the communications pipeline [16], which go beyond Shannon's strict source-channel separation and tailor communication decisions such as data compression and power scheduling to specific application goals [7], [17]. Nevertheless, the interactions between control and communication policies are non-trivial in complex scenarios, requiring further developments for solving intractable optimization problems in a cooperative way [18].

In the present work, we consider a finite-state Markovian physical process to be remotely monitored and, possibly, controlled, over a costly zero-delay communication channel. We model the system as a two-agent decentralized partially observable Markov decision process (Dec-POMDP), in which the first agent, named *encoder*, observes and transmits the states of the process in real time, and the second agent, named *decoder*, receives that information and controls the behavior of the process. In this system, on one hand, what the decoder receives depends on the communication decisions regarding transmitting the status updates over the channel. On the other hand, what the encoder observes in the future depends on the control decisions that can affect the process' state evolution. We investigate two different decision-making architectures. In the first one, called *pull-based* architecture, the decoder is the only decision-maker and is in charge of both communication and control decisions. In the second, called *push-based* archi-

Part of this work was funded by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, under-partnership on "Telecommunications of the Future" (PE0000001 - Program "RESTART"). Limited partial versions of this work were previously published / submitted as two conference papers [1], [2]. Talli and Edoardo David Santi contributed equally to this work. Federico Chiariotti and Touraj Soleymani also contributed equally to this work. Pietro Talli, Federico Mason, Federico Chiariotti, and Andrea Zanella are with the Department of Information Engineering, University of Padova, 35131 Padua, Italy (emails: {tallipietr, masonfed, chiariot, zanella}@dei.unipd.it). Edoardo David Santi, Touraj Soleymani, and Deniz Gündüz are with the Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2AZ, United Kingdom (e-mails: {edoardo.santi17, touraj, d.gunduz}@imperial.ac.uk). Touraj Soleymani is also with the City St George's School of Science and Technology, University of London, London EC1V 0HB, United Kingdom.

[1]In the literature, pragmatic communication is also known as goal-oriented, task-oriented, or effective communication.

tecture, the encoder and the decoder constitute two independent decision-makers, responsible for communication and control decisions, respectively. Intuitively, the former is simpler, while the latter can lead to higher performance due to an extended information structure.

Although the literature includes multiple solutions for the remote estimation and control of Markov processes, a unified theory for handling these tasks is still missing. Many works investigate heuristic or data-driven algorithms that, despite providing high performance under specific conditions, are not general enough and do not take into account the game theoretical limitations of multi-agent systems. In this work, we aim to fill the gaps in the literature, by designing a new and more general framework for the remote control of Markov processes, that can effectively represent several real-world problems and provide a theoretical grounding to pragmatic communication schemes. By doing so, we want to highlight and analytically model the subtle optimization challenges that characterize CPSs, especially in the case of limited bandwidth or other communication resources. One of these challenges is associated with the exploitation of *implicit information*, i.e., knowledge about the state of the process that the decoder can infer when the encoder does not transmit any data.

This paper builds upon our preliminary findings reported in [1], [2]: in particular, we analyze the key features of pull-based and push-based architectures in [1], while we investigate the benefits of implicit information in the push-based architecture in [2]. The current work significantly improve these results by integrating our techniques in a unified model, highlighting the challenges associated with the joint design of control and communication policies, and formally proving the correctness and the limitations of the proposed schemes. This generalization extends to broader theoretical and numerical results, as well as computational complexity considerations. Our ultimate goal is to develop a theoretical framework for solving the problem of the remote control of finite-state Markov processes over costly zero-delay communication channels either exactly or approximately, and delineating the fundamental properties of possible solutions. In short, the main contributions of this paper are as follows:

- *(i)* We introduce a CPS model that can effectively represent pragmatic communication in remote control tasks, and can readily be specialized to remote estimation tasks. This model, in general, consists of an encoder and a decoder, which can exchange information and should cooperatively accomplish a goal. Based on this model, we investigate two decision-making architectures, namely, pull-based and push-based;
- *(ii)* In the pull-based setting, we propose the modified policy iteration (MPI) scheme, proving that it can reach an ε-optimal solution in polynomial time. However, we also prove that the pull-based setting represents a restricted version of the problem, with inherent performance limits;
- *(iii)* In the push-based setting, we propose two schemes, namely, alternating policy iteration (API), which converges to a locally ε-optimal solution in polynomial time, and joint policy optimization (JPO), which can achieve a globally ε-optimal solution but requires exponential computation time. We discuss formally the computational complexity and the fundamental limits of these schemes;
- *(iv)* We validate our theoretical results by means of experiments for remote estimation and control tasks, comparing the performance of the proposed schemes in a simulated environment. Our results, in particular, show that practical polynomial-time algorithms seem to be robust in the remote control tasks, while the optimality gap becomes much wider in the remote estimation tasks.

The rest of the paper is organized as follows. We discuss the related work on remote control and remote estimation as well as on pragmatic communication in Sec. II. Then, we introduce the system model and formulate the problem of interest in Sec. III. We propose our algorithms in Sec. IV, and provide our theoretical results in Sec. V. Then, we present our numerical analysis and simulation results in Sec. VI. Finally, we provide our concluding remarks in Sec. VII.

II. Related Work

In feedback control, the effectiveness of control actions directly depends on the quality of state estimates. This implies that, in order to tackle a remote control problem, one should be able to address first the corresponding remote estimation problem, as the most foundational task. The design of optimal policies for remote estimation and control of dynamical processes over communication channels has frequently been addressed in the literature. Notably, the work in [19] analyzes the best strategies for lossy transmissions without any communication cost in a finite time horizon. The optimality of these strategies in the case of an infinite time horizon is proven in [20], along with the ε-optimality of finite-memory quantizers. The same work shows that deterministic policies lead to optimal encoding solutions, while the work in [21] adopts a reinforcement learning (RL) approach to address the target problem. These works generally deal with lossy encoding, i.e., with the decision over how to encode system states with a fixed packet length. The general properties of encoding policies for remote estimation are discussed in [22], which shows that the trade-off curve between the estimation error and the transmission rate can be modeled by a piece-wise linear convex decreasing function. However, these results only hold under specific conditions, i.e., for Markov processes with a rigid structure that simplifies the problem.

The problem of the remote estimation of scalar continuous-state Markov processes was addressed in [23], [24], where the optimal policies are also derived. A more recent study in [25] investigates the same problem in the presence of delay and packet loss, shedding light on the structure of the optimal policies and the role of implicit information. However, the findings of these past works cannot be generalized to finite-state Markov processes. In [26], the authors introduce the channel noise and design an encoding policy for monitoring a

two-state Markov chain, taking into account the importance of each state in terms of the actions to be taken by the agent. The same framework is extended to N-state Markov chains in [27], which investigates the optimal configuration for a sampling and communication policy to minimize the reconstruction error under different constraints. Other works [28]–[30] propose solutions for single-agent Markov processes in which an agent needs to pay a fixed price to either observe the current or the next state. This model suits pull-based settings, where a receiver must request state updates from a sensor over a resource-constrained channel.

Finally, several works have proposed learning-based pragmatic communication solutions for remote control [31], showing that a significant performance improvement is obtained by the joint training of sensors and agents (i.e., encoders and decoders) [32], [33]. These works usually exploit the multi-agent reinforcement learning (MARL) paradigm, modeling the channel as an information bottleneck [34] and considering the mutual information between agent beliefs [35] as part of the system state. However, MARL solutions provide no theoretical guarantees hard to adapt to scenarios different from those in which they were initially trained.

Indeed, there is a significant gap between the theoretical results on remote estimation and learning-based pragmatic communication schemes: while the former mostly give the optimal encoding in specific scenarios, the latter are often presented without any optimality guarantees, limiting their generality and trustworthiness. Our work contributes to filling this gap by providing formal theoretical results for the general case of finite-state Markov processes with two agents, which can exchange data across a costly communication channel with the objective of either monitoring or controlling the process. With respect to the state of the art, our study unifies recent results about the performance of pull-based and push-based pragmatic communication approaches, extending previous analyses by deriving computational complexity and optimality bounds.

III. Problem Formulation and Modeling

Let us consider a CPS model with two agents: an *encoder* and a *decoder*. At each time step, the encoder observes the state of a discrete-time Markov process and should transmit this information to the decoder, whose task is to control the behavior of the process. Our objective is to design the encoder's and the decoder's strategies in order to optimize the trade-off between the control performance (i.e., reward) and the communication cost (i.e., channel use) over an infinite time horizon. This problem can be expressed mathematically as a two-agent Dec-POMDP [36] characterized by the tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{C}, \mathbf{P}, o, r, \gamma \rangle$[2], where

- $\mathcal{S}$ is the discrete and finite set of states of the underlying Markov chain; the state at time t is denoted by s_t;
- $\mathcal{O} = \mathcal{S} \cup \{\chi\}$ is the discrete and finite set of possible observations of the decoder, where the symbol χ represents the absence of transmission; the observation of the decoder at time t is denoted by o_t. Note that the observation set of the encoder is $\mathcal{S}$, as the encoder can directly observe the system state at each time;
- $\mathcal{A}$ is the discrete and finite set of possible actions of the decoder; the control action at time t is denoted by a_t;
- $\mathcal{C} = \{0, 1\}$ is the set of possible encoder actions (actions 0 and 1 correspond to no transmission and transmission of the current state, respectively), and the communication action at time t is denoted by c_t;
- $\mathbf{P} \in [0,1]^{|\mathcal{S}|\times|\mathcal{A}|\times|\mathcal{S}|}$ is the transition matrix, whose entry (s, a, s') represents the probability of moving to state s' when performing action $a \in \mathcal{A}$ in state s;
- $o : \mathcal{S} \times \mathcal{C} \mapsto \mathcal{O}$ is the decoder observation function[3]
$$o_t = o(s_t, c_t) = \begin{cases} s_t, & \text{if } c_t = 1, \\ \chi, & \text{if } c_t = 0; \end{cases} \tag{1}$$
- $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$ is the reward function, and the reward at time t is given by $r_t = r_{s_t, s_{t+1}}(a_t)$;
- $\gamma \in [0, 1)$ is the exponential discount factor.

At each time step, if the encoder transmits a status update, the decoder will have perfect knowledge about the current state of the process. Otherwise, the decoder will have imperfect knowledge, and must rely on its own history of past observations and actions to control the system.

We represent encoding and decoding policies by $\pi_{\mathrm{e},t}$ and $\pi_{\mathrm{d},t}$, respectively. These policies are causal, meaning that at each time step they depend only on the knowledge of the encoder and the decoder at that time step. To jointly optimize performance under a communication constraint, one needs to solve the following problem:

$$\text{maximize} \;\; \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right], \;\; \text{subject to} \;\; \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t c_t\right] \leq C, \tag{2}$$

where C is value specifying a constraint on the cumulative channel use. The above problem is a constrained Dec-POMDP, which may require stochastic policies. In this work, we reformulate (2) as an unconstrained Dec-POMDP, allowing us to safely restrict our attention to deterministic policies [37]:

$$\text{maximize} \;\; \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t (r_t - \beta c_t)\right], \tag{3}$$

where $\beta \in \mathbb{R}^+$ is a weighting coefficient. Note that β is in fact the price that the system has to pay for the *reliable communication* of a message from the encoder to the decoder.

Our first result is the following theorem, which dramatically simplifies the structures of the encoding and decoding policies.

Theorem 1. *Without any loss of optimality, at each time t, the knowledge of the encoder can be described by $\langle s_t, \Delta_t, s_{t-\Delta_t} \rangle$, and that of the decoder by $\langle \Delta_t, s_{t-\Delta_t} \rangle$, where Δ_t is the time elapsed since the last transmission.*

[2] In this work, we consider an infinite-horizon discounted formulation, but the theoretical and practical considerations can be adapted to related problems which aim at maximizing the average reward over a finite or infinite horizon.

[3] In the Dec-POMDP literature, this commonly depends on the last control action a as well. As a is irrelevant in our model, we drop it for brevity's sake.

Proof: Note that the state of a Markov decision process (MDP) is a sufficient statistic for the decision made by its underlying agent. In our problem, for any fixed decoder policy, at each time t, the encoder experiences a Markovian environment whose state is $\langle s_t, o_{0:t-1}\rangle$. Since this process is Markovian, the state $\langle s_t, o_{0:t-1}\rangle$ embodies all the information from the past evolution of the process. Moreover, the observation o_t at the decoder depends on the encoder action c_t that, in turn, only depends on $\langle s_t, o_{0:t-1}\rangle$. In addition, the current belief of the decoder is only a function of its history of observations and control actions since the last renewal instant, i.e., the last successful transmission. Note that the encoder provides either no explicit information to the decoder when $c_t = 0$, or the exact state s_t when $c_t = 1$. Therefore, we deduce that the tuple $\langle \Delta_t, s_{t-\Delta_t}\rangle$ is a sufficient statistic for the decoder. Finally, following the same logic, we also deduce that the tuple $\langle s_t, \Delta_t, s_{t-\Delta_t}\rangle$ is a sufficient statistic for the encoder. ■

By Theorem 1, the decoder can then use $\langle \Delta_t, s_{t-\Delta_t}\rangle$ as a Markovian state. Hence, the decoder selects an action $a_t \in \mathcal{A}$ according to a control policy $\boldsymbol{\pi}_{\mathrm{d}} : \mathcal{S} \times \{0, \dots, T_{\max}\} \mapsto \mathcal{A}$ (and also the following communication action $c_{t+1} \in \mathcal{C}$ in a pull-based system), gaining an instantaneous reward of $r_t = r_{s_t, s_{t+1}}(a_t)$, whose long-term maximization represents the system goal. To maintain a finite state space, we assume that the system allows a maximum inter-transmission time of $T_{\max}$, after which a new communication is always triggered. In particular, we can set $T_{\max}$ to a value large enough to make the boundary approximation error arbitrarily small. We can also avoid considering a boundary when shifting to a belief-based problem formulation, as we will discuss later.

There can be two possible decision-making architectures for the optimization problem in (3): *pull-based* and *push-based* architectures. In the pull-based setting, the decoder is the only decision maker and has to decide whether to request status updates and to select control actions, reducing the system to a single-agent problem. However, in the push-based setting, the encoder and the decoder are two independent decision-makers and must adapt to each other. Both settings have several practical applications, and the choice between the two also depends on architectural, energy, and cost considerations.

We highlight that our results can be easily extended to more realistic communication channels with time delay and a fixed packet loss rate, as long as the information structure is preserved, i.e., in these cases, the delay should be known to both nodes so that the encoder can compute the decoder's belief; an acknowledgment mechanism should also be implemented, so that the encoder knows the outcome of its transmissions.

A. Pull-Based Remote Control

The pull-based setting describes a restricted single-agent problem in which the system is entirely controlled by the decoder, and can be mapped to a problem class known as action-contingent noiselessly observable MDP (ACNO-MDP) [30], in which the state s_t is available to the agent through a (costly) sensing action corresponding to a communication request to the encoder, which has perfect state information and can transmit it on demand. If we fix the communication policy, the resulting system is a standard POMDP and can be solved near-optimally [38], [39].

Practically, each time step can be organized into two substeps, i.e., a communication step followed by a control step. During the first substep, the decoder must decide whether to request a new update from the transmitter, according to a policy $\boldsymbol{\tau} : \mathcal{S} \times \{0, \dots, T_{\max}\} \mapsto \mathcal{C}$. Following Theorem 1, this is equivalent to computing the full *a priori* belief distribution given the history of the last Δ_t actions $\mathbf{a}_{t-\Delta_t:t-1} = (a_{t-\Delta_t}, a_{t-\Delta_t+1}, \dots, a_{t-1})$, i.e.,

$$\boldsymbol{\omega}^{\mathbf{a}_{t-\Delta_t:t-1}}_{\langle \Delta_t, s_{t-\Delta_t}\rangle} = \left(\prod_{\ell=t-\Delta_t}^{t} \mathbf{P}^{a_\ell} \right)^{\intercal} \mathbf{u}_{s_{t-\Delta_t}}, \tag{4}$$

where $\mathbf{u}_s$ is a one-hot column vector whose elements are all 0, except for the one corresponding to s, which is equal to 1, $\mathbf{P}^a$ is the transition matrix when taking action a, and $(.)^{\intercal}$ denotes the transpose operation. As the transmission process relies only on the decoder's estimations of the system state, the *a posteriori* belief distribution is identical to the *a priori* one. During the second substep, the decoder selects a control action according to policy $\boldsymbol{\pi}_{\mathrm{d}} : \mathcal{S} \times \{0, \dots, T_{\max}\} \mapsto \mathcal{A}$. Between the two substeps, the decoder's state will change from $\langle \Delta_t, s_{t-\Delta_t}\rangle$ to $\langle 0, s_t\rangle$ if an update is requested, i.e., $c_t = 1$, resetting the belief to a singleton, while $\boldsymbol{\omega}_t$ will be updated in a Bayesian manner using the transition matrix $\mathbf{P}$ otherwise.

B. Push-Based Remote Control

The push-based setting corresponds to a two-agent problem, in which the encoder acts first, and the knowledge of the decoder is affected by the encoder's update. Practically, at each time step t, the encoder uses policy $\boldsymbol{\pi}_{\mathrm{e}}$ to decide whether to send a status update, while the decoder only determines control actions. Therefore, the communication action is determined by the policy $\boldsymbol{\pi}_{\mathrm{e}} : \mathcal{S} \times \{0, T_{\max}\} \times \mathcal{S} \mapsto \mathcal{C}$. This two-agent problem is a fully cooperative Markov game with asymmetric information, as the encoder knows everything about the system, while the decoder does not. It is also an exact potential game [40], as the reward of the two agents is exactly the same.

As the channel is zero-delay and lossless, the decoder knows that the encoder has decided not to transmit in the last Δ_t states $s_{t-\Delta_t+1}, \dots, s_t$. Accordingly, the *a posteriori* belief distribution can be expressed recursively for $\Delta_t \geq 1$ as:

$$\boldsymbol{\omega}^{\mathbf{a}_{t-\Delta_t:t-1}, \boldsymbol{\pi}_{\mathrm{e}}}_{\langle \Delta_t, s_{t-\Delta_t}\rangle} = \frac{\mathrm{D}(\mathbf{1} - \mathbf{c}(\Delta_t, s_{t-\Delta_t}))(\mathbf{P}^{a_{t-1}})^{\intercal} \boldsymbol{\omega}^{\mathbf{a}_{t-\Delta_t:t-2}, \boldsymbol{\pi}_{\mathrm{e}}}_{\langle \Delta_t - 1, s_{t-\Delta_t}\rangle}}{(\mathbf{1} - \mathbf{c}(\Delta_t, s_{t-\Delta_t}))\,(\mathbf{P}^{a_{t-1}})^{\intercal} \boldsymbol{\omega}^{\mathbf{a}_{t-\Delta_t:t-2}, \boldsymbol{\pi}_{\mathrm{e}}}_{\langle \Delta_t - 1, s_{t-\Delta_t}\rangle}}, \tag{5}$$

where function $\mathrm{D}(\mathbf{x})$ returns a diagonal matrix whose i-th diagonal entry is the i-th element of vector $\mathbf{x}$, and $\mathbf{c}(\Delta_t, s_{t-\Delta_t})$ is a row vector of length $|\mathcal{S}|$ whose s-th entry is $\pi_{\mathrm{e}}(s, \Delta_t, s_{t-\Delta_t})$.

While the belief update in (4) is naive, as it does not consider the encoder's policy, the push-based scenario presents higher complexity because of the existence of *implicit information*. By not transmitting, the encoder gives the decoder a single bit of

information, which may prove extremely useful for the decoder strategy. Consequently, implicit information intertwines the design of both the encoder and the decoder. Neglecting this aspect *a priori* decouples the design problem, but at the cost of a significant optimality gap (for more details, see [2]). The pull-based setting also does not allow for the use of implicit information, as it restricts the available information to the knowledge of the decoder, making the problem easier to solve but incurring the same performance cost.

C. Special Case: Remote Estimation

The proposed architectures can be applied to remote estimation problems where the decoder's task is not controlling the physical process, but rather monitoring it, i.e., estimating the correct state s_t at each step t. This is a special case of the general Dec-POMDP problem, in which $\mathcal{A} = \mathcal{S}$ and $\mathbf{P}^a = \mathbf{P}^{a'} \, \forall \, a, a' \in \mathcal{A}$, and can be addressed for both the pull-based and push-based architectures. Specifically, we model this scenario by using the $0-1$ reward function: $r_{s_t,s_{t+1}}(a_t) = \delta_{s_t,a_t}$, where $\delta_{m,n}$ is the Kronecker delta function, which is equal to 1 if the arguments are equal and 0 otherwise.

We observe that the results obtained for this settings hold for any reward function that depends directly on the current state and selected action, i.e., $r_{s_t,s_{t+1}}(a_t) = f(s_t, a_t)$. Indeed, the remote estimation problem can be seen as a remote contextual bandit [41], and solutions to this problem reveal properties uncovered by previous work on the subject [42], [43].

IV. Proposed Algorithms

In the following, we propose three algorithms for optimizing our CPS model and discuss the optimality properties of each solution. We first focus on the pull-based setting, and extend the classical policy iteration approach by taking into account the *a priori* belief of the decoder. We propose two algorithms for the push-based setting: the first alternately optimizes the encoding and decoding policies, while the second is based on an intricate transformation into a single-agent POMDP.

A. Pull-Based Modified Policy Iteration Scheme

To solve the ACNO-MDP optimally, we design an algorithm, called MPI, which is a modified version of policy iteration considering the belief of the decoder over the state space. Note that, by Theorem 1, the knowledge of the decoder is entirely described by the tuple $\langle \Delta_t, s_{t-\Delta_t} \rangle$, which can be used as a modified state. Considering the division into phases presented in Sec. III-A, we thus define the following policies for the decoder:

- $\boldsymbol{\pi}_\mathrm{d}$: the control policy, which maps each state $\langle \Delta_t, s_{t-\Delta_t} \rangle \in \{0, \dots, T_{\max}\} \times \mathcal{S}$ to a control action $\pi_\mathrm{d}(\Delta_t, s_{t-\Delta_t}) \in \mathcal{A}$;
- $\boldsymbol{\tau}$: the communication policy, mapping state $s_{t-\Delta_t} \in \mathcal{S}$ to a delay $\tau(s_{t-\Delta_t}) \in \{0, \dots, T_{\max}\}$ until the next update.

Algorithm 1 Pull-Based Modified Policy Iteration (MPI)

Require: $\mathbf{P}, r, \beta$
Initialize $\mathbf{V}_\mathrm{d}(\Delta_t, s_{t-\Delta_t}) \leftarrow 0$, randomize $\boldsymbol{\pi}_\mathrm{d}(\Delta_t, s_{t-\Delta_t})$, $\boldsymbol{\tau}$
while true **do**
  for $(\Delta_t, s_{t-\Delta_t}) \in \mathcal{S} \times \{0, ..., T_{\max}\}$ **do**
    $V'_\mathrm{d}(\Delta_t, s_{t-\Delta_t}) \leftarrow$Update using (6)
  $\mathbf{V}_\mathrm{d} \leftarrow \mathbf{V}'_\mathrm{d}$ ▷ Value update step
  for $\Delta_t \in \{0, \dots, T_{\max}\}$ **do** ▷ Iterative step
    for $s_{t-\Delta_t} \in \mathcal{S}$ **do**
      $\pi'_\mathrm{d}(\Delta_t, s_{t-\Delta_t}) \leftarrow$Update using (7)
      $\tau'(s) \leftarrow$Update using (9)
  if $\boldsymbol{\pi}'_\mathrm{d} = \boldsymbol{\pi}_\mathrm{d} \wedge \boldsymbol{\tau}' = \boldsymbol{\tau}$ **then**
    return $\boldsymbol{\pi}_\mathrm{d}, \boldsymbol{\tau}$ ▷ Convergence
  else
    $\boldsymbol{\pi}_\mathrm{d}, \boldsymbol{\tau} \leftarrow \boldsymbol{\pi}'_\mathrm{d}, \boldsymbol{\tau}'$ ▷ Policy improvement step

The above model for the communication policy $\boldsymbol{\tau}$ is more compact than a mapping from $\langle \Delta_t, s_{t-\Delta_t} \rangle$ to a binary pull decision, as the decoder will pull only after $\tau(s_{t-\Delta_t})$ steps.

Since the state transitions until a pull decision are deterministic, i.e., the decoder's state always goes from $\langle \Delta_t, s_{t-\Delta_t} \rangle$ to $\langle \Delta_t + 1, s_{t-\Delta_t} \rangle$ unless it requests an update, the action vector $\mathbf{a}_{t-\Delta_t:t-1}(s_{t-\Delta_t}|\boldsymbol{\pi}_\mathrm{d}) \in \mathcal{A}^{\Delta_t}$ can be determined in advance. We can then compute the belief $\boldsymbol{\omega}^{\mathbf{a}_{t-\Delta_t:t-1}(s_{t-\Delta_t}|\boldsymbol{\pi}_\mathrm{d})}_{\langle \Delta_t, s_{t-\Delta_t} \rangle}$ over the state space using (5), and the value function $V_\mathrm{d}(\Delta_t, s_{t-\Delta_t})$ of the control policy $\boldsymbol{\pi}_\mathrm{d}$ using the Bellman equation:

$$\begin{aligned} V_\mathrm{d}(\Delta_t, s_{t-\Delta_t}) &= \sum_{s_t, s_{t+1} \in \mathcal{S}} \omega^{\mathbf{a}_{t-\Delta_t:t-1}(s_{t-\Delta_t}|\boldsymbol{\pi}_\mathrm{d})}_{\langle \Delta_t, s_{t-\Delta_t} \rangle}(s_t) P^{\pi_\mathrm{d}(\Delta_t, s_{t-\Delta_t})}_{s_t, s_{t+1}} \\ &\times \big[r_{s_t,s_{t+1}}(\pi_\mathrm{d}(\Delta_t, s_{t-\Delta_t})) + \gamma \delta_{\tau(s_{t-\Delta_t}), \Delta_t+1} V_\mathrm{d}(1, s_{t+1}) \\ &+ \gamma \left(1 - \delta_{\tau(s_{t-\Delta_t}), \Delta_t+1}\right) \left(V_\mathrm{d}(\Delta_t + 1, s_{t-\Delta_t}) - \beta\right) \big]. \end{aligned} \tag{6}$$

Since the belief depends on previous actions, we start from $\Delta_t = 0$ and iterate, using previously updated policy steps to compute the value function for larger values of Δ_t:

$$\begin{aligned} \pi'_\mathrm{d}(\Delta_t, s_{t-\Delta_t}) &= \arg\max_{a \in \mathcal{A}} \sum_{s_t, s_{t+1} \in \mathcal{S}} \omega^{\mathbf{a}_{t-\Delta_t:t-1}(s_{t-\Delta_t}|\boldsymbol{\pi}'_\mathrm{d})}_{\langle \Delta_t, s_{t-\Delta_t} \rangle}(s_t) \\ &\times P^a_{s_t, s_{t+1}} \Big[r(s_t, a, s_{t+1}) + \gamma \delta_{\tau(s_{t-\Delta_t}), \Delta_t+1} V_\mathrm{d}(0, s_{t+1}) \\ &+ \gamma \left(1 - \delta_{\tau(s_{t-\Delta_t}), \Delta_t+1}\right) \left(V_\mathrm{d}(\Delta_t + 1, s_{t-\Delta_t}) - \beta\right) \Big]. \end{aligned} \tag{7}$$

To perform the communication policy improvement step, we modified the standard update rule to consider the evolution of the Markov chain until the state is sampled again. Given a state sequence $\mathbf{s}_{t+1:t+m} \in \mathcal{S}^m$, after observing state $s_{t-\Delta_t}$, the belief $\omega^{\boldsymbol{\pi}_\mathrm{d}}_{\langle \Delta_t, s_{t-\Delta_t} \rangle}(\mathbf{s}_{t+1:t+m})$ follows from (5):

$$\begin{aligned} \omega^{\boldsymbol{\pi}_\mathrm{d}}_{\langle \Delta_t, s_{t-\Delta_t} \rangle}(\mathbf{s}_{t+1:t+m}) &= \omega^{\mathbf{a}_{t-\Delta_t:t}(s_{t-\Delta_t}|\boldsymbol{\pi}_\mathrm{d})}_{\langle \Delta_t, s_{t-\Delta_t} \rangle}(s_{t+1}) \\ &\times \prod_{\ell=1}^{m-1} (1 - \delta_{\tau(s_{t-\Delta_t}), \Delta_t+\ell}) P^{\pi_\mathrm{d}(\Delta_t+\ell, s_{t-\Delta_t})}_{s_{t+\ell}, s_{t+\ell+1}}. \end{aligned} \tag{8}$$

Using the whole sequence of intermediate steps in the update equations is essential to take into account that each state $\langle \Delta_t, s_{t-\Delta_t} \rangle, \forall \, \Delta_t > 0$ is non-Markovian, as the belief distribution depends on the action sequence taken since the last communication. Hence, the communication policy can be

Algorithm 2 Push-Based Alternating Policy Iteration (API)

Require: $\mathbf{P}, r, \beta, \boldsymbol{\pi}_{\mathrm{e}}^{(0)}$
Initialize $\boldsymbol{\pi}_{\mathrm{e}} \leftarrow \boldsymbol{\pi}_{\mathrm{e}}^{(0)}$, randomize $\boldsymbol{\pi}_{\mathrm{d}}$
while true **do**
  $\boldsymbol{\pi}'_{\mathrm{d}} \leftarrow$ ControlPolicyIteration($\mathbf{P}, r, \beta, \boldsymbol{\pi}_{\mathrm{e}}$)
  $\boldsymbol{\pi}'_{\mathrm{e}} \leftarrow$ CommunicationPolicyIteration($\mathbf{P}, r, \beta, \boldsymbol{\pi}'_{\mathrm{d}}$)
  if $\boldsymbol{\pi}'_{\mathrm{d}} = \boldsymbol{\pi}_{\mathrm{d}} \wedge \boldsymbol{\pi}'_{\mathrm{e}} = \boldsymbol{\pi}_{\mathrm{e}}$ **then**
    return $\boldsymbol{\pi}_{\mathrm{d}}, \boldsymbol{\pi}_{\mathrm{e}}$ ▷ Convergence
  else
    $\boldsymbol{\pi}_{\mathrm{d}}, \boldsymbol{\pi}_{\mathrm{e}} \leftarrow \boldsymbol{\pi}'_{\mathrm{d}}, \boldsymbol{\pi}'_{\mathrm{e}}$ ▷ Best response

updated as:

$$\tau'(s_t) = \arg\max_{m \in \{0,\dots,T_{\max}\}} \sum_{\mathbf{s}_{t+1:t+m} \in \mathcal{S}^m} \omega^{\boldsymbol{\pi}_{\mathrm{d}}}_{\langle 0, s_t\rangle}(\mathbf{s}_{t+1:t+m}) \Big[\gamma^m V_{\mathrm{d}}(0, s_{t+m}) - \gamma^m \beta + r_{s_t, s_{t+1}}(\pi_{\mathrm{d}}(0, s_t)) + \sum_{\ell=1}^{m-1} \gamma^\ell r_{s_{t+\ell}, s_{t+\ell+1}}(\pi_{\mathrm{d}}(\ell, s_t))\Big]. \tag{9}$$

Note that MPI provides an optimal solution for the pull-based setting, which however represents a restricted version of the general Dec-POMDP problem presented in this paper, as we will discuss in Sec. V. The pseudocode for MPI is given in Alg. 1.

B. Push-Based Alternating Policy Iteration Scheme

For the push-based setting, we first design a computationally light iterative algorithm, called API. In this case, the training is organized into subsequent rounds, each split into two phases. During the first phase of each round, the encoding policy is fixed, and the decoding policy is optimized, while during the second phase, the decoding policy is fixed and the encoding policy is optimized. The procedure is repeated until both policies converge, i.e., each agent follows an optimal policy with respect to the other one.

Note that, during each phase, one agent's policy is static, so that the other agent experiences a Markovian environment and standard policy iteration can be applied. In particular, the decoder computes the full belief in (5) to obtain the expected reward for each tuple $\langle \Delta_t, s_{t-\Delta_t}\rangle$, while the problem on the encoder side is a fully observable MDP. As we will show in Sec. V, this algorithm always converges to a Nash equilibrium (NE) in a finite number of steps. The pseudocode for API is given in Alg. 2.

C. Push-Based Joint Policy Optimization Scheme

We now design a more complex algorithm, called JPO. In this case, we transform the original problem into a single-agent POMDP, where the decision maker has full knowledge of the decoder's status and its decisions represent the actions taken by the encoder and the decoder in succession. This POMDP can be solved numerically and near-optimally using a point-based POMDP solver, such as successive approximations of the reachable space under optimal policies (SARSOP) [44], which is used in this paper. We are then able to exploit the piecewise linear convex structure of the value function, which allows us to have an anytime bound on the optimality of the result. As we will show in Sec. V, this algorithm finds an optimal solution for the general Dec-POMDP problem presented in this paper, but requires an exponentially increasing computation time. The pseudocode for JPO is given in Alg. 3.

Algorithm 3 Push-Based Joint Policy Optimization (JPO)

Require: $\langle \mathcal{S}, \mathcal{O}, \hat{\mathcal{A}}, \hat{\mathbf{P}}, \hat{\boldsymbol{O}}, \hat{\boldsymbol{r}}, \gamma\rangle, \Omega_0, \epsilon$ ▷ Ω_0 is the set of possible initial decoder beliefs, given the initial state distribution
Initialize Γ, $\underline{V}$ using Blind Lower Bound [45] ▷ Γ is the α-vector representation of the lower bound
Initialize $\overline{V}$ using Fast Informed Bound [46]
while $\underline{V}(\omega_0) + \varepsilon < \overline{V}(\omega_0),\ for\ any\ \omega_0 \in \Omega$ **do**
  Insert $w_0 \in \Omega_0$, s.t. $\underline{V}(\omega_0) + \varepsilon < \overline{V}(\omega_0)$ as the root of tree $T_{\mathcal{R}}$.
  Sample($T_{\mathcal{R}}, \Gamma$)
  For a subset of beliefs ω from $T_{\mathcal{R}}$,
  Backup($T_{\mathcal{R}}, \Gamma, \omega$)
  Prune($T_{\mathcal{R}}, \Gamma$)
$\boldsymbol{\pi}_{\mathrm{e},0} \leftarrow$ Update using (13)

More specifically, we define the single-agent POMDP $\hat{\mathcal{M}} = \langle \mathcal{S}, \mathcal{O}, \hat{\mathcal{A}}, \hat{\mathbf{P}}, \hat{\boldsymbol{O}}, \hat{\boldsymbol{r}}, \gamma\rangle$, where $\mathcal{S}$ is the same state space as in the underlying MDP; $\mathcal{O} = \mathcal{S} \cup \{\chi\}$ is the observation set; $\hat{\mathcal{A}} = \mathcal{A} \times \mathcal{C}^{|\mathcal{S}|}$ is a modified action space, where each action is represented by a tuple $\langle a, \boldsymbol{c}\rangle$, where $a \in \mathcal{A}$ represents the decoder's action and $\boldsymbol{c}$ is a vector with $|\mathcal{S}|$ elements, each of which represents the encoder's action for each possible state; $\hat{P} = \{\hat{P}^{\hat{a}} | \hat{a} \in \hat{\mathcal{A}}\}$ is the set of transition matrices $\hat{P}^{\hat{a}}_{s,s'} = \Pr(s_{t+1} = s' | s_t = s, \hat{a}_t = \hat{a} = \langle a, \boldsymbol{c}\rangle) = P^a_{s_t, s_{t+1}}$; $\hat{\boldsymbol{r}} : \mathcal{S} \times \hat{\mathcal{A}} \mapsto \mathbb{R}$ is the reward function defined as

$$\hat{r}(s, \langle a, \boldsymbol{c}\rangle) = \sum_{s' \in \mathcal{S}} P^a_{s,s'} \left(r(s, a, s') - \gamma\beta \boldsymbol{c}_s\right); \tag{10}$$

and $\hat{o} : \mathcal{S} \times \mathcal{C}^{|\mathcal{S}|} \mapsto \mathcal{O}$ is the observation function mapping communication actions and states to decoder observations

$$\hat{o}_t = \hat{o}(s_t, \boldsymbol{c}_t) = \begin{cases} \chi, & c_{t,s_t} = 0, \\ s_t, & c_{t,s_t} = 1. \end{cases} \tag{11}$$

Note that we can consider the optimal policy as a function of decoder's belief, instead of the observation history. The algorithm then identifies the optimal policy as

$$\hat{\boldsymbol{\pi}}^*_{\text{joint}} = \arg\max_{\hat{\boldsymbol{\pi}}: \Omega \mapsto \hat{A}} \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\boldsymbol{\omega}_0, \hat{\boldsymbol{\pi}}} \left[\hat{r}(s_t, \hat{\boldsymbol{\pi}}(\boldsymbol{\omega}_t))\right], \tag{12}$$

where Ω is the probability simplex over $\mathcal{S}$, i.e., the set of possible belief distributions, and $\boldsymbol{\omega}_0$ is the decoder's initial belief. The joint policy can then be split into policies $\boldsymbol{\pi}^*_{\mathrm{d},t}(\boldsymbol{\omega}_t) = a_t$ and $\boldsymbol{\pi}^*_{\mathrm{e},t}(\boldsymbol{\omega}_{t-1}, s_t) = c_t$, which represent deterministic mappings to actions that can be implemented in a distributed fashion. We can represent the optimal encoding policy as a function of the current state, the last transmitted state, and number of time steps since the last transmission, and the optimal control policy as function of the last two. In addition, the encoder policy in the first step, $\boldsymbol{\pi}_{\mathrm{e},0} : \mathcal{S} \mapsto \mathcal{C}$, is computed separately as

$$\boldsymbol{\pi}^*_{\mathrm{e},0} = \arg\max_{\boldsymbol{\pi}_0 : \mathcal{S} \mapsto \mathcal{C}} \sum_{s_0 \in \mathcal{S}} P_0(s) \left[\underline{V}(\omega_0(\mathbf{P}_0, \pi_0(s_0))) - \beta \pi_0(s_0)\right], \tag{13}$$

where $\mathbf{P}_0$ is the initial state distribution and $\underline{V}$ is the lower

bound of the value function given after using the numerical algorithm to solve (12).

V. Theoretical Results and Computational Limits

In this section, we discuss the advantages and drawbacks of the proposed algorithms from a theoretical point of view. Specifically, we first compare the pull-based and push-based architectures, showing that the optimal push-based solution is always better than the optimal pull-based one. Then, we analyze the MPI and API schemes. Finally, we consider the JPO scheme, proving that it can reach an ε-optimal solution, although with a higher computational cost.

A. Optimality Analysis

Let $\boldsymbol{\pi} = \langle \boldsymbol{\tau}, \boldsymbol{\pi}_{\mathrm{d}} \rangle$ be a joint policy in the pull-based setting and $J^{\boldsymbol{\pi}}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, \boldsymbol{\pi}\right]$ be the long-term reward (without considering the communication cost) starting from state s. We can give a compact definition of the performance of the joint policy $\boldsymbol{\pi} = \langle \boldsymbol{\tau}, \boldsymbol{\pi}_{\mathrm{d}} \rangle$:

$$R_{\beta}^{\boldsymbol{\pi}} = \sum_{s \in \mathcal{S}} \xi(s) \left(J^{\boldsymbol{\pi}}(s) - \mathbb{E}\left[\sum_{n=0}^{\infty} \beta \gamma^{n \tau(s')} P(s' | s, \boldsymbol{\tau}) \right] \right), \quad (14)$$

where $\xi(s)$ is the initial state distribution. As a benchmark approach, we consider a periodic policy, in which the encoder communicates at regular intervals. This solution, which is common in CPSs, sets a fixed update period $\tau \in \{0, \dots, T_{\max}\}$, which is the same for each state in the system:

$$\boldsymbol{\pi}^*_{\text{per}}(\beta) = \underset{\substack{\tau \in \{0,\dots,T_{\max}\}, \\ \boldsymbol{\pi}_{\mathrm{d}}: \{0,\dots,T_{\max}\} \times \mathcal{S} \mapsto \mathcal{A}}}{\arg\max} R_{\beta}^{\tau, \boldsymbol{\pi}_{\mathrm{d}}}. \quad (15)$$

To improve the performance, we can consider an adaptive pull-based policy, where each communicated message is associated with a different inter-transmission period, with a strategy $\boldsymbol{\tau} \in \{0, \dots, T_{\max}\}^{|\mathcal{S}|}$. The optimization problem hence becomes

$$\boldsymbol{\pi}^*_{\text{pull}}(\beta) = \underset{\substack{\boldsymbol{\tau} \in \{0,\dots,T_{\max}\}^{|\mathcal{S}|}, \\ \boldsymbol{\pi}_{\mathrm{d}}: \{0,\dots,T_{\max}\} \times \mathcal{S} \mapsto \mathcal{A}}}{\arg\max} R_{\beta}^{\boldsymbol{\tau}, \boldsymbol{\pi}_{\mathrm{d}}}. \quad (16)$$

To solve (16), we can exploit the policy iteration strategy presented in Sec. III-A, which can reach the optimal solution is polynomial time, as expressed by Lemma 2.1.

To further improve the performance we can consider the push-based architecture, where the communication policy $\pi_{\mathrm{e}}(s_t, \Delta_t, s_{t-\Delta_t})$ depends on the current state s_t as well as the last state update $s_{t-\Delta_t}$ and the elapsed time Δ_t. The optimal communication policy maximizes the value function

$$\begin{aligned} V_{\mathrm{e}}(s_t, \Delta_t, s_{t-\Delta_t}) &= \pi_{\mathrm{e}}(s_t, \Delta_t, s_{t-k})(V_{\mathrm{e}}(s_t, 0, s_t) - \beta) \\ &+ (1 - \pi_{\mathrm{e}}(s_t, \Delta_t, s_{t-\Delta_t})) \sum_{s_{t+1} \in \mathcal{S}} \gamma P_{s_t s_{t+1}}^{\pi_{\mathrm{d}}(\Delta_t, s_{t-\Delta_t})} \\ &\times \left[\frac{r_{s_t, s_{t+1}}(\pi_{\mathrm{d}}(\Delta_t, s_{t-\Delta_t}))}{\gamma} + V_{\mathrm{e}}(s_{t+1}, \Delta_t + 1, s_{t-\Delta_t}) \right]. \quad (17) \end{aligned}$$

In particular, the superiority of the push-based approach over the pull-based one is proved by Theorem 2, where we write $\boldsymbol{\pi} \succeq_{\beta} \boldsymbol{\pi}'$ to denote that the joint policy $\boldsymbol{\pi}$ outperforms the joint policy $\boldsymbol{\pi}'$, i.e., that $R_{\beta}^{\boldsymbol{\pi}} \geq R_{\beta}^{\boldsymbol{\pi}'}$.

Lemma 2.1. *The MPI scheme in Alg. 1 returns the optimal pull-based joint policy in polynomial time over $|\mathcal{A}|$ and $|\mathcal{S}|$.*

Proof: The modified value function in (6) returns the pull-based policy's state value when considering the decoder's belief. Applying PI over the problem then preserves its optimality properties [47]. The proof that PI is strongly polynomial was first given in [48, Th. 4.2]. Each iteration, even with the modified value and update function, requires $|\mathcal{A}||\mathcal{S}|^2$ steps, resulting in a complexity increase by a factor $|\mathcal{S}|$, which preserves the strongly polynomial nature. ∎

Theorem 2. *The optimal joint policy in the push-based setting always outperforms the optimal joint policy in the pull-based setting, which outperforms any periodic policy:*

$$\boldsymbol{\pi}^*_{push}(\beta) \succeq_{\beta} \boldsymbol{\pi}^*_{pull}(\beta) \succeq_{\beta} \boldsymbol{\pi}^*_{per}(\beta), \ \forall \ \langle \mathcal{S}, \mathcal{A}, \mathbf{P}, r, \gamma, \beta \rangle. \quad (18)$$

Proof: We can first prove that the optimal pull-based policy outperforms any periodic policy by *reductio ad absurdum*: we consider a hypothetical optimal interval $\boldsymbol{\pi}^*_{\text{per}}(\beta)$, which performs better than the pull-based policy for a given value of β. In this case, $\boldsymbol{\pi}^*_{\text{pull}}(\beta)$ cannot be optimal, as the vector in which all elements are equal to $\boldsymbol{\pi}^*_{\text{per}}(\beta)$ is one of the possible choices for pull-based. Then, we prove that the optimal push-based policy outperforms any pull-based policy in the same way: as the encoder can act with full knowledge of the state, the pull-based optimal policy is a possible solution to the push-based problem, and often better push-based policies exist due to an extended information structure. ∎

B. Computational and Performance Limitations

The API scheme presented in Sec. IV-B tries to estimate the optimal value function V_{e} given in (17) by modeling the CPS system as a Markov potential game [49]. In particular, the encoder and the decoder act as players in a game, where the moves are the possible policies, and the payoff for each player is the expected reward in the initial state.

Lemma 3.1. *The API scheme leads to an ε-NE policy in the push-based problem in polynomial time over $|\mathcal{A}|$ and $|\mathcal{S}|$.*

Proof: Firstly, we can trivially prove that the considered Markov game is an exact potential game [40]: as the reward for the two agents is the same, the expected long-term reward is a potential function for the game. We then consider the API scheme: each round of the iterated algorithm leads to the optimal policy when the policy of the other agent is given, due to the optimality of standard policy iteration (PI). The API scheme is then an iterated best response (IBR) scheme for the game, which leads to an NE in a finite number of steps in all finite potential games due to the finite improvement path property [50]. Unfortunately, reaching an NE in an exact potential game may require an exponential number of rounds [51], as exact potential games belong to the

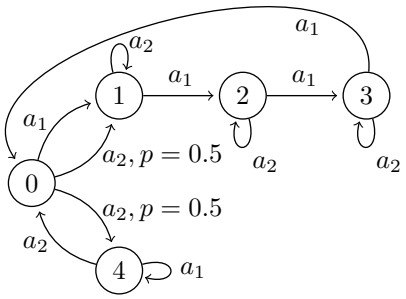


Fig. 1. A Markov model with 5 states and 2 actions in which the API scheme may reach suboptimal solutions.

polynomial local search (PLS) class[4], for which no polynomial-time algorithms are currently known [52]. However, IBR has been shown to converge to an ε-NE in $O\left(\frac{1}{\varepsilon}\right)$ iterations [53]. As each iteration requires $|\mathcal{A}||\mathcal{S}|^2$ steps, the API scheme is still strongly polynomial, as long as we accept approximate convergence. ■

Lemma 3.2. *The solution obtained by the API scheme, $\boldsymbol{\pi}^{API}_{push}(\beta|\boldsymbol{\pi}^0_e)$ does not always outperform the periodic policy:*

$$\exists\langle\mathcal{S},\mathcal{A},\mathbf{P},r,\gamma,\beta\rangle,\boldsymbol{\pi}^0_e : \boldsymbol{\pi}^{API}_{push}(\beta|\boldsymbol{\pi}^0_e) \not\succeq_\beta \boldsymbol{\pi}^*_{per}(\beta). \tag{19}$$

Proof: We can prove the theorem by considering a simple counterexample, represented by an MDP with 5 states and 2 actions, whose evolution is depicted in Fig. 1. The two transition matrices corresponding to a_1 and a_2 are:

$$\mathbf{P}^{a_1} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{P}^{a_2} = \begin{bmatrix} 0 & 0.5 & 0 & 0 & 0.5 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{20}$$

The reward is then always 0, except for when the environment transits to state 0, i.e., we have $r^a_{s,0} = 1$ and $r^a_{s,s'} = 0\ \forall s' \neq 0$.

We can easily see that taking action a_1 in state 0 leads to a loop with 4 states, while taking action a_2 may lead to a shorter path back to the reward-giving state. We consider an encoder policy τ_{per} that transmits in each odd step, ensuring that the decoder always knows if it lands in state 1 or state 4 after taking action a_2. Its expected long-term reward is

$$R^{\boldsymbol{\pi}_{\text{per}}}_\beta = \frac{2-(2\gamma+\gamma^3)\beta}{2(1-\gamma^2-\gamma^4)}. \tag{21}$$

Considering a push-based approach and applying the API scheme, the final results depend on the initial policy of the encoder. If the process starts from a policy that communicates often, e.g., one that always communicates the state, the algorithm will converge to the optimal joint policy, which only communicates if it deviates from the short cycle (i.e., if the system ends up in state 1). The expected reward is

$$R^{\boldsymbol{\pi}^{\text{API}}_{\text{push}}(\beta|\boldsymbol{\pi}^1_{\text{e}})}_\beta = \frac{2-\gamma\beta}{2(1-\gamma^2-\gamma^4)}, \tag{22}$$

[4]The PLS class includes local optimization problem for which the cost of a solution can be computed in polynomial time and its neighborhood can be searched in polynomial time. Several NE computation problems are in this complexity class.

which is better than the periodic policy for any value of $\beta \in \mathbb{R}^+$ and $\gamma \in (0,1)$. On the other hand, if the API scheme starts from a policy that *never* communicates, the decoder will take the conservative choice, and always take action a_1. This is another NE, as the encoder should never communicate if the decoder's policy is independent of the state. The expected reward is

$$R^{\boldsymbol{\pi}^{\text{API}}_{\text{push}}(\beta|\boldsymbol{\pi}^0_{\text{e}})}_\beta = (1-\gamma^4)^{-1}. \tag{23}$$

In this case, the solution provided by the API scheme is not Pareto optimal, as it performs worse than a simple periodic policy if

$$\beta < 2(2+\gamma^2)^{-1}(1-\gamma^4)^{-1}. \tag{24}$$

We can also easily construct a parallel MDP for which starting from always communicating is a suboptimal solution. ■

We now prove that the JPO algorithm presented in Sec. IV-C reaches an ε-optimal solution for the original push-based pragmatic communication problem $\mathcal{M}$.

Theorem 3. *The push-based pragmatic communication problem $\mathcal{M}$ in* (3)*, is equivalent to the single-agent POMDP $\hat{\mathcal{M}}$, whose optimal policy is given by* (12)*.*

Proof: Theorem 1 shows that the optimal encoder policy at time t can be written in the form $\boldsymbol{\pi}_{\text{e},t} \in \mathcal{S}\times\mathcal{O}^t \mapsto \mathcal{C}$. Similarly, the information available to the decoder is $\langle o_{0:t}, a_{0,t-1}\rangle$, the history of messages and the decoder actions, and the optimal decoder policy at time t is $\boldsymbol{\pi}_{\text{d},t} \in \mathcal{O}^{t+1} \mapsto \mathcal{A}$, as we use deterministic policies and the action history can be reconstructed from the observations. Note that the optimal communication action at time $t = 0$ is given by (13). For the rest of the terms including the rewards and the communications costs, the optimization problem can be written, using the notation from POMDP $\hat{\mathcal{M}}$ and the joint policy $\hat{\boldsymbol{\pi}}_t \in \mathcal{O}^{t+1} \mapsto \mathcal{C}^{|\mathcal{S}|} \times \mathcal{A}$, as

$$\begin{aligned}
&\max_{\boldsymbol{\pi}} \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\boldsymbol{\omega}_0,\boldsymbol{\pi}}[r_{s_t,s_{t+1}}(a_t) - \gamma\beta c_{t+1}] = \max_{\boldsymbol{\pi}} \sum_{t=0}^{\infty} \gamma^t \\
&\quad \times \sum_{\langle s_t, o_{0:t}\rangle \in \mathcal{S}\times\mathcal{O}^{t+1}} \Pr(s_t, o_{0:t}|\boldsymbol{\omega}_0, \boldsymbol{\pi}) \sum_{a_t\in\mathcal{A}} \pi_{\text{d},t}(a_t|o_{0:t}) \sum_{s_{t+1}\in\mathcal{S}} P^{a_t}_{s_t,s_{t+1}} \\
&\quad \times \sum_{c_{t+1}\in\mathcal{C}} \pi_{\text{e},t+1}(c_{t+1}|\langle s_{t+1}, o_{0:t}\rangle)[r(s_t,a_t,s_{t+1}) - \gamma\beta c_{t+1}] \\
&= \max_{\hat{\boldsymbol{\pi}}} \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\boldsymbol{\omega}_0,\hat{\boldsymbol{\pi}}}\left[\sum_{\hat{a}_t\in\hat{\mathcal{A}}} \hat{\pi}_t(\hat{a}_t|o_{0:t})\hat{r}(s_t,\hat{a}_t)\right].
\end{aligned} \tag{25}$$

We can then see that $\hat{\mathcal{M}}$ is equivalent to the original problem. This does not restrict the set of achievable policies, as all deterministic policies are separable. It is well-known that infinite horizon discounted POMDPs can be solved optimally by a stationary deterministic policy, which is a function of the belief. As a result, the initial belief $\boldsymbol{\omega}_0$ does not affect the optimal policy $\hat{\boldsymbol{\pi}}^*$. ■

Finding the optimal solution requires learning a continuous function. Numerical algorithms in the literature [54] achieve an ε-optimal lower bound solution for a specified initial belief $\boldsymbol{\omega}_0$, i.e. they find a policy $\boldsymbol{\pi}$ such that $\underline{V}^{\boldsymbol{\pi}}(\boldsymbol{\omega}_0) \leq V^*(\boldsymbol{\omega}_0) \leq$

$\underline{V}^{\boldsymbol{\pi}}(\boldsymbol{\omega}_0) + \varepsilon$. The set of all possible initial beliefs $\boldsymbol{\omega}_0$ for all initial encoder policies and initial state realizations is Ω_0. Thus, ensuring that our policy is ε-optimal lower bounded $\forall \boldsymbol{\omega}_0 \in \Omega_0$ is a sufficient condition for an ε-optimal final policy including the initial encoder policy, as can be trivially shown.

Lemma 4.1. *The computational complexity of the JPO algorithm is at least exponential over $|\mathcal{S}|$.*

Proof: The action space $\hat{\mathcal{A}} = \mathcal{A} \times \mathcal{C}^{|\mathcal{S}|}$ of the modified single-agent POMDP $\hat{\mathcal{M}}$ grows exponentially with $|\mathcal{S}|$. As the modified problem's state space Ω is larger than $\mathcal{S}$, solving the modified problem in polynomial time is impossible, as it would require an $O(\log(|\mathcal{A}|))$ solution to a linear optimization problem. ■

Theorem 4. *Finding the optimal push-based policy $\boldsymbol{\pi}^*_{push}$ is not possible in polynomial time over $|\mathcal{S}|$ and $|\mathcal{A}|$.*

Proof: The complexity of Dec-POMDPs has been studied in [18], [36]. We can easily show that the remote POMDP does not have independent observations [18, Def. 2], as the observation of the encoder depends on the decoder's action. According to [18, Cor. 3], a jointly fully observable Dec-POMDP is NEXP-complete[5] unless it has the independent observation property, along with other properties. As NEXP≠P [55], there is no polynomial-time algorithm that can yield the solution to our problem. This is consistent with results in game theory that show that enumerating NEs is a difficult problem [56]. ■

C. The Remote Estimation Problem

The remote estimation problem is conceptually simpler than the remote control problem, as the decoder's actions do not affect the evolution of the system state. In the pull-based regime, the problem reduces to finding the policy $\boldsymbol{\tau}^*$, as the optimal decoder action is to guess the highest-probability state based on the belief function in (4). This also simplifies the decoder's best response computation in the API scheme: as the decoder's actions do not affect the evolution of the system, the optimal policy is the one that maximizes the immediate reward. Given the binary definition of the reward function, the decoder's best response is then simply

$$\boldsymbol{\pi}^*_{\mathrm{d}}(o_{0:t}, \boldsymbol{\pi}_{\mathrm{e}}) = \arg\max \left(\boldsymbol{\omega}^{\boldsymbol{\pi}_{\mathrm{e}}}_{\langle \Delta_t, s_{t-\Delta_t} \rangle} \right). \tag{26}$$

However, Theorem 4 still holds: due to the interactions between the agents' observations, i.e., to the existence of implicit information, even this special case belongs to the NEXP-complete complexity class [18, Lemma 3].

There are some restrictions of the problem for which a global optimum can be found in polynomial time even in the push-based setting. One such case is the rate of communication minimization problem with the constraint of perfect estimation. This requires the decoder to be certain of the current state at any time: its belief over the underlying state should belong to the set of natural basis vectors of length $|\mathcal{S}|$, i.e., $\boldsymbol{\omega}_t \in \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_{|\mathcal{S}|}\}$.

Theorem 5. *The policy that minimizes the communication cost while guaranteeing perfect state estimation at the decoder sends a message if and only if the current state realization does not correspond to the pre-communication maximum belief.*

Proof: When a message is transmitted, the constraint is always met. On the other hand, the decoder may still be certain of the state even without a transmission if it uses implicit information, i.e., its knowledge of the communication policy, to rule out the states that would have resulted in a transmission according to (5). If the decoder belief at time $t-1$ is $\boldsymbol{\omega}_{t-1} = \mathbf{e}^{s_{t-1}}$, the pre-transmission belief at time t is $\boldsymbol{\omega}'_t = \boldsymbol{P}^{\intercal}\mathbf{e}^{s_{t-1}}$, while the post-transmission belief follows (5). Thus, either transmitting in all states or all but one state are the only ways to obtain a belief vector that is a natural basis vector. Any of these choices results in perfect estimation and are equivalent in terms of the evolution of the system, thus the optimal choice is to minimize the immediate transmission cost, i.e., not to transmit in the likeliest state. ■

Corollary 5.1. *If $\beta < 1$, there are at least $|\mathcal{S}|$ local optima in the push-based remote estimation problem, even after relaxing the perfect estimation constraint.*

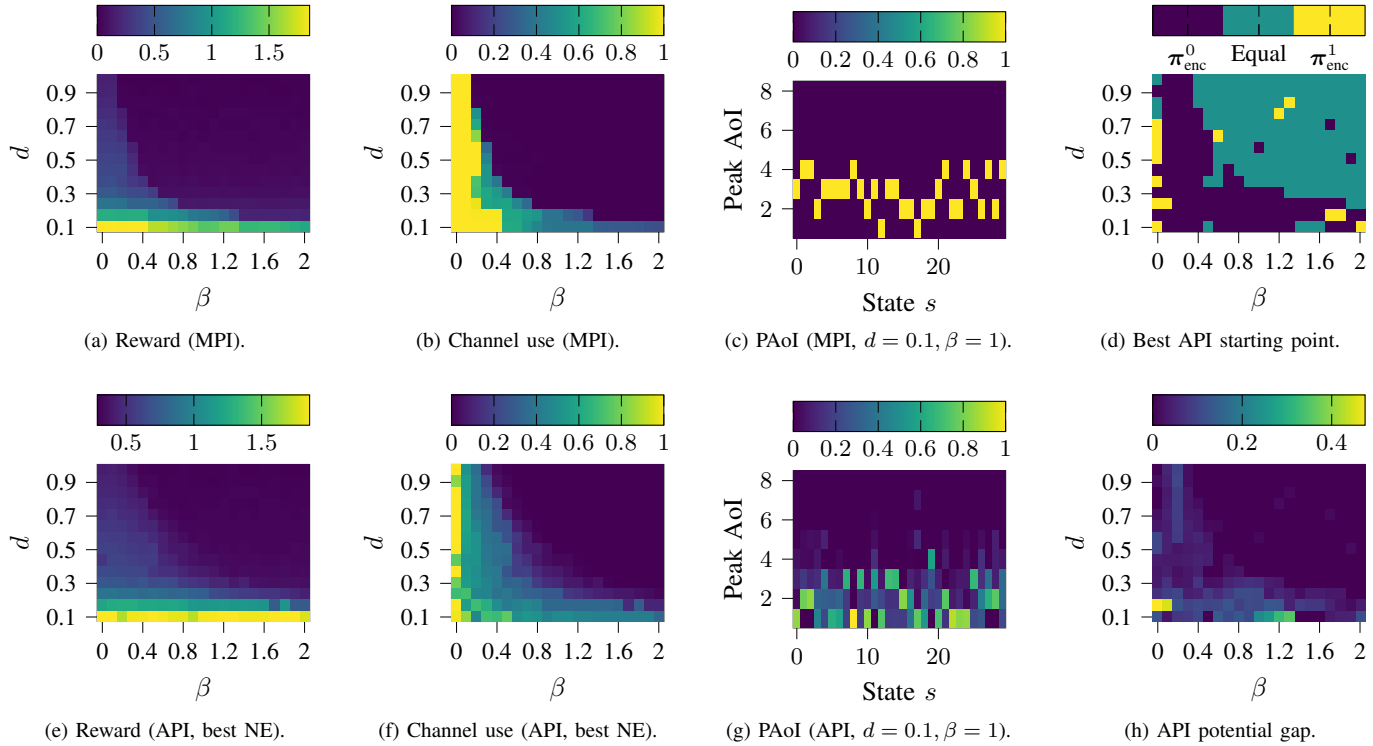
Proof: Let us consider the following solution: the encoder sends an update for all the states, except state s, while the decoder estimates the received state if $\Delta_t = 0$ and s if $\Delta_t > 0$. If $\beta < 1$, this solution is an NE: the decoder has no incentive to change, as it always estimates the correct state. If the encoder switches to not transmitting in another state s', its total reward is $1 - \beta > 0$. Transmitting in state s would mean losing the benefit of implicit information, and having a total reward β. As the two policies are mutual best responses, the solution is an NE for any s, i.e., there are at least $|\mathcal{S}|$ NEs. ■

VI. Numerical Results

In this section, we perform numerical simulations to validate the correctness of our theoretical results. We observe that, because of the general formulation of MDPs, providing a comprehensive performance evaluation is not trivial. An MDP could be obtained by randomly generating rewards and transition matrices, but results would be hard to compare and interpret. In order to provide a meaningful comparison, we hence consider a family of MDPs generated by varying the number of non-zero elements in the transition matrices.

Specifically, we start from a randomly generated MDP with deterministic transitions for each state and action and then obtain different MDPs by splitting the transition probability among neighboring states. The resulting MDPs have transition matrices with different densities d, where the density is defined as the number of non-zero elements divided by the total number of entries of the matrix. In the case of the deterministic transition matrix, the density is simply $|\mathcal{S}|^{-1}$. If the initial

[5]The NEXP class represents the set of problems that can be solved by non-deterministic Turing machines in time $\exp\left(N^{O(1)}\right)$, where N is the size of the problem.



(a) Reward (MPI). (b) Channel use (MPI). (c) PAoI (MPI, $d = 0.1, \beta = 1$). (d) Best API starting point.

(e) Reward (API, best NE). (f) Channel use (API, best NE). (g) PAoI (API, $d = 0.1, \beta = 1$). (h) API potential gap.

Fig. 2. Results for the remote control task in an MDP with 30 states.

deterministic transition matrix $\mathbf{P}^a$ has a transition from s to s', i.e., $P^a_{s,s'} = 1$, the distribution at density d is:

$$P^a_{s,s''} = \frac{\frac{(5+\delta_{s',s''})m}{5} - 2|s' - s''|}{\frac{(m-1)^2}{2} + \frac{6m}{5}}, \ \forall s'' \text{ s.t. } |s' - s''| < \frac{d|\mathcal{S}|}{2}, \tag{27}$$

where $m = d|\mathcal{S}|$ is the number of non-zero elements, and the index distance is over the Galois field of size $|\mathcal{S}|$. According to this approach, the probability of transitioning to neighboring states decreases linearly with the distance from the original transition state. Higher-density MDPs are then simply less predictable versions of the same initial model.

In the remote control problem, we consider a two-peak model for the reward: there is a target state that gives a high reward, but also a lower-value state where the agent gets a smaller reward for reaching the state or its immediate neighborhood. In the remote estimation special case, the reward function follows the binary model given in Sec. III-C. All the code used in this work is available online[6].

A. General Case: Remote Control

For the remote control problem, we consider MDPs with 30 states, evaluating the system performance as a function of the density d of the transition matrix and the communication cost β. Fig. 2 shows the results for the MPI and API schemes. In particular, we can compare the performance of the MPI scheme in Fig. 2a-c with the that of the API scheme in Fig. 2d-g.

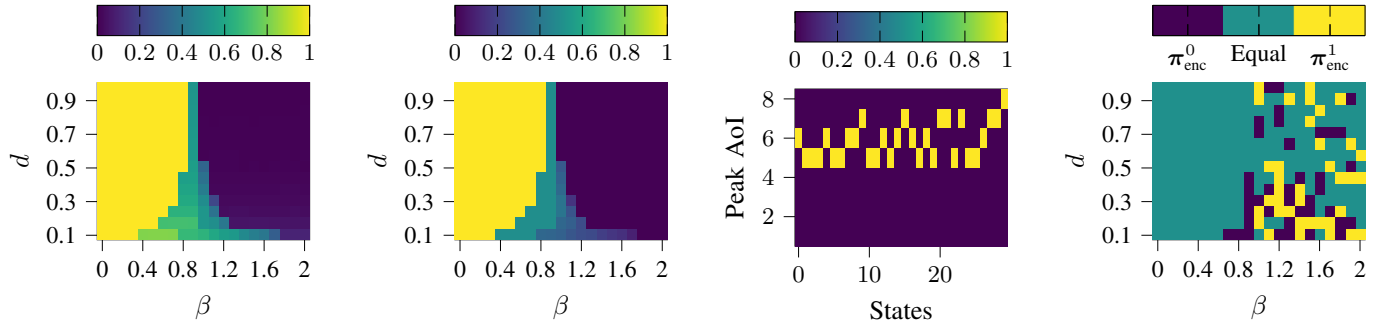
[6]https://github.com/pietro-talli/EffCom

In the pull-based setting, the system tends to transmit more often, obtaining a similar or lower reward: as expected, this is because the decoder has no additional information to decide when to request new updates other than the last transmitted state. This is also evident in the peak AoI (PAoI), which is a deterministic function of the last observed state in the pull-based setting. While the channel use is monotonically decreasing in β, the trade-off between the reward and the channel use is more complex when we consider it as a function of the density d.

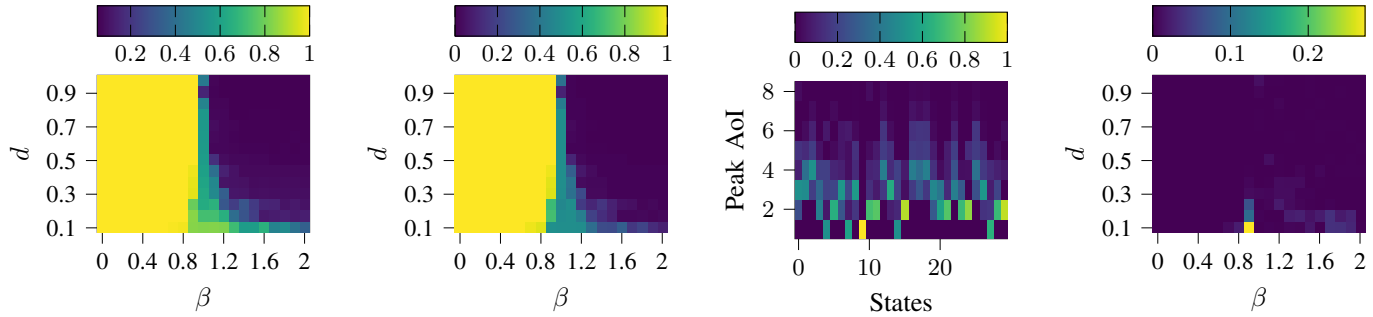
In the push-based setting, we considered two different starting points for the API scheme, $\boldsymbol{\pi}^0_{\text{e}}$ and $\boldsymbol{\pi}^1_{\text{e}}$, which correspond to the two extreme policies of never and always transmitting, respectively. Fig. 2d shows that the two starting points often reach the same solution (within a margin of $\varepsilon = 10^{-3}$) if the communication cost is high, and reach different equilibria otherwise. Fig. 2h also shows the potential gap between the two NEs, which is generally small, except for a few cases.

B. Special Case: Remote Estimation

We perform the same analysis for the remote estimation problem. In this case, the transition matrix is fixed and does not depend on the decoder's actions. Fig. 3 shows the results in a structure analogous to that of Fig. 2. A significant difference with respect to the remote control problem is that, in the remote estimation case, we begin to see a trade-off between the estimation error and the channel use only at higher communication costs.



(a) Reward (MPI). (b) Channel use (MPI). (c) PAoI (MPI, $d=0.1, \beta=1.5$). (d) Best API starting point.



(e) Reward (API, best NE). (f) Channel use (API, best NE). (g) PAoI (API, $d=0.1, \beta=1.5$). (h) API potential gap.

Fig. 3. Results for the remote estimation task in an MDP with 30 states.

Another difference is that, at lower values of d, the system is able to take advantage of the more predictable transitions; and thus, can reduce the channel use even at lower values of β. This is in sharp contrast with the control case, as there is no monotonic relationship between d and the channel use in that case: this is because the decoder is able to affect the state transitions, and thus the probability of future communications, by changing its policy, leading to a more complicated interaction between the control and communication policies. On the other hand, the channel use becomes a monotonically increasing function of d as well as a decreasing function of β in the estimation case, as the decoder can only estimate the state but not affect its evolution. This also leads solution reached by the API scheme, under both starting points, to be extremely close to the solution of the pull-based MPI scheme.

C. Optimality Gap

In the following, we compare the MPI and the API schemes with the JPO scheme in the two cases. Due to computational issues, we consider a single, smaller MDP, with $|\mathcal{S}| = 10$, plotting the trade-off curves between the average reward and the channel use. Each point in Figs. 4 and 5 corresponds to a different value of β. However, a different value of β does not necessarily result in a different solution[7].

[7]Deterministic policies can be combined with time-sharing, i.e., every time we reach a certain belief, we proceed with one of the policies with a certain probability and with the other with another probability. This allows us to obtain any convex combination of the performance of any two policies obtained using a specific algorithm, and thus the Pareto region is convex.

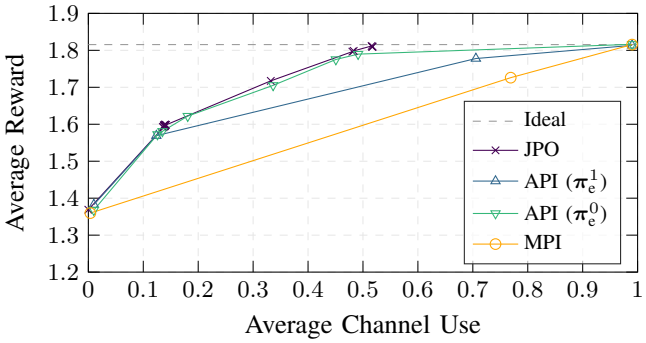


Fig. 4. The trade-off curves between the average reward and the average channel use in the control task with 10 states. The density of the transition matrix is $d = 0.9$.

We compare the Pareto frontiers between the reward and the channel use for the three proposed algorithms in Fig. 4. For all the algorithms, the performance is trivially similar when β is high and the encoder almost never communicates. As the communication cost increases, the JPO and API schemes with π^0_e initialization achieve similar performances, showing that the API scheme converges almost to the Pareto-efficient NE. When it is initialized with π^1_e, the API scheme achieves a lower reward, converging to a worse NE. However, the performance in the latter case still Pareto dominates the MPI scheme, showing that the performance cost of restricting the problem to the pull-based case is significant. The results in Fig. 4 are also consistent with Fig. 2d, as the π^0_e initialization seems to be a better starting point for the API scheme than π^1_e

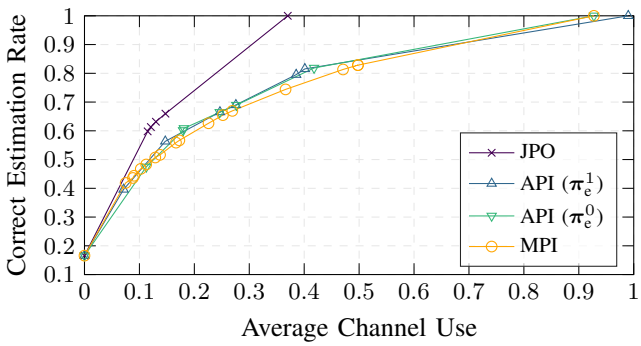


Fig. 5. The trade-off curves between the average correct estimation rate and the average channel use in the estimation task with 10 states. The density of the transition matrix is $d = 0.19$ and the matrix was built *ad hoc*.

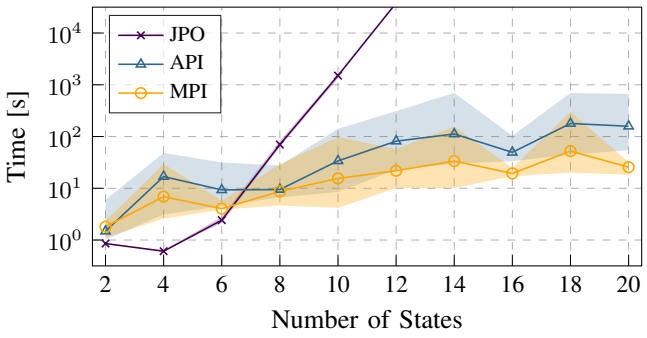


Fig. 6. Running time of the proposed algorithms as a function of the size of the MDP.

in the larger MDP.

Conversely, for the remote estimation problem, we can see that the API scheme is far from the optimum. Fig. 5 shows the JPO scheme obtaining a very good trade-off between the reward and the channel use. However, both API initializations have a very poor performance, and that the iterative procedure terminates in a relatively bad NE in both cases. The working points obtained in this scenario are similar to the pull-based strategy, which we have proved to be a sub-optimal solution to the remote estimation problem. A possible explanation of this behavior is that a high level of cooperation through implicit information is more difficult to obtain, as the decoder has fewer ways to influence the trajectory across the state space. Other approaches might mitigate this issue by using different initializations, but the remote estimation problem seems to be generally harder for iterative policies.

Finally, we can confirm the asymptotic running time of the algorithms: Fig. 6 represents the running time, averaged over 10 random MDPs, of the three algorithms, over the same hardware. While the implementation of the JPO scheme is more efficient due to the use of Python libraries that exploit parallel computation, its running time grows exponentially, taking hours even for relatively small MDP. On the other hand, the MPI and API schemes show a polynomial growth, corresponding to a sub-linear increase over the logarithmic plot, even though the specific implementation is relatively inefficient.

VII. Conclusions

In this paper, we have developed a theoretical framework that provides general results and theoretical limits for the efficiency of pragmatic communication. Our model considers the estimation and control of finite-state Markov processes over costly zero-delay communication channels, and includes two decision-making architectures, in which communication is either pull-based or push-based. We proposed three algorithms to optimize our system, discussed different optimality solution concepts associated with these algorithms, and shed light on their computational complexity and fundamental limits. We showed that while the optimal solution can be reached in polynomial time in the pull-based case, which represents a restricted version of the problem with a significant performance gap, the push-based case poses a trade-off between global optimality and computational complexity.

We should highlight that the trade-off between communication cost and application performance becomes even more important for safety-critical, high-throughput systems such as autonomous vehicles, proving the need for pragmatic communication. The framework developed in this paper is flexible enough to allow for several future extensions. These might include more complex scenarios involving communication impairments such as time delay, packet loss, and channel noise. Another interesting extension would be to consider multiple nodes, which may have different information about the system state, requiring them to reason about each other's beliefs to coordinate their actions and transmissions over the shared channel.

References

[1] P. Talli *et al.*, "Push-and pull-based effective communication in cyber-physical systems," in *IEEE INFOCOM Wkshp. Age and Semantics of Inf.*, 2024.

[2] E. D. Santi, T. Soleymani, and D. Gunduz, "Remote estimation of Markov processes over costly channels: On the benefits of implicit information," *Submitted to IEEE Globecom (arXiv:2401.17999)*, 2024.

[3] J. Baillieul and P. J. Antsaklis, "Control and communication challenges in networked real-time systems," *Proc. IEEE*, vol. 95, pp. 9–28, 2007.

[4] K.-D. Kim and P. R. Kumar, "Cyber–Physical Systems: A perspective at the centennial," *Proc. IEEE*, vol. 100, pp. 1287–1308, 2012.

[5] E. A. Lee *et al.*, "Consistency vs. availability in distributed cyber-physical systems," *ACM Trans. Embedded Computing Sys.*, vol. 22, p. 138, 2023.

[6] P. Popovski, "Time, simultaneity, and causality in wireless networks with sensing and communications," *IEEE Open J. Commun. Soc.*, vol. 5, pp. 1693–1709, 2024.

[7] T. Soleymani, J. S. Baras, S. Hirche, and K. H. Johansson, "Feedback control over noisy channels: Characterization of a general equilibrium," *IEEE Trans. Autom. Control*, vol. 67, no. 7, pp. 3396–3409, 2021.

[8] L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Trans. Sys., Man, and Cybernetics, Part C*, vol. 38, no. 2, pp. 156–172, 2008.

[9] D. Gündüz *et al.*, "Beyond transmitting bits: Context, semantics, and task-oriented communications," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 1, pp. 5–41, 2023.

[10] E. Uysal *et al.*, "Semantic communications in networked systems: A data significance perspective," *IEEE Network*, vol. 36, pp. 233–240, 2022.

[11] R. D. Yates *et al.*, "Age of information: An introduction and survey," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 5, pp. 1183–1210, 2021.

[12] S. Banerjee, R. Bhattacharjee, and A. Sinha, "Fundamental limits of age-of-information in stationary and non-stationary environments," in *Intl. Symp. Inf. Theory*. IEEE, 2020, pp. 1741–1746.

[13] T. Soleymani, J. S. Baras, and S. Hirche, "Value of Information in feedback control: Quantification," *IEEE Trans. Autom. Control*, vol. 67, no. 7, pp. 3730–3737, 2022.

[14] T. Soleymani, J. S. Baras, S. Hirche, and K. H. Johansson, "Value of Information in feedback control: Global optimality," *IEEE Trans. Autom. Control*, vol. 68, no. 6, pp. 3641–3647, 2023.

[15] D. Gündüz *et al.*, "Timely and massive communication in 6G: Pragmatics, learning, and inference," *IEEE BITS Inf. Theory Mag.*, vol. 3, no. 1, pp. 27–40, 2024.

[16] A. Maatouk, M. Assaad, and A. Ephremides, "The age of incorrect information: An enabler of semantics-empowered communication," *IEEE Trans. Wireless Commun.*, vol. 22, no. 4, pp. 2621–2635, 2022.

[17] P. Talli, F. Pase, F. Chiariotti, A. Zanella, and M. Zorzi, "Effective communication with dynamic feature compression," *IEEE Trans. Commun.*, 2024, Early Access.

[18] C. V. Goldman and S. Zilberstein, "Decentralized control of cooperative systems: Categorization and complexity analysis," *J. Artif. Intelligence Res.*, vol. 22, pp. 143–174, 2004.

[19] J. Walrand and P. Varaiya, "Optimal causal coding-decoding problems," *IEEE Trans. Inf. Theory*, vol. 29, no. 6, pp. 814–820, 1983.

[20] R. G. Wood, T. Linder, and S. Yüksel, "Optimal zero delay coding of Markov sources: Stationary and finite memory codes," *IEEE Trans. Inf. Theory*, vol. 63, no. 9, pp. 5968–5980, 2017.

[21] L. Cregg, F. Alajaji, and S. Yüksel, "Reinforcement learning for zero-delay coding over a noisy channel with feedback," in *Conf. Decision and Control*. IEEE, 2023, pp. 3939–3944.

[22] J. Chakravorty and A. Mahajan, "Fundamental limits of remote estimation of autoregressive Markov processes under communication constraints," *IEEE Trans. Autom. Control*, vol. 62, pp. 1109–1124, 2016.

[23] G. M. Lipsa and N. C. Martins, "Remote State Estimation With Communication Costs for First-Order LTI Systems," *IEEE Trans. Autom. Control*, vol. 56, no. 9, pp. 2013–2025, 2011.

[24] A. Molin and S. Hirche, "Event-triggered state estimation: An iterative algorithm and optimality properties," *IEEE Trans. Autom. Control*, vol. 62, no. 11, pp. 5939–5946, 2017.

[25] T. Soleymani, J. S. Baras, and K. H. Johansson, "State estimation over delayed and lossy channels: An encoder-decoder synthesis," *IEEE Trans. Autom. Control*, vol. 69, no. 3, pp. 1658–1583, 2024.

[26] M. Salimnejad, M. Kountouris, and N. Pappas, "State-aware real-time tracking and remote reconstruction of a Markov source," *J. Commun. and Networks*, vol. 25, no. 5, pp. 657–669, 2023.

[27] ——, "Real-time reconstruction of Markov sources and remote actuation over wireless channels," *IEEE Trans. Commun.*, 2024, Early Access.

[28] M. Krale, T. D. Simão, and N. Jansen, "Act-Then-Measure: Reinforcement learning for partially observable environments with active measuring," in *Intl. Conf. Automated Planning and Scheduling*, vol. 33. AAAI, 2023, pp. 212–220.

[29] C. Bellinger, M. Crowley, and I. Tamblyn, "Dynamic observation policies in observation cost-sensitive reinforcement learning," in *Adv. Neural Inf. Proc. Sys. WANT Wkshp.*, vol. 36, 2023.

[30] H. A. Nam, S. Fleming, and E. Brunskill, "Reinforcement learning with state observation costs in action-contingent noiselessly observable Markov decision processes," in *Adv. Neural Inf. Proc. Sys.*, vol. 34, 2021, pp. 15 650–15 666.

[31] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Adv. Neural Inf. Proc. Sys.*, vol. 29, 2016.

[32] T.-Y. Tung, S. Kobus, J. P. Roig, and D. Gündüz, "Effective communications: A joint learning and communication framework for multi-agent reinforcement learning over noisy channels," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2590–2603, 2021.

[33] F. Mason, F. Chiariotti, A. Zanella, and P. Popovski, "Multi-agent reinforcement learning for coordinating communication and control," *IEEE Trans. Cognitive Commun. and Networking*, 2024, Early Access.

[34] R. Wang *et al.*, "Learning efficient multi-agent communication: An information bottleneck approach," in *Intl. Conf. Machine Learning*. PMLR, 2020, pp. 9908–9918.

[35] W. Kim, W. Jung, M. Cho, and Y. Sung, "A variational approach to mutual information-based coordination for multi-agent reinforcement learning," in *Intl. Conf. Autonomous Agents and Multiagent Sys.*, 2023, pp. 40–48.

[36] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of Markov decision processes," *Math. Op. Res.*, vol. 27, no. 4, pp. 819–840, 2002.

[37] F. A. Oliehoek, M. T. J. Spaan, and N. Vlassis, "Optimal and approximate Q-value functions for decentralized POMDPs," *J. Artif. Intelligence Res.*, vol. 32, pp. 289—-353, 2008.

[38] J. Pineau *et al.*, "Point-based value iteration: An anytime algorithm for POMDPs," in *Intl. Joint Conf. on AI*, vol. 3, 2003, pp. 1025–1032.

[39] T. Smith and R. Simmons, "Point-based POMDP algorithms: improved analysis and implementation," in *Conf. on Uncertainty in AI*. UAI, 2005, pp. 542–549.

[40] D. Monderer and L. S. Shapley, "Potential games," *Games and Economic Behavior*, vol. 14, no. 1, pp. 124–143, 1996.

[41] F. Pase, D. Gündüz, and M. Zorzi, "Rate-constrained remote contextual bandits," *IEEE J. Sel. Areas Inf. Theory*, vol. 3, pp. 789–802, 2022.

[42] V. Gupta and B. Vineeth, "Remote control of bandits over queues–relevance of information freshness," in *Intl. Symp. on Modeling and Opt. in Mobile, Ad Hoc, and Wireless Networks*, 2023, pp. 619–626.

[43] A. S. Leong *et al.*, "Stability enforced bandit algorithms for channel selection in remote state estimation of Gauss-Markov processes," *IEEE Trans. Autom. Control*, vol. 68, pp. 8308–8315, 2023.

[44] H. Kurniawati, D. Hsu, and W. S. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces," in *Robotics Sci. & Sys. Conf.*, 2008, pp. 65–72.

[45] M. Hauskrecht, "Incremental methods for computing bounds in partially observable Markov decision processes," in *14th Nat. Conf. on AI*. AAAI, 1997, pp. 734–739.

[46] ——, "Value-function approximations for partially observable Markov decision processes," *J. Artif. Intelligence Res.*, vol. 13, pp. 33–94, 2000.

[47] R. A. Howard, *Dynamic programming and Markov processes*. MIT Press & John Wiley, 1960.

[48] Y. Ye, "The simplex and policy-iteration methods are strongly polynomial for the Markov decision problem with a fixed discount rate," *Math. Op. Res.*, vol. 36, no. 4, pp. 593–603, 2011.

[49] X. Wang and T. Sandholm, "Reinforcement learning to play an optimal Nash equilibrium in team Markov games," in *Adv. Neural Inf. Proc. Sys.*, vol. 15, 2002.

[50] N. S. Kukushkin, "Best response dynamics in finite games with additive aggregation," *Games and Economic Behavior*, vol. 48, no. 1, pp. 94–110, 2004.

[51] A. A. Schäffer, "Simple local search problems that are hard to solve," *SIAM J. Computing*, vol. 20, no. 1, pp. 56–87, 1991.

[52] M. Yannakakis, "Equilibria, fixed points, and complexity classes," *Computer Sci. Rev.*, vol. 3, no. 2, pp. 71–85, 2009.

[53] G. Christodoulou, V. S. Mirrokni, and A. Sidiropoulos, "Convergence and approximation in potential games," *Theoretical Computer Sci.*, vol. 438, pp. 13–27, 2012.

[54] M. H. Lim *et al.*, "Optimality guarantees for particle belief approximation of POMDPs," *J. Artif. Intelligence Res.*, vol. 77, pp. 1591–1636, 2023.

[55] J. I. Seiferas, M. J. Fischer, and A. R. Meyer, "Separating nondeterministic time complexity classes," *J. ACM*, vol. 25, pp. 146–167, 1978.

[56] I. Gilboa and E. Zemel, "Nash and correlated equilibria: Some complexity considerations," *Games and Economic Behavior*, vol. 1, no. 1, pp. 80–93, 1989.