entropy

MDPI

Article

On the Interactive Capacity of Finite-State Protocols †

Assaf Ben-Yishai [1,*], Young-Han Kim [2], Rotem Oshman [3] and Ofer Shayevitz [4]

[1] School of Computer Science and Engineering, Hebrew University of Jerusalem, Jerusalem 9190401, Israel
[2] Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093 USA; yhk@ucsd.edu
[3] Department of Computer Science, Tel Aviv University, Tel Aviv 6997801, Israel; roshman@tauex.tau.ac.il
[4] Department of EE–Systems, Tel Aviv University, Tel Aviv 6997801, Israel; ofersha@eng.tau.ac.il
* Correspondence: assafbster@gmail.com
† This paper was presented in part at the 2019 IEEE International Symposium on Information Theory (ISIT), Paris, France, 7–12 July 2019.

Abstract: The interactive capacity of a noisy channel is the highest possible rate at which arbitrary interactive protocols can be simulated reliably over the channel. Determining the interactive capacity is notoriously difficult, and the best known lower bounds are far below the associated Shannon capacity, which serves as a trivial (and also generally the best known) upper bound. This paper considers the more restricted setup of simulating finite-state protocols. It is shown that all two-state protocols, as well as rich families of arbitrary finite-state protocols, can be simulated at the Shannon capacity, establishing the interactive capacity for those families of protocols.

Keywords: interactive communication; Shannon theory; channel capacity

1. Introduction

In the classical one-way communication problem, a transmitter (Alice) wishes to reliably send a message to a receiver (Bob) over a memoryless noisy channel. She does so by mapping her message into a sequence of channel inputs (a codeword) in a predetermined way, which is corrupted by the channel and then observed by Bob, who tries to recover the original message. The Shannon capacity of the channel quantifies the most efficient way of conducting reliable communication, and is defined as the maximal number of message bits per channel-use that Alice can convey to Bob with vanishingly low error probability. In the two-way channel setup [1], both parties draw independent messages and wish to exchange them over a two-input two-output memoryless noisy channel, and the Shannon capacity (region) is defined similarly. Unlike the one-way case, both parties can now employ adaptive coding, by incorporating their respective observations of the past channel outputs into their transmission processes. However, just as in the one-way setup, the messages they wish to exchange are determined before communication begins. In other words, if Alice and Bob had been connected by a noiseless bit pipe, they could have simply sent their messages without any regard to the message of their counterpart.

In a different two-way communication setup, generally referred to as interactive communication, the latter assumption is no longer held true. In this interactive communication setup, Alice and Bob do not necessarily wish to disclose all their local information. What they want to tell each other depends, just like in human conversation, on what the other would tell them. A simple instructive example (taken from [2]) is the following. Suppose that Alice and Bob play correspondence chess. Namely, they are located in two distinct places and play by announcing their moves over a communication channel (using, say, 12 bits per move, which is clearly sufficient). If the moves are conveyed without error, then both parties can keep track of the state of the board, and the game can proceed to its termination. The sequence of moves occurring over the course of this noiseless game is

Citation: Ben-Yishai, A.; Kim, Y.-H.; Oshman, R.; Shayevitz, O. On the Interactive Capacity of Finite-State Protocols. *Entropy* **2021**, *23*, 17. https://doi.org/10.3390/e23010017

Received: 3 December 2020
Accepted: 22 December 2020
Published: 25 December 2020

called a transcript, and it is dictated by the protocol of the game, which constitutes Alice and Bob's respective strategies determining their moves at any given state of the board.

Now, assume that Alice and Bob play a chess game over a noisy two-way channel, yet wish to simulate the transcript as if no noise were present. In other words, they would like to communicate back and forth in a way that ensures, once communication is over, that the transcript of the noiseless game can be reproduced by both parties with a small error probability. They would also like to achieve this goal as efficiently as possible, i.e., with the least number of channel uses. One direct way to achieve this is by having both parties describe their entire protocol to their counterpart, i.e., each and every move they might take given each and every possible state of the board. This reduces the interactive problem to a non-interactive one, with the protocol becoming a pair of messages to be exchanged. However, this solution is grossly inefficient; the parties now know much more than they really need to in order to simply reconstruct the transcript. At the other extreme, Alice and Bob may choose to describe the transcript itself by encoding each move separately on-the-fly, using a short error correcting code. Unfortunately, this code must have some fixed error probability, and hence an undetected error is bound to occur at some unknown point, causing the states of the board held by the two parties to diverge, and rendering the remainder of the game useless. It is important to note that if Alice and Bob had wanted to play sufficiently many games in parallel, then they could have used a long error-correcting code to simultaneously protect the set of all moves taken at each time point, which in principle would have let them operate at the one-way Shannon capacity (which is the best possible). The crux of the matter therefore lies in the fact that the interactive problem is one-shot, namely, only a single instance of the game is played.

In light of the above, it is perhaps surprising that it is nevertheless possible to simulate any one-shot interactive protocol using a number of channel uses that is proportional to the length of the transcript, or in other words, that there is a positive interactive capacity whenever the Shannon capacity is positive. This fact was originally proved by Schulman [3], who was also the first to introduce the notion of interactive communication over noisy channels. The lower bound on the interactive capacity was recently studied in [4], and was found to be at least 0.0302 of the Shannon capacity for all binary memoryless symmetric channels.

Main Contributions

Characterizing the interactive capacity of a channel (a definition that also depends on many auxiliary assumptions such as order of speakers, randomness resources, and more [4]) remains a formidable task, which is currently yet to be completed. One of the main obstacles to that end lies in the fact that one needs to guarantee a reliable simulation of all possible protocols. In this work, rather than taking this intractable worst-case approach, we study a relaxed notion of interactive capacity where constraints are imposed on the family of protocols to be simulated; this relaxation still retains the challenging interactive nature of the problem, but at the same time facilitates a more complete characterization of the fundamental limits. Specifically, we define the family of finite-state protocols, and show that for a large class of these protocols, the Shannon capacity is achievable. In particular, we prove that Shannon capacity is achievable for all protocols having only two states (two-state protocols). For larger state-spaces, it is easy to show that almost all protocols can be reliably simulated at the Shannon capacity. However, this is simply due to the very mundane reason that almost all protocols will occasionally "reset" to a known state, thereby decoupling them into many smaller protocols, which can be easily exploited for coding. However, real-world interactive protocols are clearly not likely to contain this type of reset. Nevertheless, we show that there is a rich family of protocols that can never reset, whose (almost all) members can be reliably simulated at the Shannon capacity. We note that the approach of studying the interactive capacity of protocols having a specific structure was previously taken in [5]. The authors of [5] limited the "interactiveness" of the protocols by considering families of protocols whose transcript is predictable to a

certain extent, and proved that they can be simulated in higher rates than general protocols. The constraints imposed on the protocols in this paper are, however, on the memory of the protocols and not on their predictability.

The rest of the paper is organized as follows: In Section 2 the interactive communication problem is formulated. In Section 3, finite-state (or M-state) protocols, which are the main model discussed in this paper, are defined. In Section 4, the basic concepts of the coding schemes are presented. In Section 5, a capacity achieving coding scheme for two-state protocols is presented. In Section 6, it is proved that the concepts in Section 4 cannot be used for at least one three-state protocol. In Section 7, families of finite-state protocols for which almost-all members can be simulated at Shannon capacity are presented. Finally, Section 8 concludes the paper.

A preliminary version of some of the results in this paper appeared in [6]. Here we extend upon the results of [6] as follows: First, ref. [6] only considered Markovian protocols, a special case of the type of protocols we consider here. Second, [6] gave a simple special case of the coding scheme of Section 4; here we generalize the scheme and give two methods that can handle more complex protocols, beyond Markovian. Finally, the Shannon capacity achieving scheme for two-state protocols in Section 5, the inachievability results for three states in Section 6 and the scheme for higher order models in Section 7 appear here for the first time.

2. The Interactive Communication Problem

In this paper, we define a length-n interactive protocol as a triplet $\boldsymbol{\pi} \triangleq (\boldsymbol{\phi}^{\text{Alice}}, \boldsymbol{\phi}^{\text{Bob}}, \boldsymbol{\mu})$, where:

$$\begin{aligned}\boldsymbol{\phi}^{\text{Alice}} &\triangleq \left\{\phi_i^{\text{Alice}} : \{0,1\}^{i-1} \mapsto \{0,1\}\right\}_{i=1}^{n}\\ \boldsymbol{\phi}^{\text{Bob}} &\triangleq \left\{\phi_i^{\text{Bob}} : \{0,1\}^{i-1} \mapsto \{0,1\}\right\}_{i=1}^{n}\\ \boldsymbol{\mu} &\triangleq \left\{\mu_i : \{0,1\}^{i-1} \mapsto \{\text{Alice}, \text{Bob}\}\right\}_{i=1}^{n}.\end{aligned}$$

The functions $\boldsymbol{\phi}^{\text{Alice}}$ are known only to Alice, and the functions $\boldsymbol{\phi}^{\text{Bob}}$ are known only to Bob. The speaker order functions $\boldsymbol{\mu}$ are known to both parties. The transcript $\boldsymbol{\tau}$ associated with the protocol $\boldsymbol{\pi}$ is sequentially generated by Alice and Bob as follows:

$$\tau_i = \begin{cases}\phi_i^{\text{Alice}}(\boldsymbol{\tau}^{i-1}) & \sigma_i = \text{Alice}\\ \phi_i^{\text{Bob}}(\boldsymbol{\tau}^{i-1}) & \sigma_i = \text{Bob},\end{cases} \tag{1}$$

where σ_i is the identity of the speaker at time i, which is given by:

$$\sigma_i = \mu_i(\boldsymbol{\tau}^{i-1}). \tag{2}$$

In the interactive simulation problem, Alice and Bob would like to simulate the transcript $\boldsymbol{\tau}$, by communicating back and forth over a noisy memoryless channel $P_{Y|X}$. Specifically, we restrict our discussion to channels with a binary input alphabet $\mathcal{X} = \{0,1\}$, and a general (possibly continuous) output alphabet $\mathcal{Y}$. We use $\mathsf{C}_{\mathsf{Sh}}(P_{Y|X})$ to denote the Shannon capacity of the channel. Note that $\mathsf{C}_{\mathsf{Sh}}(P_{Y|X}) \leq 1$, since the input of the channel is binary. Naturally, we also limit the discussion to channels whose Shannon capacity is non-zero.

Note that while the order of speakers in the interactive protocol itself might be determined on-the-fly (by the sequence of functions $\boldsymbol{\mu}$), we restrict the simulating protocol to use a predetermined order of speakers. The reason is that allowing an adaptive order over a noisy channel will lead to a non-zero probability of disagreement regarding the order of speakers. This disagreement might lead to simultaneous transmissions at both parties, which is not supported by the chosen physical channel model

To achieve their goal, Alice and Bob employ a length-N coding scheme Σ that uses the channel N times. The coding scheme consists of a disjoint partition $\tilde{A} \sqcup \tilde{B} = \{1, \ldots, N\}$

where $\tilde{A}$ (resp. $\tilde{B}$) is the set of time indices where Alice (resp. Bob) speaks. This disjoint partition can be a function of $\boldsymbol{\mu}$, but not of $\boldsymbol{\phi}^{\text{Alice}}, \boldsymbol{\phi}^{\text{Bob}}$. At time $j \in \tilde{A}$ (resp. $j \in \tilde{B}$), Alice (resp. Bob) sends some Boolean function $X_j^{\textbf{Alice}}$ (resp. X_j^{Bob}) of $\boldsymbol{\phi}^{\text{Alice}}$ (resp. $\boldsymbol{\phi}^{\text{Bob}}$) and $\boldsymbol{\mu}$, and of all channel outputs (to be defined immediately) received so far from her (resp. his) counterpart. This sent bit is transmitted through a memoryless channel with law $P_{Y|X}$, and the received output viewed by Bob (resp. Alice) is denoted by Y_j^{Bob} (resp. Y_j^{Alice}). The rate of the scheme is $R = \frac{n}{N}$ bits per channel use. When communication terminates, Alice and Bob produce their simulations of the transcript τ, denoted by $\hat{\boldsymbol{\tau}}_A(\Sigma, \boldsymbol{\phi}^{\text{Alice}}, \boldsymbol{\mu}, \boldsymbol{Y}^{\text{Alice}}) \in \{0,1\}^n$ and $\hat{\boldsymbol{\tau}}_B(\Sigma, \boldsymbol{\phi}^{\text{Bob}}, \boldsymbol{\mu}, \boldsymbol{Y}^{\text{Bob}}) \in \{0,1\}^n$, respectively. The error probability attained by the coding scheme is the probability that either of these simulations is incorrect, i.e.,

$$P_e(\Sigma, \boldsymbol{\pi}) \triangleq \Pr\Big(\hat{\boldsymbol{\tau}}_A(\Sigma, \boldsymbol{\phi}^{\text{Alice}}, \boldsymbol{\mu}, \boldsymbol{Y}^{\text{Alice}}) \neq \boldsymbol{\tau} \ \vee \ \hat{\boldsymbol{\tau}}_B(\Sigma, \boldsymbol{\phi}^{\text{Bob}}, \boldsymbol{\mu}, \boldsymbol{Y}^{\text{Bob}}) \neq \boldsymbol{\tau}\Big).$$

We note that in our constructions, the dependence on the channel will be implicit and appear only via the use of optimal Shannon-theoretic channel codes; hence, from this point on, we will completely suppress the notation for channel inputs/outputs.

A rate R is called achievable if there exists a sequence Σ_n of length-N_n coding schemes with rates $\frac{n}{N_n} \geq R$, such that

$$\lim_{n\to\infty} \max_{\boldsymbol{\pi} \text{ of length } n} P_e(\Sigma_n, \boldsymbol{\pi}) = 0, \tag{3}$$

where the maximum is taken over all length-n interactive protocols. Accordingly, we define the interactive capacity $\mathsf{C}_{\mathsf{I}}(P_{Y|X})$ as the maximum of all achievable rates for the channel $P_{Y|X}$. Note that this definition parallels the definition of maximal error capacity in the one-way setting, as we require the error probability attained by the sequence of coding schemes to be upper bounded by a vanishing term uniformly for all protocols.

It is clear that at least n bits need to be exchanged in order to reliably simulate a general protocol, and hence the interactive capacity satisfies $\mathsf{C}_{\mathsf{I}}(P_{Y|X}) \leq 1$. In the special case of a noiseless channel, i.e., where the output deterministically reveals the input bit, and assuming that the order of speakers is predetermined (namely $\boldsymbol{\mu}$ contains only constant functions), this upper bound can be trivially achieved; Alice and Bob can simply evaluate and send τ_i sequentially according to (1) and (2). Note, however, that if the order of speakers is general, then this is not a valid solution, since we required the order of speakers in the coding scheme to be fixed in advance. Nevertheless, any general interactive protocol can be sequentially simulated using the channel $2n$ times with alternating order of speakers, where each party sends a dummy bit whenever it is not their time to speak. Conversely, a factor two blow-up in the protocol length in order to account for a non predetermined order of speakers is also necessary. To see this, consider an example of a protocol where Alice's first bit determines the identity of the speaker for the rest of time; in order to simulate this protocol using a predetermined order of speakers, it is easy to see that at least $n-1$ channel uses must be allocated to each party in advance. We conclude that under our restrictive capacity definition, the interactive capacity of a noiseless channel is exactly $\frac{1}{2}$.

When the channel is noisy, a tighter trivial upper bound holds:

$$\mathsf{C}_{\mathsf{I}}(P_{Y|X}) \leq \frac{1}{2}\mathsf{C}_{\mathsf{Sh}}(P_{Y|X}), \tag{4}$$

To see this, consider the same example given above, and note that each party must have sufficient time to reliably send $n-1$ bits over the noisy channel. Hence, the problem reduces to a pair of one-way communication problems, in which the Shannon capacity is the fundamental limit. We remark that it is reasonable to expect the bound (4) to be loose, since general interactive protocols cannot be trivially reduced to one-way communication as the parties cannot generate their part of the transcript without any interaction. However, the tightness of the bound remains a wide open question.

In the remainder of this paper we limit the discussion to protocols in which the order of speakers is predetermined and bit vs. bit. Namely, Alice speaks at odd times ($\sigma_i =$ Alice for odd i) and Bob speaks at even times ($\sigma_i =$ Bob for even i). We note that for such protocols, the $1/2$ penalty required for the adaptive order of speakers is not needed and the upper bound is therefore

$$\mathsf{C}_{\mathsf{I}}(P_{Y|X}) \leq \mathsf{C}_{\mathsf{Sh}}(P_{Y|X}).$$

Background and Related Work The interactive communication problem introduced by Schulman [3,7] is motivated by Yao's communication complexity scenario [8]. In that latter scenario, the input of a function f is distributed between Alice and Bob, who wish to compute f with negligible error by exchanging (noiseless) bits using some interactive protocol. The length of the shortest protocol achieving this is called the communication complexity of f, and denoted by $CC(f)$. In Schulman's (random) interactive communication setup, Alice and Bob must achieve their goal by communicating through a pair of independent noisy channels, where the physical model does not allow simultaneous transmissions. For that setup, Schulman showed that one can attain this goal with negligible error, using only a constant blow-up in the length of the communication.

Several works studied the interactive capacity of a binary symmetric (BSC). The binary symmetric channel with a crossover probability $0 \leq \varepsilon \leq \frac{1}{2}$ (BSC(ε)) is standardly defined by the input-to-output relation

$$Y = X \bigoplus Z$$

where $X, Y, Z \in \mathbb{F}_2$, $\bigoplus$ denotes addition over $\mathbb{F}_2$. Z is statistically independent of X with $\Pr(Z = 1) = \varepsilon$. Its Shannon capacity is known to be [9]:

$$\mathsf{C}_{\mathsf{Sh}}(\varepsilon) \triangleq 1 - h(\varepsilon),$$

where $h(\varepsilon) \triangleq -\varepsilon \log \varepsilon - (1-\varepsilon)\log(1-\varepsilon)$ is the binary entropy function, and $\log(x) \triangleq \log_2(x)$.

In [10], Kol and Raz considered the interactive communication problem, with no simultaneous transmissions over a BSC(ε). They denoted the minimal expected length of a coding scheme computing f with a negligible error probability, by $CC_\varepsilon(f)$. They then defined the corresponding interactive capacity as:

$$\mathsf{C}_{\mathsf{I}}^{\mathsf{KR}}(\varepsilon) \triangleq \lim_{n\to\infty} \min_{f:CC(f)=n} \frac{n}{CC_\varepsilon(f)}.$$

with the additional assumption that the order of speakers in the input protocol is predetermined. They proved that

$$\mathsf{C}_{\mathsf{I}}^{\mathsf{KR}}(\varepsilon) \leq 1 - \Omega\left(\sqrt{h(\varepsilon)}\right). \tag{5}$$

in the limit of $\varepsilon \to 0$. They further proved that a rate of $1 - O(\sqrt{h(\varepsilon)})$ is achievable under an additional assumption that the order of speakers in the input protocol is has a small period. The assumption on the order of speakers is crucial. Indeed, consider again the example where the function f is either Alice's input or Bob's input as decided by Alice. In this case, the communication complexity with a predetermined order of speakers is double that without this restriction, and hence considering such protocols renders $\mathsf{C}_{\mathsf{I}}^{\mathsf{KR}}(\varepsilon) \leq \frac{1}{2}$.

For a fixed nonzero ε, the coding scheme presented in [3] (which precedes [10]) implies that $\mathsf{C}_{\mathsf{I}}(\varepsilon) \geq \alpha \cdot \mathsf{C}_{\mathsf{Sh}}(\varepsilon)$ for some universal constant α, but the constant was not computed. In [11], Haeupler considered a different physical channel model where Alice and Bob can access the channel simultaneously and have three input symbols (as essentially described above). In this setup, he showed that a rate of $1 - O(\sqrt{\varepsilon})$ is achievable for any alternating

input protocol, which is higher than the upper bound (5). His results also hold in the more difficult adversrial setting assuming shared randomness, and reduces slightly to $1 - O\left(\sqrt{\varepsilon}\log\log\frac{1}{\varepsilon}\right)$ when no randomness is available. Other channel models have been addressed in the literature. Much work has been dedicated to the adversarial setting, where the channel is controlled by an adversary with some limited jamming budget, see, for example, [7,11–13].

As said before, this paper differs than previous works in the literature by not taking a worst case assumption with respect to the protocols to be simulated, but rather, restricts the simulation to a certain (yet rich) family of protocols. This restriction facilitates the construction of coding schemes whose rate is exactly Shannon's capacity of the respective channel, whereas previous coding schemes in the literature analyzed the rate at the limit of very clean channels [10,11] or up to constants [3,7].

3. Finite-State Protocols

Let us start by defining the notions of interactive rate and capacity for families of protocols. Let $\mathbf{\Pi} = \{\Pi_1, \Pi_2, ...\}$ be a sequence of families of protocols, where Π_n denotes some family of length-n protocols. A rate R is called achievable for $\mathbf{\Pi}$ if there exists a sequence Σ_n of (n, N_n) coding schemes where $N_n \leq \frac{n}{R}$, and such that

$$\lim_{n\to\infty} \max_{\pi\in\Pi_n} P_e(\Sigma_n, \boldsymbol{\pi}) = 0.$$

Namely, the difference from (3) is that now the maximum is taken over the protocols in Π_n and not over the entire family of protocols with length n. Accordingly, we denote the interactive capacity respective to the channel $P_{Y|X}$ and the family of protocols $\mathbf{\Pi}$ by $\mathsf{C}_\mathsf{I}(\mathbf{\Pi}, P_{Y|X})$, and define it as the maximum of all achievable rates for $P_{Y|X}$ and $\mathbf{\Pi}$.

The family of protocols studied in this paper is the family of finite-state protocols with M states, which will be referred to in short as M-states protocols. In these protocols, the entire history of the transcript is encapsulated in a state variable taken from a set with a finite cardinality. The state variable determines the following transcript bit, and is advanced by both parties using a predetermined update rule.

The notation of finite-state protocols is given here:

Definition 1. *Let $\mathbf{\Phi}_M = \{\Phi_{M,1}, \Phi_{M,2}, ...\}$ denote the family of M-state protocols of increasing lengths. For these protocols Alice speaks at odd times and Bob speaks on even times: namely $\sigma_i =$ Alice if i is odd, and $\sigma_i =$ Bob if i is even. The transcript of these protocols is generated by*

$$\tau_i = \psi_i(s_{i-1}), \tag{6}$$

where s_i is the state variable at time i, $s_i \in S$. S is the state-space, with cardinality $|S| = M$ assumed to be $S = \{0, 1, ..., M-1\}$ without loss of generality. $\phi_i : S \mapsto \{0, 1\}$ is the transmission function at time i, owned by Alice at odd i and by Bob at even i and assumed to be unknown to the counterpart. In addition, the state s_i is advanced in time according to

$$s_i = \eta(s_{i-1}, \tau_i), \tag{7}$$

where $\eta : (S, \{0, 1\}) \mapsto S$ is the state-advance function, which is time invariant and known to both parties.

The following example for a finite-state protocols is the family of Markovian protocols previously presented in [6] and defined as follows:

Example 1. *For a Markovian protocol, the number of states M is a power of two, and the state variable corresponds to the last* $\log M$ *bits of the transcript. Namely, the state can be regarded as the binary vector*

$$\boldsymbol{s}_{i-1} = (s_{i-1}(1), ..., s_{i-1}(\log M)) = (\tau_{i-\log M}, ..., \tau_{i-1}).$$

and the state-advance function is

$$\boldsymbol{s}_i = \eta(\boldsymbol{s}_{i-1}, \tau_i) = (s_{i-1}(2), ..., s_{i-1}(\log M - 1), \tau_i).$$

4. Basic Concepts of the Coding Schemes

The proofs in this paper are based on constructive coding schemes which use the concept of vertical simulation presented below, implemented in conjunction with either one of the two methods described in Sections 4.2 and 4.3.

4.1. Vertical Simulation

As explained before, the transcript bits of interactive protocols are produced sequentially $(\tau_1, \tau_2, \tau_3, \ldots)$. Simulating a protocol over noisy channel requires the reliable transmission of the bits sent in every round, whose number is potentially small (and can even be equal to one, in the extreme case and in the finite-state protocols discussed in this paper), which impedes the use of efficient channel codes due to finite block-length bounds [14].

One way of circumventing the problem of a short block-length (i.e., small number of bits per round) is using vertical simulation as explained in this subsection. The concept of a vertical simulation is depicted in Table 1, in which the protocol is simulated in vertical blocks, according to the indexing at the bottom row of the table. Namely, the first vertical block contains the transcript bits $(\tau_1, \tau_{m+1}, \tau_{2m+1}, \ldots)$, the second vertical block contains the transcript bits $(\tau_2, \tau_{m+2}, \tau_{2m+2}, \ldots)$ and so on. As shall be explained in the sequel, the vertical blocks can be constructed to be sufficiently long in order to allow reliable transmission at rates approaching Shannon capacity. The main obstacle of using this technique in the general case is the assumption that future transcript bits (for examplem τ_{m+1}, τ_{2m+1} etc. for the first vertical block) are known prior to the simulation of the protocol. In the sequel, we shall provide methods for the efficient computation of these future transcript bit, which facilitate the simulation of the entire protocol at Shannon capacity.

Let us now explicitly define the concept of vertical simulation. Let the n times of the protocols be divided into n/m blocks of length m, and assume that all the initial-states respective to the beginnings of all blocks (s_0, s_m, s_{2m} etc.) are known to both parties before the transcript is simulated. For simplicity of presentation, one can consider at this point that the initial states are calculated and revealed by a genie, who knows the transmission functions of both parties. More realistic methods for calculating the initial states are elaborated later in this section. We now note, that by the finite-state property in Definition 1, having the initial-states of all the blocks known, the parties can continue simulating the transcript of every block, without needing to know the transcripts of its preceding blocks. In other words, the knowledge of the initial state at every blocks decouples the simulation problems of distinct blocks.

Table 1. Vertical protocol simulation.

Block #	**Initial State**	**Transcript**				
1	s_0	τ_1	τ_2	$\cdots$	τ_{m-1}	τ_m
2	s_m	τ_{m+1}	τ_{m+2}	$\cdots$	τ_{2m-1}	τ_{2m}
3	s_{2m}	τ_{2m+1}	τ_{2m+2}	$\cdots$	τ_{3m-1}	τ_{3m}
$\vdots$	$\vdots$	$\vdots$				$\vdots$
n/m	s_{n-m}	τ_{n-m+1}	τ_{n-m+2}	$\cdots$	τ_{n-1}	τ_n
speaker		Alice	Bob	$\ldots$	Alice	Bob
vertical block #		1	2	$\cdots$	$m-1$	m

Using this decoupling assumption, the following coding scheme can be used for the simulation of the protocol over $P_{Y|X}$. We start by defining the vectors of state estimates and transcript estimates held by Alice and Bob. We use a distinct notation for every party and emphasize the fact that these are estimates, since they are computed over noisy channels. We denote the vector of initial state estimates at Alice's side for vertical block j by

$$\hat{\boldsymbol{s}}^A(j) \triangleq (\hat{s}^A_{j-1}, \hat{s}^A_{j+m-1}, \hat{s}^A_{j+2m-1}, \ldots, \hat{s}^A_{j+n-m-1}),$$

and the respective vector of transcript estimates by

$$\hat{\boldsymbol{\tau}}^A(j) \triangleq (\hat{\tau}^A_j, \hat{\tau}^A_{j+m}, \hat{\tau}^A_{j+2m} \ldots, \hat{\tau}^A_{j+n-m}).$$

Bob's counterparts to $\hat{\boldsymbol{s}}^A(j)$ and $\hat{\boldsymbol{\tau}}^A(j)$ are respectively denoted by $\hat{\boldsymbol{s}}^B(j)$ and $\hat{\boldsymbol{\tau}}^B(j)$ and are similarly defined. The scheme can now be presented for odd j from 1 to m:

1. Assume that Alice and Bob have $\hat{\boldsymbol{s}}^A(j)$ and $\hat{\boldsymbol{s}}^B(j)$.
2. Alice uses $\hat{\boldsymbol{s}}^A(j)$ to calculate $\hat{\boldsymbol{\tau}}^A(j)$ according to (6).
3. Alice encodes $\hat{\boldsymbol{\tau}}^A(j)$ using a block code with rate $R_v < \mathsf{C_{Sh}}(P_{Y|X})$, and sends it to Bob over the channel, using $\frac{n/m}{R}$ times. This code will be referred to as a vertical block code.
4. Bob decodes the output of the channel and obtains $\hat{\boldsymbol{\tau}}^B(j)$.
5. Alice (resp. Bob) uses $\hat{\boldsymbol{s}}^A(j)$ and $\hat{\boldsymbol{\tau}}^A(j)$ (resp. $\hat{\boldsymbol{s}}^B(j)$ and $\hat{\boldsymbol{\tau}}^B(j)$) to calculate $\hat{\boldsymbol{s}}^A(j+1)$ (resp. $\hat{\boldsymbol{s}}^B(j+1)$) according to (7).
6. Alice and Bob advance j by one.

For even j, the same steps are implemented by exchanging the roles of Alice and Bob. We recall that we previously assumed that for the first block, both parties know the actual initial states of the noiseless protocol, i.e., $\hat{\boldsymbol{s}}^A(1) = \hat{\boldsymbol{s}}^B(1)$ and both are equal to the state vector of the noiseless protocol. It is clear from the construction of the scheme, that if all block codes are reliably decoded, the transcript is simulated without error. The following basic lemma gives a condition for the reliable decoding of block codes:

Lemma 1. *Suppose $l(n)$ independent blocks of $b(n)$ bits are to be conveyed over channel $P_{Y|X}$ at rate $R < \mathsf{C_{Sh}}(P_{Y|X})$ and $n \to \infty$. Then, if $l(n) = o(e^{b(n)})$, the probability of error in the decoding of one or more blocks is $o(1)$.*

The proof is due to the basic fact that the probability of error decays exponentially in the block length and appears in Appendix A.

From this point on, we set $m = \sqrt{n}$. We assume that if needed, the transcript is extended by zeros in order to ensure that $\sqrt{n}$ is an integer. Using Lemma 1 with $l(n) = m(n) = \sqrt{n}$ ensures that reliable transmission of the vertical blocks can be accomplished at any rate $R_v < \mathsf{C_{Sh}}(P_{Y|X})$.

Let us now bound the total length N of the simulating protocol:

$$N = \frac{n}{m}\frac{m}{R_v} = \frac{n}{R_v}.$$

Therefore $\frac{n}{N} = R_v$ for every $R_v < \mathsf{C_{Sh}}(P_{Y|X})$, which means that the protocol can be reliably simulated at Shannon capacity if $n \to \infty$.

So far, we assumed without justification, that initial states of all the blocks were revealed to both parties before the beginning of the simulation. We now present two alternative methods for their efficient calculation.

4.2. Efficient State Lookahead

This method is based on two assumptions:

1. For every block, the last state can be calculated by both parties given the first state, without knowing the entire transcript of the block, using only $o(m)$ (clean) bits exchanged between the parties.
2. The $\frac{n}{m}o(m)$ bits required for this calculation for the entire protocol, can be reliably exchanged over the noisy channels at a strictly positive rate.

Assuming that the very first state of the protocol is known to both parties, and that the first condition holds, Alice and Bob can go from the first block to the last and calculate all their respective initial states. The second condition guarantees that only additional $\Theta(\frac{n}{m}o(m)) = o(n)$ channel uses are required for this process. The total length of the simulating protocol can thus be bounded by

$$N \leq \frac{n}{m}\frac{m}{R_v} + o(n),$$

so, as before, $\lim_{n\to\infty}\frac{n}{N} = R_v$ for every $R_v < \mathsf{C_{Sh}}(P_{Y|X})$, which means that the protocol can be reliably simulated at Shannon capacity provided that $n \to \infty$.

4.3. Efficient Exhaustive Simulation

The following method was previously presented in [6] for the simulation of Markovian protocols. So far, we assumed that for every block, only the transcript related to single initial state, which was assumed to be the actual state in the noiseless protocol, was simulated. Alternatively, it is possible to simulate all transcripts resulting from all possible initial states in every block, and then go from the first block to the last and estimate the transcript of the noiseless protocol according the the final state of the previous block. Such a simulation can be made possible, for example, if the parties simply describe the identities of their transmission functions to their counterparts. While it is easy to show that the required bits can be conveyed at Shannon capacity, if there are more than two possible transmission functions at every time, the total rate of such a coding scheme is bound to be lower than Shannon capacity.

However, Shannon capacity can be achieved if the following conditions hold:

1. At every block, the transcripts associated with all possible M initial states, can be encoded using only $m + o(m)$ bits.
2. The required bits can be reliably conveyed over the noisy channels at any rate below Shannon capacity.

If both conditions hold then the total number of channel usese required for the simulation is

$$N \leq \frac{n}{m}\frac{m + o(m)}{R_v},$$

and the protocol can be simulated at any rate below Shannon capacity as long as $n \to \infty$.

5. Achieving Shannon Capacity with Two States

The first result presented in this paper is that any two-state protocol can be simulated at Shannon capacity. An equivalent statement is given in the following theorem:

Theorem 1. *Let $\mathbf{\Phi}_2$ be the family of two-state protocols. Then*

$$\mathsf{C_I}(\mathbf{\Phi}_2, P_{Y|X}) = \mathsf{C_{Sh}}(P_{Y|X}),$$

The proof is based on the following coding scheme:

Proof. We assume without loss of generality that $S = \{0, 1\}$ and start by presenting an algorithm for the efficient state lookahead method from Section 4.2 For simplicity of exposition we use the time indices of the first block. For other blocks the indices should be appropriately shifted. We also assume that the bits required for the algorithm are

exchanged between Alice and Bob without error. In the sequel we explain how they can be reliably conveyed over the noisy channels.

The first step in the algorithm is the calculation of the following sequence of composite functions, $\nu_i : S \mapsto S$, defined as:

$$\nu_i(s_{i-1}) \triangleq \eta(s_{i-1}, \psi_i(s_{i-1})), \tag{8}$$

for $1 \leq i \leq m$, which is done by Alice at odd i and Bob at even i. We note that knowing $\nu_i(s_{i-1})$, and the value of s_{i-1}, the following state s_i can be calculated. We also note that since $\nu_i : \{0,1\} \mapsto \{0,1\}$, $\nu_i(s_{i-1})$ must be one of the following four functions:

$$\nu_i(s_{i-1}) = s_{i-1} \bigoplus 0, \quad \nu_i(s_{i-1}) = s_{i-1} \bigoplus 1, \quad \nu_i(s_{i-1}) = 0, \quad \nu_i(s_{i-1}) = 1,$$

which can also be described in the following form:

$$\nu_i(s_{i-1}) = s_{i-1} \bigoplus c_i \text{ or } \nu_i(s_{i-1}) = b_i, \tag{9}$$

where $b_i, c_i \in \{0,1\}$. The basic idea of the algorithm is the following. If for all $1 \leq i \leq m$ the composite functions are $\nu_i(s_{i-1}) = s_{i-1} \bigoplus c_i$, then the final state s_m can be calculated by:

$$\begin{aligned} s_m &= s_0 \bigoplus \left[\bigoplus_{i=1}^{m} c_i \right] \\ &= s_0 \bigoplus d_{\text{Alice}} \bigoplus d_{\text{Bob}} \end{aligned} \tag{10}$$

where

$$\begin{aligned} d_{\text{Alice}} &\triangleq \bigoplus_{i \text{ is odd}, i \in \{1,\ldots,m\}} c_i \\ d_{\text{Bob}} &\triangleq \bigoplus_{i \text{ is even}, i \in \{1,\ldots,m\}} c_i. \end{aligned}$$

In other words, s_m can be calculated by its initial value s_0 and the parity of the number of times in the block it is flipped (from 0 to 1 or vice versa) by either Alice or Bob. All in all, assuming that the parties know s_0, they only need to exchange d_{Alice} and d_{Bob} (i.e., two bits) in order to calculate s_m. However, so far we assumed that all the compsite functions in the block in the following form $\nu_i(s_{i-1}) = s_{i-1} \bigoplus c_i$. In the general case in which $\nu_i(\cdot)$ are taken from the complete set of four functions in (9), the algorithm can be modified by first exchanging the location and the value of the last constant composite function in the block, i.e., the last composite function of the form $\nu_i(s_{i-1}) = b_i$. We note that this process requires only exchanging $O(\log m)$ between Alice and Bob. Then, s_m can be calculated similarly to (10) but from the location of the last constant composite function and not from the beginning of the block.

The algorithm is formulated as follows:

1. Alice sends Bob her latest (odd) time index in the block for which $\nu_i(s_{i-1}) = b_i$, $b_i \in \{0,1\}$ (i.e., her latest constant composite function), along with value of b_i. If such an index does not exist she sends zero to Bob. Bob then repeats the same process with the appropriate alterations. We use i_{const} to denote the maximum of the indices, which therefore represents the location of the last constant composite function in the block. We now set $b_0 = s_0$ if $i_{\text{const}} = 0$ and $b_{i_{\text{const}}}$ if $i_{\text{const}} > 0$. This process requires exchanging $O(\log m)$ bits between Alice and Bob.

2. We now note, that since i_{const} is the index of the latest constant composite function in the block, then for all $i_{\text{const}} < i \leq m$, $\nu_i(s_{i-1}) = s_{i-1} \bigoplus c_i$ for some $c_i \in \{0,1\}$. The final state in the block, s_m, can therefore be calculated by

$$\begin{aligned} s_m &= b_{i_{\text{const}}} \bigoplus \bigoplus_{i=i_{\text{const}}+1}^{m} c_i \\ &= b_{i_{\text{const}}} \bigoplus d_{\text{Alice}} \bigoplus d_{\text{Bob}} \end{aligned} \tag{11}$$

where

$$\begin{aligned} d_{\text{Alice}} &\triangleq \bigoplus_{i \text{ is odd}, i \in \{i_{\text{const}}+1,\ldots,m\}} c_i \\ d_{\text{Bob}} &\triangleq \bigoplus_{i \text{ is even}, i \in \{i_{\text{const}}+1,\ldots,m\}} c_i. \end{aligned}$$

We finally note, that d_{Alice} and d_{Bob} are single bits that can be calculated by their respective parties and then exchanged, leaving the total number of required exchanged bits for the algorithm $O(\log m)$.

After repeating this operation for all blocks, it is possible to calculate all the final states of all blocks (i.e., all the initial states of their following blocks) by applying (11) from the first block to the last.

It only remains to verify that the respective $O(\log m)$ bits per block can be reliably conveyed over the noisy channels between Alice and Bob using the channel times $o(n)$ times as required in Section 4.2. This task can be easily performed, for example by using one block code per party containing $O(\frac{n}{m} \log m) = O(\sqrt{n} \log n)$ bits. □

For the sake of completeness we now give the high level of an alternative coding scheme based on the efficient exhaustive simulation method described in Section 4.3. This coding scheme is a little more involved than the previously described one, and depends on the identity of the state-advance function $\eta(\cdot)$. We start by noting that $\eta(\cdot)$ is a binary function with two binary inputs, so there are in total sixteen possible such functions. In particular, there are four state-advance function that do not depend on transcript bit τ_i:

$$\eta(s_{i-1}, \tau_i) = 0 \bigoplus s_{i-1}, \quad \eta(s_{i-1}, \tau_i) = 1 \bigoplus s_{i-1}, \quad \eta(s_{i-1}, \tau_i) = 0, \quad \eta(s_{i-1}, \tau_i) = 1.$$

As the very first state of the protocol is assumed to be known to both parties, having one of these state-advance functions, the state sequence of the entire protocol can be determined before its simulation, rendering the entire protocol non-interactive, and hence trivial to simulate. For the remaining twelve state-advance functions, the following coding scheme is proposed, which is described for simplicity for the first block, but should be independently implemented for all blocks:

1. Before the simulation begins, both parties communicate the locations of the first (rather than the last) constant composite function in the block: the smallest value $1 \leq i \leq j$ for which $\nu_i(s_{i-1}) = b_i$, for some $b_i \in \{0,1\}$. This process requires exchanging $O(\log m)$ bits.
2. The parties exchange the identities of their transmission functions (i.e., $\psi i(\cdot)$) before the location of the first constant composite function in the block, using a single bit per time index. In the sequel we show that there are indeed only two relevant functions to describe, so their description requires only a single bit. At the end of this process, the parties can independently simulate the transcripts for both initial states until the location of the first constant composite function.

3. For time indices after the location of the first constant composite function, the transcripts associated with both initial states coincide, so they can both be simulated using a single bit per time index.

Using this coding scheme, only $m + o(m)$ bits are required for the simulation of the transcripts associated with both initial states. These bits can be reliably conveyed for all blocks using vertical block codes, as required in the description of the scheme in Section 4.3.

To see that, observe that there are only three canonical types of state-advance functions, depicted in the state-diagrams in Figure 1. The nodes represent the state variables, and the directed edges show the possible state transitions. The specific values of the states and transcript bits on the edges are deliberately not indicated; it is easy to check that there are four possible setting for every type, summing up to twelve functions in total. An example for a Type I state-advance function is $\eta(s_{i-1}, \tau_i) = \tau_i$, for a Type II state-advance function is $\eta(s_{i-1}, \tau_i) = s_{i-1} \wedge \tau_i$, and for a Type III state-advance function is: $\eta(s_{i-1}, \tau_i) = 1 \bigoplus (s_{i-1} \wedge \tau_i)$. We now return to the definition of the composite functions $\nu_i(s_{i-1})$ in (8), and note that $\nu_i(s_{i-1})$ is constant (i.e., set to either 0 or 1) if the transmission functions are such that s_i receives the same value for both $s_{i-1} = 0$ and $s_{i-1} = 1$. As the transmission function $\psi_i(s_{i-1})$ determines the values associated with the edges of the state diagram, it can be seen that for every type of advance function, there exist only two transmission functions which render $\nu_i(s_i)$ constant. Since there are in total four possible transmission function, there are therefore only two possible transmission functions before the appearance of one of the two that makes $\nu_i(s_i)$ constant, as required by the scheme.

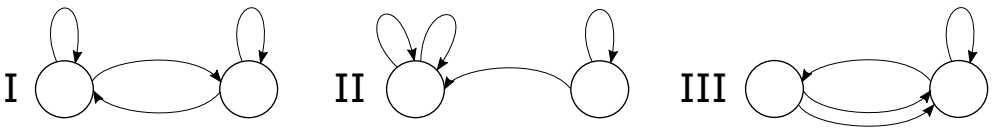

Figure 1. State diagrams of the three types of state-advance functions.

6. Failure of the Coding Scheme for Three States

We now provide an example of a protocol for which both methods described in Sections 4.2 and 4.3 fail. Since the protocol is to be used on a block (rather than on the entire protocol), we use m to denote its length.

Example 2. *We define the following interactive three-state protocol ($S = \{0, 1, 2\}$) of length m. The state-advance rule $s_i = \eta(s_{i-1}, \tau_i)$ is depicted in Figure 2. Namely, at state $s_{i-1} = 0$, the next state is $s_i = \tau_i$. At state $s_{i-1} = 1$ the next state is $s_i = 0$ if $\tau_i = 0$ and $s_i = 2$ if $\tau_i = 1$. At state $s_{i-1} = 2$, the next state is $s_i = 2$ regardless the value of τ_i.*

Let $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, ...\alpha_m)$ and $\boldsymbol{\beta} = (\beta_1, \beta_2, ...\beta_m)$ be binary sequences. Assume Alice knows the elements of both sequences only at odd indices, and Bob knows the elements of both sequences only at even indices. The following transmission function is used by Alice at odd time indices:

$$\tau_i = \psi_i(s_{i-1}) = \begin{cases} \alpha_i & \text{if } s_{i-1} \in \{0, 1\} \\ \beta_i & \text{if } s_{i-1} = 2, \end{cases}$$

and the following transmission function is used by Bob at even time indices:

$$\tau_i = \psi_i(s_{i-1}) = \begin{cases} \alpha_i \wedge s_{i-1} & \text{if } s_{i-1} \in \{0, 1\} \\ \beta_i & \text{if } s_{i-1} = 2. \end{cases}$$

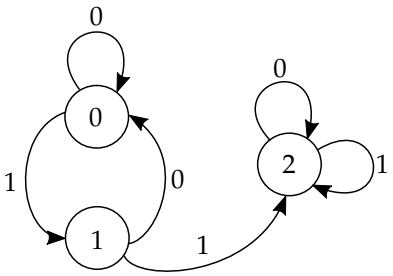

Figure 2. A diagram representing the state-advance function $\eta(s_{i-1}, \tau_i)$ of the three-state protocol from Section 6. The nodes represent the states, and an edge (s_{i-1}, s_i) represents a transition from state s_{i-1} to s_i. The numbers attached to the edges are the transcript bit $\tau_i = \psi_i(s_{i-1})$.

We start by proving the failure of the efficient state lookahead scheme from Section 4.2, by showing a reduction from the disjointness problem commonly used in the communication complexity literature [15].

Definition 2 (Disjointness). *Alice and Bob are given as input the sets* $X, Y \subseteq \{1, \ldots, m/2\}$*, respectively. The disjointness function is defined as*

$$\mathrm{DISJ}(X, Y) \triangleq \mathbb{1}(X \cap Y = \emptyset),$$

where $\mathbb{1}(\cdot)$ *is the indicator function, which equals one if the condition is satisfied and zero otherwise.*

We now show how $\mathrm{DISJ}(X, Y)$ can be computed using the three-state protocol of Example 2. We set the values of the vector $\boldsymbol{\alpha}$ for $k \in \{1, 2, \ldots, m/2\}$ according to

$$\alpha_{2k-1} = \mathbb{1}(k \in X), \quad \alpha_{2k} = \mathbb{1}(k \in Y).$$

The values of the elements of $\boldsymbol{\beta}$ do not affect the reduction from disjointness and can be all set to zero for simplicity. They will be used in the proof of the failure of the exhaustive simulation scheme shown in the sequel.

Observe that $s_m = 2$ if and only if there exist at least one $k \in \{1, 2, \ldots, m/2\}$ for which $\alpha_{2k-1} = 1$ and $\alpha_{2k} = 1$, which means that $k \in X$ and $k \in Y$ and the intersection of X and Y is not empty. Namely,

$$\mathrm{DISJ}(X, Y) = \mathbb{1}(s_m \in \{0, 1\}), \tag{12}$$

which means that s_m can be used to compute $\mathrm{DISJ}(X, Y)$.

Since it is assumed that s_m can be computed using $o(m)$ bits, and $\mathrm{DISJ}(X, Y)$ can be computed using s_m without additional communication due to (12), it follows that $\mathrm{DISJ}(X, Y)$ can also be computed using $o(m)$ bits. However, it is well-known that the communication complexity of the disjointness function is $\Omega(m)$: even if Alice and Bob can use a shared randomness source in their communication protocol, and even if they are allowed to err with probability $1/3$, they must still exchange $\Omega(m)$ bits in the worst-case in order to compute $\mathrm{DISJ}(X, Y)$ [16,17]. In fact, disjointness remains hard even in the amortized case, where Alice and Bob are given a sequence of inputs $X_1, \ldots, X_l \subseteq \{1, \ldots, m/2\}$ and $Y_1, \ldots, Y_l \subseteq \{1, \ldots, m/2\}$ (respectively), and their goal is to output the sequence $\mathrm{DISJ}(X_1, Y_1), \ldots, \mathrm{DISJ}(X_l, Y_l)$. The average communication per-copy for this task is still $\Omega(m)$ (i.e., the total communication is $\Omega(m \cdot l$, where l is the number of copies)). This result is the direct consequence of the following three results: (i) the information cost of disjointness is linear [18]; (ii) information cost is additive [19]; and (iii) information is a lower bound on communication [18].

We now prove that for Example 2, the efficient exhaustive simulation of Section 4.3 also fails, by providing a setting of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ for which simulating the transcripts of all three possible initial states requires the parties to reliably exchange $\frac{3}{2}m$ bits. This is impossible to

accomplish using only $m + o(m)$ exchanged bits, as assumed in the scheme, and therefore the scheme must fail. We set up the example as follows: we set $\boldsymbol{\beta}$ to be an arbitrary binary vector whose odd elements are known only to Alice and whose even elements are known only to Bob. In addition, we set the odd elements of $\boldsymbol{\alpha}$ to be arbitrary and known only to Alice, and set all the even elements of $\boldsymbol{\alpha}$ to zero. We observe that the transcript associated with the initial state $s_0 = 2$ essentially sends the sequence $\beta_1, \ldots, \beta_m$ non-interactively, implying that the parties exchange m bits that are initially unknown to their counterparts. For the other two initial states, $s_0 \in \{0, 1\}$, the setting of the even entries of α to zero guarantees that the transcripts associated with the two initial states $s_0 \in \{0, 1\}$ will never reach the state $s_i = 2$ at $i \leq m$. This way, in order to simulate the associated transcripts, Alice must convey to Bob the even entries of $\boldsymbol{\alpha}$. Hence, successful exhaustive simulation means that $\frac{3}{2}m$ bits must be exchanged overall, which cannot be done using only $m + o(m)$ bits.

We note that the underlying reason for the failure of our methods in this specific protocols, is the existence of the "stuck state" 2. Indeed, as we will shortly see, a strong connectivity condition of being able to reach the same state from any two states at the same time, will prove crucial for the achievability of the Shannon capacity in finite-state protocols over large state spaces (see the notion of coinciding state-advance functions in Definition 3). It is therefore interesting to ask whether there exists a fully connected finite-state protocol with a small number of states, for which our methods fail as well.

7. Achieving Shannon Capacity with More than Two States

In the previous sections, we presented a coding scheme that achieves capacity for all two-state protocols, but fails to achieve capacity for at least one three-state protocol. In this section, we present specific families of M-state protocols which obey two conditions, and show that within these families, almost all protocols can be simulated at Shannon capacity. The notion of achieving capacity for almost all members of a family is demonstrated in the following example:

Example 3. *Consider the family of Markovian protocols with M states defined in Example 1 where M is a power of two, and whose transmission functions are taken from the entire set of $S \mapsto \{0, 1\}$ functions. We shall now show that capacity is achievable for almost all protocols in this family.*

To see this, first observe that the set of possible transmission functions contains two constant functions: one that maps all states to 0 and one function that maps all states to 1. Now, assume that vertical simulation is implemented as described in Section 4, but all transcripts for initial states in all blocks are simulated for the last $n^{1/4}$ times in every block (which requires only $o(n)$) channel uses. It is easy to show (and a stronger statement is proved in Theorem 2 below) that almost all protocols in the family have at least one sequence of $\log M$ constant functions within the last $n^{1/4}$ times in every transmission block. Having this sequence of constant functions will ensure that all transcripts in every block will have the same final state, which could be used for the efficient state lookahead method described in Section 4.2.

The example above can be easily generalized to the finite-state case, where the existence of a "reset" to a constant state can be exploited similarly to decouple the transcript and attain the Shannon capacity. However, one might argue that the presence of constant transmission functions/resets is not a realistic assumption in interactive protocols. Nevertheless, it was previously shown in [6] for the Markovian setup, that the scheme described above can be used for protocols whose transmission functions are taken from smaller families of non-constant functions, such as the family of balanced Boolean functions. We shall now extend the results from [6] to finite-state protocols. For this purpose, we define two conditions that the family of protocols should fulfill. The first condition is related to the state-advance function, and the second condition is related to the transmission functions.

Definition 3. *A state-advance function η of an M-state protocol is called "coinciding" if for any pair of states $j, j' \in S$, there exists a pair of binary sequences $(b_1^j, b_2^j, \ldots, b_K^j)$ and $(b_1^{j'}, b_2^{j'}, \ldots, b_K^{j'})$, for some $K \in \mathbb{N}$, for which $\tilde{s}_K^j = \tilde{s}_K^{j'}$ where $\tilde{s}_K^j$ is generated by applying*

$$\tilde{s}_i^j = \eta(\tilde{s}_{i-1}^j, b_i^j),$$

for i going from 1 to K with the initial condition $\tilde{s}_0^j = j$, and where $\tilde{s}_K^{j'}$ is generated similarly, replacing j by j'.

Note that without loss of generality, we can assume that K is the same for all pairs of states in the definition above, by taking it to be the maximum. Note further that any machine that performs a "reset" is coinciding, but not vice-versa.

Definition 4. *A set $\mathcal{F}$ of M-state transmission functions $S \mapsto \{0,1\}$ is called "useful" if for every pair of distinct states $s, s' \in S$, $s \neq s'$ there exists at least one set of four functions $\{f^{00}, f^{01}, f^{10}, f^{11}\} \subseteq \mathcal{F}$ for which*

$$f^{tt'}(s) = t \text{ and } f^{tt'}(s') = t'$$

for all pairs $(t, t') \in \{0,1\}^2$.

The following theorem formalizes the notion of achieving capacity for almost all members of these families of protocols:

Theorem 2. *Let η be an M-state coinciding state-advance function, and let $\mathcal{F}$ be a useful set of M-state transmission functions. Let Π_n be the family of all length-n, M-state protocols, whose state-advance function is η and whose transmission functions are taken from $\mathcal{F}$. Then there exists a sequence of families of protocols $\boldsymbol{S} = \{S_1, S_2, \ldots\}$, $S_n \subseteq \Pi_n$ and $|S_n|/|\Pi_n| = 1 - o(1)$, for which the interactive capacity is equal to the Shannon capacity. Namely,*

$$\mathsf{C}_{\mathsf{I}}(\boldsymbol{S}, P_{Y|X}) = \mathsf{C}_{\mathsf{Sh}}(P_{Y|X}).$$

Proof. The proof is based on implementing the methods from Section 4.2 or Section 4.3 using one of the following two constructions. We start by presenting the construction for the efficient state lookahead method from Section 4.2: for the last $p = n^{1/4}$ times in every transmission block, exhaustively simulate the transcripts related to all possible M initial states. We assume for simplicity that the protocol is extended by zeros so that $n^{1/4}$ is an integer and in addition, so that $n^{1/4}/K$ is also an integer, as shall be required in the sequel. This simulation can be implemented by each side describing all its respective $p/2$ transmission functions to its counterpart. After this is done, both parties can simulate the transcripts for all possible initial states in the last p times in every block without any additional channel uses. Since there are only 2^M functions $S \mapsto \{0,1\}$, the description of every function in $\mathcal{F}$ requires at most M bits. The bits required for the description of all transmission functions of a party, for the last p times in all n/m transmission blocks, can be reliably conveyed over the noisy channel, either by a single block code per party, or by a distinct block code per time instance. It is easy to see that the setting of $p = n^{1/4}$ and $m = n^{1/2}$ ensures the transmission of these bits with a vanishing error using the channels only $o(n)$ times. Now, if in every block, the transcripts respective to all possible initial state have the same (possibly block dependent) final state, we can use this set of states as the state lookahead. We call this phenomenon state-coincidence and note that if it occurs, since the channel was used only $o(n)$ times for the calculation of the state lookahead, Shannon capacity can be achieved, as explained in Section 4.2.

Alternatively, the efficient exhaustive simulation described in Section 4.3 can be similarly implemented by using the construction for the first (rather than the last) p times

in all blocks. If all states coincide in all blocks, then for every block there is only a single transcript to simulate for the last $m-p$ times in the block. All in all, only $m+o(m)$ bits are required for the simulation the transcripts of all the initial states, as required by the method.

We now use $S_n \subseteq \Pi_n$ to denote the subset of protocols for which the states coincide, so their respective transcripts can be simulated at Shannon capacity as explained above. It remains to prove that $|S_n|/|\Pi_n| = 1-o(1)$. This is done by assuming that the protocols in S_n are generated by drawing all their transmission function uniformly from the set $\mathcal{F}$ and independently in time, and denoting the probability of drawing a protocol in S_n by $\Pr(S_n)$, so

$$\Pr(S_n) = \frac{|S_n|}{|\mathcal{F}|^n} = \frac{|S_n|}{|\Pi_n|}.$$

Hence, proving that $\Pr(S_n) = 1-o(1)$ will prove the statement in the theorem.

We now show that indeed, the assumptions in the theorem ensure that $\Pr(S_n) = 1-o(1)$. We start by analyzing the probability of state-coincidence respective to a specific small block of length p. For convenience, we assume that its time indices are 1 to p. We start by denoting the transcript related to the initial state $j \in S$ in by $(\tau_1^j, \tau_2^j, \ldots, \tau_p^j)$ and the respective sequence of states by $(s_1^j, s_2^j, \ldots, s_p^j)$. More explicitly, $(\tau_1^j, \tau_2^j, \ldots, \tau_p^j)$ and $(s_1^j, s_2^j, \ldots, s_p^j)$ are generated by the following iteration of (6), (7):

$$\tau_i^j = \psi_i(s_{i-1}^j), \tag{13}$$

$$s_i^j = \eta(s_{i-1}, \tau_i^j), \tag{14}$$

for i going from $i=1$ to $i=p$ with the initial condition $s_0^j = j$. We similarly define the transcript and the sequence of state respective to the initial state $j' \neq j$, $j' \in S$ by $(\tau_1^{j'}, \tau_2^{j'}, \ldots, \tau_p^{j'})$ and $(s_1^{j'}, s_2^{j'}, \ldots, s_p^{j'})$. We shall now bound the probability of state-coincidence: $\Pr(s_p^j = s_p^{j'})$.

Now, by the assumption that the state-advance function is coinciding (Definition 3), there exist two binary sequences of $(b_1^j, b_2^j, \ldots, b_K^j)$ and $(b_1^{j'}, b_2^{j'}, \ldots, b_K^{j'})$ for which $\tilde{s}_K^j = \tilde{s}_K^{j'}$ (the tilde in the notation of $\tilde{s}_i^j$ and $\tilde{s}_i^{j'}$ is used to distinguish them from s_i^j and $s_i^{j'}$). The distinction is required, since $\tilde{s}_i^j$ and $\tilde{s}_i^{j'}$ are created by specific binary vectors, which in the general might not correspond to transcripts of protocols in Π_n. We shall now use the coincidence and usefulness properties of the family of protocols in order to bound the probability of drawing a sequence of transmission functions, $\psi_1, \psi_2, \ldots, \psi_K$, for which the iteration in (13), (14) yields $s_K^j = s_K^{j'}$. We denote by k the smallest time index for which $\tilde{s}_k^j = \tilde{s}_k^{j'}$. It follows that for every $1 \leq i \leq k$ we have $\tilde{s}_k^j \neq \tilde{s}_k^{j'}$. We now show that by the usefulness assumption, the binary sequences $(b_1^j, b_2^j, \ldots, b_k^j)$, $(b_1^{j'}, b_2^{j'}, \ldots, b_k^{j'})$ which generated the state sequences $(\tilde{s}_1^j, \tilde{s}_2^j, \ldots, \tilde{s}_k^j)$, $(\tilde{s}_1^{j'}, \tilde{s}_2^{j'}, \ldots, \tilde{s}_k^{j'})$ can also be produced by a sequence of transmission functions $(\psi, \psi_2, \ldots, \psi_k)$ drawn uniformly and independently from $\mathcal{F}$. The proof follows by observing that by the usefulness assumption that for every pair $(b_i^j, b_i^{j'})$ for $1 \leq i \leq k$, there exists at least one function $\psi_i \in \mathcal{F}$ such that

$$\psi_i(s) = b_i^j, \text{ and} \tag{15}$$

$$\psi_i(s') = b_i^{j'} \tag{16}$$

for every $s, s' \in S$, $s \neq s'$. In particular (15) and (16) also hold for the states in the sequences $(\tilde{s}_1^j, \tilde{s}_2^j, \ldots, \tilde{s}_k^j)$, $(\tilde{s}_1^{j'}, \tilde{s}_2^{j'}, \ldots, \tilde{s}_k^{j'})$. Therefore, there exists a sequence of transmission functions $(\psi, \psi_2, \ldots, \psi_k)$, with $s_k^j = s_k^{j'}$ drawn with probability

$$\Pr\left(s_k^j = s_k^{j'}\right) \geq |\mathcal{F}|^{-k}.$$

We now note that due to (13) and (14), for every $k < i \leq K$, we have that $s_i^j = s_i^{j'}$ for every choice of transmission functions $(\psi_{k+1}, \ldots, \psi_K)$ and particularly for $s_K^j = s_K^{j'}$. Therefore,

$$\Pr\left(s_K^j = s_K^{j'}\right) \geq |\mathcal{F}|^{-k} \geq |\mathcal{F}|^{-K}. \tag{17}$$

We now observe, that (17) only assumed that the initial states are distinct, i.e., $s^j = j \neq s^{j'} = j'$. Therefore, in case $s_K^j \neq s_K^{j'}$ we can consider the drawing of the following K functions as a repeated, statistically identical and independent experiment. Following this argumentation, we can consider consecutive p/K such experiments, and observe that the a failure in the coincidence at the end of the block of length p, $s_p^j \neq s_p^{j'}$, implies that all these p/K experiments failed. We can therefore state the following bound:

$$\begin{aligned}
\Pr\left(s_p^j \neq s_p^{j'}\right) &\leq \left[\Pr\left(s_K^j \neq s_K^{j'}\right)\right]^{p/K} && (18)\\
&\leq (1 - |\mathcal{F}|^{-K})^{p/K} && (19)\\
&\leq \exp\left[-|\mathcal{F}|^{-K} p/K\right], && (20)
\end{aligned}$$

where (18) is potentially loose since it considers only the coincidence events occurring in non-overlapping blocks of length K, (19) is due to (17), and finally is by the inequality $(1-x)^a \leq \exp(-ax)$ which holds for any $x > 0$ and $a \in \mathbb{N}$.

We emphasize that so far we examined the coincidence of only two initial states, j and j' in a single transmission block. We denote by $\mathcal{E}_1$ the event in which all the transcripts corresponding to all initial states did not coincide to the same final state. The probability of $\mathcal{E}_1$ can be bounded by:

$$\begin{aligned}
\Pr(\mathcal{E}_1) &= \Pr\left(\bigcup_{j=1}^{M-1} s_p^0 \neq s_p^j\right)\\
&\leq (M-1)\exp\left[-|\mathcal{F}|^{-K} p/K\right] && (21)\\
&< M\exp\left[-|\mathcal{F}|^{-K} p/K\right]
\end{aligned}$$

where in (21) we used the union bound and (20). Finally, we denote by $\mathcal{E}_2$ the event that the final states did not coincide in all transmission blocks. Using the union bound again, this probability can be bounded by:

$$\begin{aligned}
\Pr(\mathcal{E}_2) &\leq n/m \Pr(\mathcal{E}_1)\\
&\leq \sqrt{n} M \exp\left[-|\mathcal{F}|^{-K} n^{1/4}/K\right]\\
&= o(1).
\end{aligned}$$

It now immediately follows that:

$$\begin{aligned}\frac{|\mathcal{S}_n|}{|\Pi_n|} &= \Pr(S_n)\\ &= 1-\Pr(\mathcal{E}_2)\\ &= 1-o(1)\end{aligned}$$

which concludes the proof. □

8. Concluding Remarks

In this paper, the problem of simulating an interactive protocol over a pair of binary-input noisy channels is considered. While previous works [3,4,7,10] approach this problem using worst-case assumptions (characterizing the rates in which all possible interactive protocols can be simulated), this work restricts the discussion to a specific set of finite-state protocols.

The proofs in this paper are based on novel coding schemes designated to finite-state protocols, and based on concepts described in Section 4. The main concept is vertical simulation, namely, breaking the protocol into block and simulating them in parallel over the noisy channels using Shannon capacity achieving coded. This method cannot be used for general interactive protocols as their transcript in any specific block can depend on its preceding ones. However, for finite-state protocols we present methods that decouple the simulation of different blocks and facilitate their parallel simulation. We show that these methods are applicable to all two-state protocols, but break down for at least a single three-state protocol. For larger state-spaces, we take a different approach, characterizing families of finite state protocols of whose almost all members can be reliably simulated at Shannon's capacity. While it is easy to show that family comprising all finite state protocol is such a family, this property is simply due a simplifying the assumption that it comprises constant functions, which causes the simulation to decouple. For this reason, we define smaller families of finite state protocols that are highly interactive and contain no constant functions. For these families, we prove that the Shannon capacity is achievable for almost all members. We also note that some of the methods used in this paper can extended to larger families of protocols. For example, the assumption of a constant cardinality of the state-space is restrictive, as most of the proofs in Section 7 still hold while a allowing a certain suitable growth in the number of states with respect to the protocol length.

Since the proofs in this paper are based on specific coding schemes, proving their failure does not prove the inachievability of Shannon capacity. It is also plausible that Shannon capacity is achievable for larger classes of nontrivial interactive protocols using different coding scheme. A nontrivial upper bound on the ratio between the Shannon capacity and the interactive capacity for a fixed channel (i.e., not in the limit of a very clean channel) still remains an intriguing open question even in the simplest binary symmetric case.

Author Contributions: Conceptualization, A.B.-Y., Y.-H. K., and O.S.; methodology, A.B.-Y., Y.-H. K. and O.S.; software, A.B.-Y.; formal analysis, A.B.-Y. and O.S.; investigation, A.B.-Y., Y.-H. K., R.O. and O.S.; writing—original draft preparation, A.B.-Y., Y.-H. K., R.O. and O.S; writing—review and editing, A.B.-Y., Y.-H. K., R.O. and O.S. All authors have read and agreed to the published version of the manuscript.

Funding: The work of O. Shayevitz was supported by the ERC under Grant 639573 and an ISF grant no. 1495/18. The work of A. Ben-Yishai was supported by ISF grant no. 1367/14.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not avaliable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Proof of Lemma 1

Proof. The proof is by straightforward implementation of Gallager's random coding error exponent and the union bound. Due to [20] [Theorem 5.6.4], the probability of decoding error in a single block is upper bounded by:

$$\Pr(\text{block error}) \leq \exp\left(-\frac{b(n)}{R}E_r(R)\right)$$

where $E_r(R)$ (the error exponent) is strictly positive for any $0 \leq R < \mathsf{C_{Sh}}(P_{Y|X})$ and $b(n)/R$ is the length of the block code. Now, having $l(n)$ independent such blocks, the probability of error in one or more blocks can be upper bounded using the union bound:

$$\begin{aligned}\Pr(\text{error in any block}) &\leq l(n)\exp\left(-\frac{b(n)}{R}E_r(R)\right)\\ &= \exp\left(-b(n)\frac{E_r(R)}{R} + \ln l(n)\right) \overset{(a)}{=} e^{-\Omega(1)} = o(1)\end{aligned}$$

where (a) is by the assumption that $l(n) = o(e^{b(n)})$. □

References

1. Shannon, C.E. Two-way communication channels. In Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics, the Regents of the University of California, Oakland, CA, USA, 20–30 July 1960; University of California Press: Berkeley, CA, USA, 1961.
2. Gelles, R. Coding for Interactive Communication: A Survey. 2019. Available online: http://www.eng.biu.ac.il/~gellesr/survey.pdf (accessed on 24 December 2020).
3. Schulman, L.J. Communication on noisy channels: A coding theorem for computation. In Proceedings of the 33rd Annual Symposium on Foundations of Computer Science, Pittsburgh, PA, USA, 24–27 October 1992; IEEE: Piscataway, NJ, USA, 1992; pp. 724–733.
4. Ben-Yishai, A.; Kim, Y.-H.; Ordentlich, O.; Shayevitz, O. A Lower Bound on the Interactive Capacity of Binary Memoryless Symmetric Channels. *arXiv* **2019**, arXiv:1908.07367.
5. Haeupler, B.; Velingker, A. Bridging the capacity gap between interactive and one-way communication. In Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, Barcelona, Spain, 16–19 January 2017; pp. 2123–2142.
6. Ben-Yishai, A.; Shayevitz, O.; Kim, Y.-H. Shannon Capacity Is Achievable for a Large Class of Interactive Markovian Protocols. In Proceedings of the 2019 IEEE International Symposium on Information Theory (ISIT), Paris, France, 7–12 July 2019.
7. Schulman, L.J. Coding for interactive communication. *IEEE Trans. Inf. Theory* **1996**, *42*, 1745–1756. [CrossRef]
8. Yao, A.C.C. Some complexity questions related to distributive computing (preliminary report). In Proceedings of the eLeventh Annual ACM Symposium on Theory of Computing, Atlanta, GA, USA, 30 April–2 May 1979; ACM: New York City, NY, USA, 1979; pp. 209–213.
9. Cover, T.M.; Thomas, J. *Elements of Information Theory*, 2nd ed.; Wiley: New York, NY, USA, 2006.
10. Kol, G.; Raz, R. Interactive channel capacity. In Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing, Palo Alto, CA, USA, 1–4 June 2013; ACM: New York City, NY, USA, 2013; pp. 715–724.
11. Haeupler, B. Interactive channel capacity revisited. In Proceedings of the 2014 IEEE 55th Annual Symposium on Foundations of Computer Science, Philadelphia, PA, USA, 19–21 October 2014; pp. 226–235.
12. Ghaffari, M.; Haeupler, B.; Sudan, M. Optimal error rates for interactive coding I: Adaptivity and other settings. In Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing, New York, NY, USA, 31 May–3 June 2014; pp. 794–803.
13. Agrawal, S.; Gelles, R.; Sahai, A. Adaptive protocols for interactive communication. In Proceedings of the 2016 IEEE International Symposium on Information Theory (ISIT), Barcelona, Spain, 10–15 July 2016; IEEE: Piscataway, New Jersey, NJ, USA, 2016; pp. 595–599.
14. Polyanskiy, Y.; Poor, H.V.; Verdú, S. Channel coding rate in the finite blocklength regime. *IEEE Trans. Inf. Theory* **2010**, *56*, 2307. [CrossRef]
15. Kushlevitz, E.; Nisan, N. *Communication Complexity*; Cambridge University Press: Cambridge, UK, 2006.
16. Kalyanasundaram, B.; Schintger, G. The probabilistic communication complexity of set intersection. *SIAM J. Discret. Math.* **1992**, *5*, 545–557. [CrossRef]
17. Razborov, A.A. On the distributional complexity of disjointness. In *International Colloquium on Automata, Languages, and Programming*; Springer: Berlin, Germany, 1990; pp. 249–253.

18. Bar-Yossef, Z.; Jayram, T.S.; Kumar, R.; Sivakumar, D. An information statistics approach to data stream and communication complexity. *J. Comput. Syst. Sci.* **2004**, *68*, 702–732. [CrossRef]
19. Braverman, M.; Rao, A. Information equals amortized communication. *IEEE Trans. Inf. Theory* **2014**, *60*, 6058–6069. [CrossRef]
20. Gallager, R.G. *Information Theory and Reliable Communication*; John Wiley & Sons: New York, NY, USA, 1968.