Time-Splitting Multiple-Access*

Bixio Rimoldi†
Mobile Communications Lab
Swiss Federal Institute of Technology
CH-1015 Lausanne
SWITZERLAND
email: bixio.rimoldi@epfl.ch

January 15, 1999

Abstract

It is shown that the encoding/decoding problem for any asynchronous M-user memoryless multiple-access channel can be reduced to corresponding problems for at most $2M-1$ single-user memoryless channels. This is done via a method called time-splitting multiple-access which is closely related to a recently developed method called rate-splitting multiple access. It is also related to to multilevel coding. The practical interest for time-splitting multiple access is is that it reduces the seemingly hard task of finding good multiple-access codes and implementable decoders for such codes to the much better understood task of finding codes and decoders for single-user channels. As a by-product, some interesting properties of the capacity region of M-user asynchronous discrete memoryless channels are derived.

Key words: Multiple-Access, Successive Decoding, Asynchronous Capacity Region, Dominant Face.

*Partial results were presented in [1].

†Part of this work has been supported by US National Science Foundation grants NCR-9357689 and NCR-9304763.

I Introduction

This paper addresses the problem of approaching any rate in the capacity region of an asynchronous M-user multiple-access channel via at most $2M-1$ encoding/decoding operations for point-to-point channels. For notational simplicity only discrete-time memoryless multiple-access channels are considered but extension to arbitrary input/output alphabets is straightforward and channel memorylessness does not seems to be essential.

The proposed approach differs from standard approaches is that we consider multiple encoders per user (but at most $2M-1$ encoders totally). This twist eliminates the difficult task of finding joint codes and the even more difficult task of finding implementable joint decoders for such codes. Seen from a different point of view, the twist of using multiple codes per user allows one to brake the original multiple-access channel into essentially independent point-to-point channels that are used in parallel. The method applies to any rate tuple in the capacity region of the channel at hand.

The coding problem for point-to-point channel is better understood than that for multiple access channel. In fact, for the most important additive white Gaussian noise channel it is now possible to approach its channel capacity with implementable complexity by means of "turbo" codes [2]. Similar progress is being made in the area of binary input memoryless channels (see [3] and references therein) using low-density parity check codes [4]. On the other hand, to our knowledge the only multiple-access channel model with a nontrivial capacity region for which explicit codes have been constructed to achieve all rate points in the capacity region is the collision channel without feedback studied by Massey and Mathys [5]. For the M-user synchronous binary-input real-adder channel, a family of asymptotically optimal codes with two codewords per user has been constructed [6]. The channels in [5] and [6] are noiseless. For a survey on codes for specific multiple-access channels see the introduction in [7].

The receiver is based on *successive decoding*. The key idea behind successive decoding is to decode one user at a time, using already produced codeword estimates as side information to decode remaining users. This technique is also known as *onion peeling*, *stripping*, and (not always appropriately) *successive cancellation*. It should be pointed out that successive decoding is mostly known for additive channels where the channel impulse response is known, like for the Gaussian multiple-access channel and its generalizations to additive channels with non-trivial impulse response and Gaussian noise. For such channels the name "successive cancellation" appropriately describes successive decoding. Indeed, for this case, successive deciding consists in decoding a user, estimating this user's contribution to the received signal, removing (canceling) its contribution from the received (sum) signal, and repeating the operation with other users. What is not so well known is that successive decoding applies in great generality. In particular, contrary to a widespread belief, for fading channels one does *not* need to know the channel impulse response to do successive

decoding. But of course, if the channel impulse response is known then the channel has a larger capacity region.

For a summary of multiple-access methods that allow one to approach any point in the capacity region see e.g. the introduction in [8].

The approach used in this paper is related to rate-splitting multiple-access (see [10] for Gaussian channels and [8] for general discrete memoryless channels). Yeh and Gallager [11] have developed a related approach which requires up to $\frac{1}{2}M \log_2 M + M$ point-to-point codes. Our apporach was motivated by [9].

The paper is organized as follows. Section II discusses time-splitting multiple-access for single-user and for two-user channels. This two cases are treated separately since they require little notation and are a good way to get started. They are also convenient to compare the two versions of time-splitting multiple-access considered in this paper, namely splitting by switching and generalized time-sharing. The single-user case is also interesting to point out the relationship of time-splitting multiple access to multilevel coding. The next two sections focus on splitting by switching. In Section III we show that one can approach any rate tuple in the capacity region of any discrete memoryless multiple-access channel by mans of at most $2M - 1$ point-to-point encoder/decoder. On the way we learn an interesting fact about the capacity region, namely that faces of the dominant face are those points for which the decoding process can be decomposed into parts. In Section IV we derive a closed-form expression to describe the achievable rates as a function of system parameters. In Section V we show that the results of the previous two sections apply also to generalized time-sharing.

II One and Two User Case

In this section we expose the main ideas by considering a general point-to-point channel and a general 2-inputs multiple-access channel.

II-A Point-To-Point Channels

Consider a single use channel $W : \mathcal{X} \to \mathcal{Y}$ and let P_X be an arbitrary but fixed input distribution which may or may not maximize $I(X;Y)$. We split X by means of a *switch* as shown in Fig. 1(a). The box in Fig. 1(a) representing the original DMC indicates the input distribution P_X, the resulting mutual information $I = I(X;Y)$ in bits/use, as well as the symbol-rate D in symbols/s. The switch has two inputs U and V, an output X,

and is controlled by a binary random variable $S \in \{1, 2\}$. If $S = 1$ the switch output is V, otherwise it is U. Notice that the input rate of the pair (U, V) (in symbols per second) is the same as the output rate of X. In other words, the switch *does not* serialize the two input streams (we will consider a serializer later). Instead, when U is passed to the output, V is dropped and vice-versa. Unless otherwise specified we will always assume that the *scheduling sequence* $\{S_n\}_{n=-\infty}^{\infty}$ is i.i.d., in which case the new channel with input (U, V) and output (Y, S) is a *discrete memoryless channel*. We will consider U and V as independent inputs (i.e. inputs that are independently encoded). Accordingly, to the new channel we assign the product input distribution $P_{U,V}(u, v) = P_X(u)P_X(v)$.

Since the switch drops half of the input symbols, it is somewhat surprising at first that the mutual information[1] $I(UV; YS)$ between inputs and outputs of the new channel equals $I(X; Y)$. Indeed, letting $\lambda = Pr\{S = 1\}$

$$\begin{aligned} I(UV; YS) &= I(UV; Y|S) \\ &= \lambda I(UV; Y|S = 1) + (1 - \lambda) I(UV; Y|S = 2) \\ &= \lambda I(X; Y) + (1 - \lambda) I(X; Y) \\ &= I(X; Y), \end{aligned}$$

where the first equality holds since UV is statistically independent of S. Hence,

$$\begin{aligned} I(X; Y) &= I(UV; YS) \\ &= I(U; YS) + I(V; YSU) \end{aligned}$$

where in the second line we used the chain rule of mutual information and the fact that U, V, and S are independent. The above relationship says that we can approach $I(X; Y)$ via stripping. Using the fact that $I(V; YSU) = I(V; YU|S) = \lambda I(V; YU|S = 1) = \lambda I(V; Y|S = 1) = I(V; YS)$, we further simplify to obtain

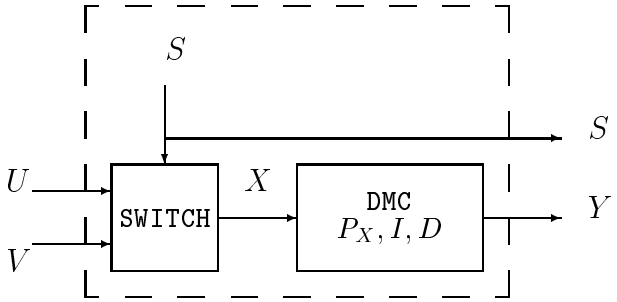
$$I(X; Y) = I(U; YS) + I(V; YS),$$

which says that one can decode inputs U and V independently (without stripping)[2]. This and the fact that $I(U; YS) = (1 - \lambda) I(X; Y)$ and $I(V; YS) = \lambda I(X; Y)$ imply that the channel of Fig. 1(a) may be thought of as consisting of two independent discrete memoryless channels as in Fig. 1(b). From the outside we have split the original channel into two independent channels. The maximal rates R_U and R_V for the two channels are plotted as a function of λ in Fig. 1(c).

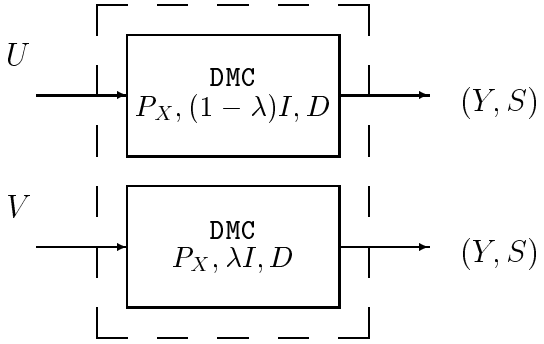
In particular, letting P_X be the capacity-achieving input distribution, the above shows that by means of the switch we reduce a DMC of capacity C bits/use into two parallel DMCs of capacity $(1 - \lambda)C$ and λC, respectively. The original channel as well as the two

[1] For compactness it is often convenient to write YS instead of Y, S.

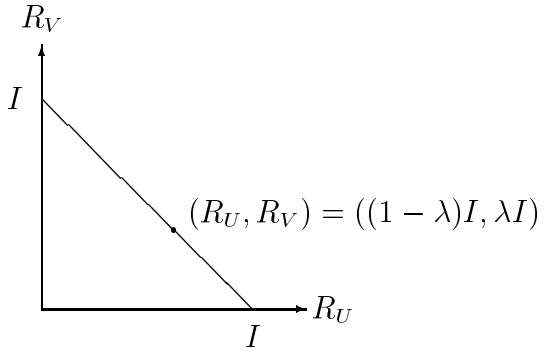
[2] The last step does not generalize to an M-input channel, i.e., stripping is needed in general.



(a) Original channel with split input.



(b) Equivalent system of two parallel channels.



(c) R_V vs R_U, parameterized by λ.

Figure 1: Single user case

resulting subchannels are used at the same symbol rate of D symbols/s. Conceptually, the switch is useful to fully utilize the capacity of a DMC by means of lower rate codes.

The above construction should not be confused with that of taking two parallel symbol streams and serializing them to form the input of a DMC. This seems to be a better idea at first since no symbol gets lost. To emphasize the parallelism with the random switch we could envision a random serializer which is also controlled by a random i.i.d. sequence $\{S_n\}$. (a buffer on each input is required to temporally store incoming symbols until they are passed to the output). The result is equivalent to having two parallel DMC of identical capacity as the original DMC and with symbol rates of λD and $(1-\lambda)D$ symbols, respectively. Hence the serializer is helpful if we do have a code of the desired rate of, say, C bits/use but its implementation is too slow to produce symbols at the desired rate of D symbols/s.

The parallelism between using a switch and using a serializer will be carried out throughout the paper and the two techniques will be referred to as *splitting by switching* and *generalized time-sharing*, respectively. The latter *is* a generalized form of time-sharing. In fact, the usual form of time-sharing of codeword is subsumed by controlling the serializer with a deterministic sequence $\{s_n\}$ which changes the switch position at the end of each codeword.

There is another way to describe what is going on in this point-to-point example with the input switch. Namely, we can think of the switch as a device that performs the operation of puncturing the two codewords entering the switch and interleaving (serializing) the resulting symbol sequences. By puncturing a codeword of rate R_U bits/symbol, one produces a now code of rate $R_U/(1-\lambda)$, where $(1-\lambda)$ is the fraction of retained codeword symbols. At the same time, the symbol rate reduces from D to $D(1-\lambda)$ symbols/s. Similarly, by puncturing the codeword entering the other input we go from dimensionless rate R_V to R_V/λ and from symbol rate D to $D\lambda$. Serializing the two resulting codeword sequences produces a symbol rate of D symbols/s and a dimensionless rate of $\frac{R_U}{(1-\lambda)}(1-\lambda) + \frac{R_V}{\lambda}\lambda = R_U + R_V$. While more elementary (since it does not require the notion of mutual information), the point of view taken in this paragraph does not seem to extend to the general case of M users. Moreover, to determine the maximal value for R_U and R_V one has to resort to the mutual information as done in previous paragraphs.

It is of course straightforward to generalize the above to an L-way split where L is an arbitrary positive integer. Specifically, consider a channel of capacity C. For any set of coefficients $\lambda_1, \lambda_2, \ldots, \lambda_L$ such that $\sum \lambda_i = 1$, one can create parallel channels of capacity $\lambda_1 C, \lambda_2 C, \ldots, \lambda_L C$ and use the ith channel independently by means of a code of rate arbitrarily closed to (but strictly less than) λC.

This idea of creating virtual inputs is related to multilevel coding [12, 13, 14, 15]

where one splits the channel input alphabet by means of some surjective function of many variables. In this paper we "split" the time axis instead of the input alphabet. In all cases, the motivation is to use various codes (generally of lower rate) as building blocks and to decode sequentially thereby reducing decoding complexity.

II-B 2-Input Multiple-Access Channels

Consider a 2-input multiple-access channel $W : \mathcal{X}_1 \times \mathcal{X}_2 \to \mathcal{Y}$. Any approachable rate tuple $R = (R_1, R_2) \in \mathsf{R}^2$ for which no component may be increased keeping fixed the other components fulfills

$$R_1 \leq I(X_1; YX_2), \tag{1}$$
$$R_2 \leq I(X_2; YX_1), \tag{1b}$$
$$R_1 + R_2 = I(X_1X_2; Y), \tag{1c}$$

for some product input distribution $P_{X_1}P_{X_2}$. From (1b) and (1c) we obtain

$$R_1 = I(X_1X_2; Y) - R_2 \geq I(X_1; Y) + I(X_2; Y|X_1) - I(X_2; Y|X_1) = I(X_1; Y),$$

where we used the chain rule of mutual information and the independence of X_1 and X_2. Hence (1) is equivalent to the following conditions

$$I(X_1; Y) \leq R_1 \leq I(X_1; YX_2), \tag{2}$$
$$R_1 + R_2 = I(X_1X_2; Y). \tag{2b}$$

We arbitrarily choose to split input X_2 by means of switch as shown in Fig. 2 and decode in the order U, X_1, V. By this we mean that while decoding U, X_1, V we may assume that the corresponding constituent decoder observes SY, SYU, and $SYUX_1$, respectively.[3] By the coding theorem for (single-user) discrete memoryless channels the following rates are approachable:

$$R_U = I(U; YS) = (1 - \lambda)I(X_2; Y) \tag{3}$$
$$R_{X_1} = I(X_1; YSU) = (1 - \lambda)I(X_1; YX_2) + \lambda I(X_1; Y) \tag{3b}$$
$$R_V = I(V; YSUX_1) = \lambda I(X_2; YX_1). \tag{3c}$$

By varying λ we can match R_{X_1} to any desired R_1 that fulfills (2). Then $R_U + R_V$ must match R_2 since

$$\begin{aligned} R_U + R_V + R_{X_1} &= (1 - \lambda)[I(X_2, Y) + I(X_1; YX_2)] + \lambda[I(X_1, Y) + I(X_2, YX_1)] \\ &= (1 - \lambda)I(X_1X_2, Y) + \lambda I(X_1X_2; Y) = I(X_1X_2; Y) \\ &= R_1 + R_2. \end{aligned}$$

[3] In the next section we will consider the general case and discuss how to rigorously account for the fact that a constituent decoder passes to the next decoder the estimates of (rather than the correct) channel input symbols.

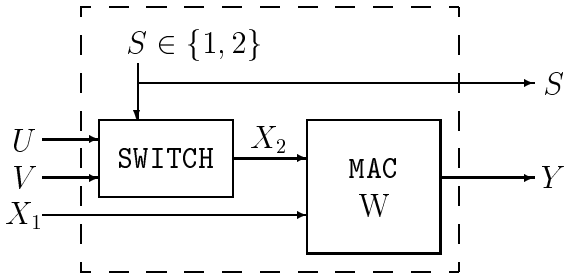


Figure 2: MAC with split input.

Hence any rate pair (R_1, R_2) that fulfills (2) is approachable using a switch and single-user coding/decoding techniques.

Now consider a serializer instead of a switch. All U inputs are send through the channel and the channel that they "see" is a discrete memoryless channel of mutual information $I(X_2;Y)$. Since they get to use the channel only a fraction $\lambda - 1)$ of time, their contribution to the rate of user 2 is at most $R_U = (1-\lambda)I(X_2;Y)$. Reasoning in this way for all of the virtual input we see that we obtain the same rates as in (3), showing that also generalized time sharing allows one to approach any rate pair (R_1, R_2) that fulfills (2) via single-user coding/decoding techniques.

In the next two sections we consider splitting by switching for the general M inputs case.

III Splitting by Switching: From Rate Vectors to System Parameters

In this section we show that, given an arbitrary rate tuple R in the capacity region of an M-user multiple-access channel, there is a way to split $M-1$ times by means of an input switch, to choose the corresponding λs, and to choose the decoding order, in such a way that the resulting rate tuple matches the desired rate R. The point is that no multiple-access coding/decoding is needed. Instead, one uses up to $2M-1$ point to point codes.

We review a few facts and introduce some notation. An M-user discrete memoryless multiple-access channel is defined in terms of M discrete[4] input-alphabets $\mathcal{X}_i$, $i \in$

[4]The discreteness of input/output alphabets is not essential. What we do in this paper works also for continuous alphabets.

$\{1, \cdots, M\}$, an output alphabet $\mathcal{Y}$, and a stochastic matrix $W : \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_M \to \mathcal{Y}$ with entries $W(y|x_1, x_2, \cdots, x_M)$ describing the probability that the channel output is y when the inputs are $x_1, x_2, \cdots, x_M$. For any product input distribution $P_{x_1}, \cdots, P_{x_M}$, define $\mathcal{R} = \mathcal{R}[W; P_{x_1}, \cdots, P_{x_M}]$ to be

$$\mathcal{R} = \{R \in \mathsf{R}_+^M : R(\mathcal{S}) \le I(X_{\mathcal{S}}; Y|X_{\mathcal{S}^c}), \quad \forall \mathcal{S} \subseteq [M]\}, \tag{4}$$

where we introduce the convenient notations

$$[i] \triangleq \{1, 2, \cdots, i\}, \tag{5}$$

$$R(\mathcal{S}) \triangleq \sum_{i \in \mathcal{S}} R_i, \tag{6}$$

$$X_{\mathcal{S}} \triangleq (X_i)_{i \in \mathcal{S}}, \tag{7}$$

$$\mathcal{S}^c \triangleq [M] \setminus \mathcal{S}, \tag{8}$$

and R_+ means the nonnegative reals. The capacity region depends on whether or not there is synchronism. A discrete-time channel (the only type of channel considered in this paper) is *synchronous* if transmitters are able to index channel input sequences in such a way that all inputs with time index n enter the channel at the same time. If this is not the case, meaning that there is an unknown shift between time indices, then the channel is said to be *asynchronous.*

The capacity region for either the synchronous or asynchronous channel may be described in terms of the region

$$\mathcal{C} = \bigcup_{P_{X_1} P_{X_2} \cdots P_{X_M}} \mathcal{R}[W; P_{X_1} P_{X_2} \cdots P_{X_M}], \tag{9}$$

where the union is over all product input distributions. The capacity region of the asynchronous multiple-access channel with arbitrarily large shifts between time indices is the closure of $\mathcal{C}$ [16, 17], whereas if shifts are bounded or the multiple-access channel is synchronous then its capacity is the closure of the convex hull of $\mathcal{C}$ [18, 19].

In this paper we consider only asynchronous channels. It follows that any point in the interior of the capacity region must be in $\mathcal{R}[W; P_{X_1} P_{X_2} \cdots P_{X_M}]$ for some $P_{X_1} P_{X_2} \cdots P_{X_M}$.[5] Hence we may (and will) assume that $P_{X_1} P_{X_2} \cdots P_{X_M}$ is arbitrary but fixed and focus on approaching any point in $\mathcal{R}[W; P_{X_1} P_{X_2} \cdots P_{X_M}]$.

The *dominant face* $\mathcal{D}$ is the set of those rates in $\mathcal{R}$ which have maximum sum-rate, i.e., $R([M]) = I(X_{[M]}; Y)$. Every point in $\mathcal{R}$ is dominated (componentwise) by a point in $\mathcal{D}$. Hence we may restrict our attention to points in $\mathcal{D}$.

[5] If the point of interest is on the *boundary* of the capacity region then for any $\epsilon > 0$ we can find a point in some $\mathcal{R}[W; P_{X_1} P_{X_2} \cdots P_{X_M}]$ which is within ϵ of the point of interest.

A key tool in this paper is the chain rule for mutual information,

$$I(X_{\mathcal{L}};Y) = I(X_{\mathcal{M}};Y) + I(X_{\mathcal{L}\setminus\mathcal{M}};Y|X_{\mathcal{M}}), \qquad \textit{(Chain Rule)}$$

which holds for any $\mathcal{M} \subseteq \mathcal{L} \subseteq [M]$. If $X_1, \cdots, X_M$ are independent random variables (which is always the case in this paper) then we obtain the alternative version

$$I(X_{\mathcal{L}};Y) = I(X_{\mathcal{M}};Y) + I(X_{\mathcal{L}\setminus\mathcal{M}};Y,X_{\mathcal{M}}). \qquad \textit{(Chain Rule for Independent Inputs)}$$

Both versions of the chain rule still hold if we condition each mutual information by $X_{\mathcal{N}}$ where $\mathcal{N} \cap \mathcal{L} = \emptyset$.

As a first application of the chain rule we verify that the following three descriptions for the dominant face are indeed equivalent.

$$\begin{aligned}
\mathcal{D} &= \{R \in \mathsf{R}_+^M : R(\mathcal{S}) \le I(X_{\mathcal{S}};Y|X_{\mathcal{S}^c}), \quad \forall \mathcal{S} \subseteq [M] \text{ with equality for } \mathcal{S} = [M]\}, \\
\mathcal{D} &= \{R \in \mathsf{R}^M : I(X_{\mathcal{S}};Y) \le R(\mathcal{S}), \quad \forall \mathcal{S} \subseteq [M] \text{ with equality for } \mathcal{S} = [M]\}, \\
\mathcal{D} &= \{R \in \mathsf{R}^M : I(X_{\mathcal{S}};Y) \le R(\mathcal{S}) \le I(X_{\mathcal{S}};Y|X_{\mathcal{S}^c}) \quad \forall \mathcal{S} \subseteq [M]\}.
\end{aligned} \tag{10}$$

The first one is the definition of $\mathcal{R}$ with the extra sum-rate constraint. The second is obtained from the first as follow.

$$\begin{aligned}
R(\mathcal{S}) &= R([M]) - R(\mathcal{S}^c) \\
&= I(X_{[M]};Y) - R(\mathcal{S}^c) \ge I(X_{[M]};Y) - I(X_{\mathcal{S}^c};Y|X_{\mathcal{S}}) = I(X_{\mathcal{S}};Y).
\end{aligned}$$

The third is obtained by combining the first two.

In this paper we find it useful to use (10) as our working definition for $\mathcal{D}$.

The "Lucky" Case: It is both convenient and instructive to consider first rates R on the boundary of $\mathcal{D}$.

Definition 1 A rate tuple $R \in \mathcal{D}$ lies in the *boundary of* $\mathcal{D}$ if there exists a proper subset $\mathcal{A} \subset [M]$ such that

$$R(\mathcal{A}) = I(X_{\mathcal{A}};Y). \tag{11}$$

For reasons that will become clear, a set $\mathcal{A}$ that satisfies (11) will be referred to as a *splitting set*.

From our working definition of $\mathcal{D}$ one immediately sees that our the definition of boundary does correspond to the geometrical interpretation of a boundary. Fig. 3 shows an example of $\mathcal{R}$ and $\mathcal{D}$ for $M = 3$ inputs. The edges of $\mathcal{D}$ are labeled with the corresponding

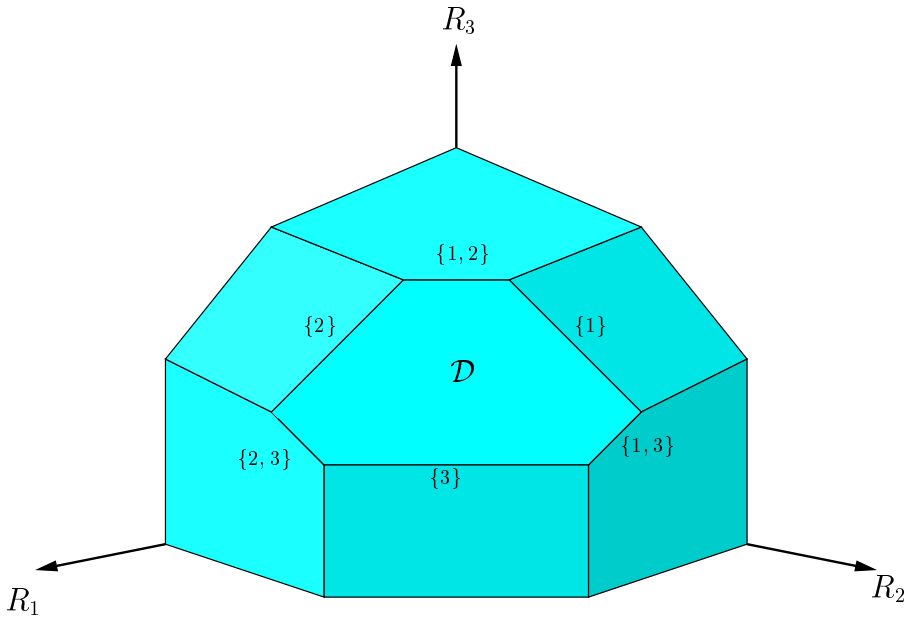


Figure 3: Example of $\mathcal{R}$, $\mathcal{D}$, and splitting sets $\mathcal{A}$ for $M = 3$ inputs.

splitting set $\mathcal{A}$. To verify that edges are labeled correctly, it is convenient to remember that $\mathcal{A}$ labels the edge of $\mathcal{D}$ for which the rate $R(\mathcal{A})$ is smallest. This is best seen from our working definition of $\mathcal{D}$. For instance, $\mathcal{A} = \{3\}$ labels the edge of $\mathcal{D}$ for which R_3 is smallest whereas $\mathcal{A} = \{1, 2\}$ labels the edge of $\mathcal{D}$ for which $R_1 + R_2$ is smallest.

Rates that are in the boundary of $\mathcal{D}$ are "lucky" since they are both "rare" (the boundary has volume zero) and, as we now see, they have the desirable property that the decoder can drastically cut decoding complexity. More specifically, if R is in the boundary of $\mathcal{D}$ and $\mathcal{A}$ is a corresponding splitting set, then we can decode (jointly) the subset of inputs with index in $\mathcal{A}$ and subsequently decode (jointly) the subset of inputs with index in the complement set $\mathcal{A}^c = [M] \setminus \mathcal{A}$. This cuts decoding complexity which grows exponentially in the number of jointly decoded inputs.

By definition, for a point in the boundary there exists at least a splitting set $\mathcal{A} \subset [M]$ such that

$$R(\mathcal{A}) = I(X_{\mathcal{A}}; Y). \tag{12}$$

From this and

$$R(\mathcal{L}) \geq I(X_{\mathcal{L}}; Y) \quad \forall \mathcal{L} \subset \mathcal{A}$$

(which indeed holds for all $\mathcal{L} \subseteq [M]$ when R is in $\mathcal{D}$), the definition of $\mathcal{D}$ (cf. (10)) implies

$$R_{\mathcal{A}} \in \mathcal{D}[W_{Y|X_{\mathcal{A}}}; P_{X_{\mathcal{A}}}], \tag{13}$$

where $\mathcal{D}[W_{Y|X_{\mathcal{A}}}; P_{X_{\mathcal{A}}}]$ is the dominant face of the channel

$$W_{Y|X_{\mathcal{A}}} = \sum_{X_{\mathcal{A}^c}} W_{Y|X_{[M]}} P_{X_{\mathcal{A}^c}}.$$

The channel $W_{Y|X_{\mathcal{A}}}$ is the one seen by inputs $X_{\mathcal{A}}$ when the other inputs are considered as "noise" with distribution $P_{X_{\mathcal{A}^c}}$.

The multiple-access coding theorem says that $X_{\mathcal{A}}$ can be reconstructed by a joint decoder that knows the codebook of users in $\mathcal{A}$ and is totally unaware of the structure of the codebooks of users in $\mathcal{A}^c$, provided that $X_{\mathcal{A}^c}$ looks like an i.i.d. random vector with distribution $P_{X_{\mathcal{A}^c}}$. The main idea is depicted in Fig. 4.

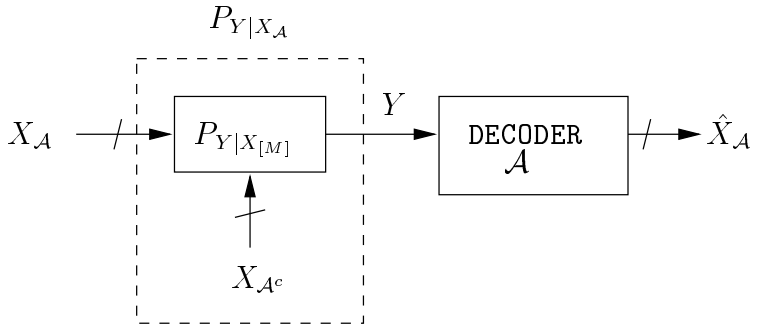


Figure 4: Constituent decoder for a "lucky rate" with splitting set $\mathcal{A}$: Part that decodes $X_{\mathcal{A}}$.

Condition (12), the fact that $R \in \mathcal{D}$, and the chain rule imply

$$R(\mathcal{A}^c) = R([M]) - R(\mathcal{A}) = I(X_{[M]}; Y) - I(X_{\mathcal{A}}) = I(X_{\mathcal{A}^c}; Y X_{\mathcal{A}}). \tag{14}$$

Moreover,

$$R(\mathcal{L}) \geq I(X_{\mathcal{L}}; Y X_{\mathcal{A}}) \quad \forall \mathcal{L} \subset \mathcal{A}^c, \tag{15}$$

which follows straightforwardly from the chain rule together with

$$\begin{aligned} R(\mathcal{L}) &= R(\mathcal{A}^c) - R(\mathcal{A}^c - \mathcal{L}), \\ R(\mathcal{A}^c - \mathcal{L}) &\leq I(X_{\mathcal{A}^c - \mathcal{L}}; Y X_{\mathcal{A}} X_{\mathcal{L}}), \end{aligned}$$

and (14). From (14) and (15),

$$R_{\mathcal{A}^c} \in \mathcal{D}[W_{Y X_{\mathcal{A}}|X_{\mathcal{A}^c}}; P_{X_{\mathcal{A}^c}}], \tag{16}$$

where $\mathcal{D}[W_{Y X_{\mathcal{A}}|X_{\mathcal{A}^c}}; P_{X_{\mathcal{A}^c}}]$ is the dominant face of the channel

$$W_{Y X_{\mathcal{A}}|X_{\mathcal{A}^c}} = P_{X_{\mathcal{A}}} W_{Y|X_{\mathcal{A}^c}} = P_{X_{\mathcal{A}}} \sum_{X_{\mathcal{A}}} W_{Y|X_{[M]}} P_{X_{\mathcal{A}}}$$

when the input distribution is $P_{X_{\mathcal{A}^c}}$.

The multiuser coding theorem says that $X_{\mathcal{A}^c}$ is decodable reliably by an hypothetical decoder that observes Y and $X_{\mathcal{A}}$. The receiver is hypothetical since it requires a "genie" that tells constituent decoder $\mathcal{A}^c$ the exact value of $X_{\mathcal{A}}$. The idea is depicted in Fig. 5.

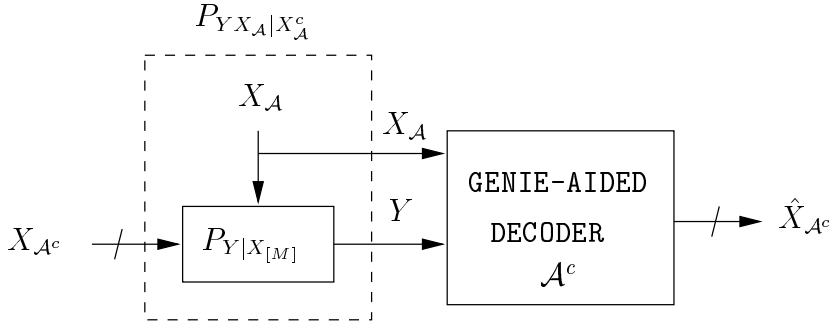


Figure 5: Constituent decoder for a "lucky rate" with splitting set $\mathcal{A}$: Part that decodes $X_{\mathcal{A}^c}$.

Putting the above two decoders together, we can decode users in $\mathcal{A}$ and users in $\mathcal{A}^c$ independently by means of the hypothetical receiver shown in Fig. 6(a).

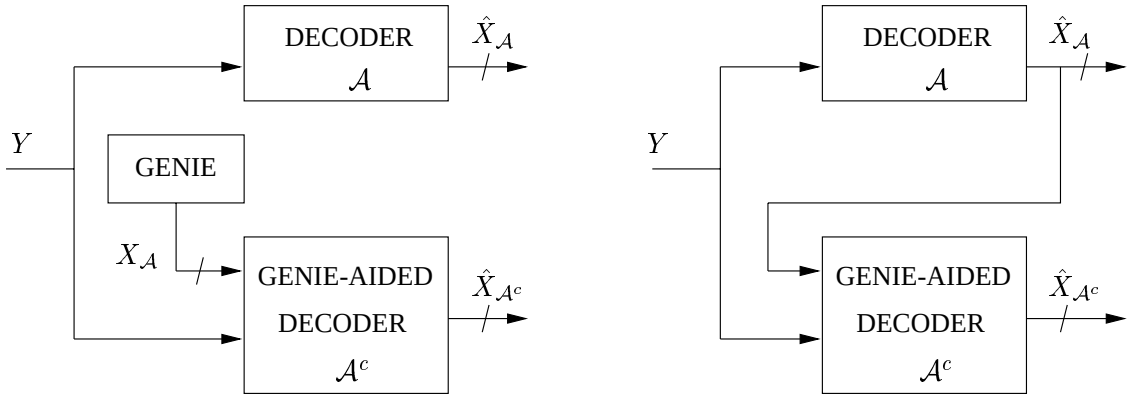


Figure 6: (a) Hypothetical genie-aided decoder; (b) corresponding successive decoder.

Fig. 6 (b) shows a corresponding successive decoder. Corresponding constituent decoders in Fig. 6(a) and 6(b) are identical. It is a surprising fact that the genie can be exorcized as in Fig. 6(b) without affecting the overall error probability or, equivalently, the probability of deciding correctly defined as

$$P_c := Pr\{X_{[M]} = \hat{X}_{[M]}\}. \tag{17}$$

This follows from the following simple argument first made in [10]. Since the top decoders receive the same input, they make the same decision. In particular, if one is correct the other is also correct. Conditioned on the top decoder being correct, also the bottom decoders of both figures observe the same input and thus make the same decision. From Bayes rule we see that the probability of being correct is the same in both cases.

The relationship between the genie-aided decoder of Fig.6(a) and the successive decoder of Fig.6(b) may be summarized as follows. The genie-aided decoder decodes users in $\mathcal{A}$ and users in $\mathcal{A}^c$ in parallel and completely independently. The probability of error of each constituent decoder can be computed using usual techniques. In particular, one can use random coding techniques to bound their error probability. Unfortunately, the genie-aided decoder can't be implemented. What we can implement is the successive decoder. Here constituent decoders work in series. The delay accumulates and the error probability of the bottom constituent decoder does depend on that of the top constituent decoder. Moreover, even conditioning on the top decoder being correct, it would be hard to determine the error probability of the bottom decoder. This is so since the fact that the top decoder is correct implies that Y is in a certain decoding region: this changes the originally known statistics of Y to some statistics that we don't know how to calculate. Fortunately, we can analyze the genie-aided decoder and implement the successive decoder knowing that their overall probability of error is identical. It remains to be argued that the overall joint error probability (17) is a meaningful performance criterion. The reason for splitting the decision about $X_{[M]}$ in two (or more) steps is that what we would like to implement, namely a maximum likelihood (joint) decoder, is too complex. For a joint decoder it is natural to use (17) as performance criteria. The successive decoder of Fig. 6(a) is also a joint decoder but of reduced complexity. Since it is a joint decoder and since we are interested in comparing its performance to that of a maximum-likelihood decoder, it makes sense to evaluate the successive decoder according to (17).

The Regular Case: This is the case when there is no (splitting) set $\mathcal{A}$ fulfilling (11). We will now show how to create the previous situation (twice) by splitting an input using the switch. We arbitrarily choose to split input M. The situation is that of Fig. 7. The new inputs will be denoted by Z_{M+1} and Z_M, respectively, and we assign to both of them the the probability distribution P_{X_M} of the split input. For notational convenience we also define $Z_{[M-1]} = X_{[M-1]}$.

Let r_{M+1} and r_M be the rates of Z_{M+1} and Z_M, respectively. To be fair (to the user of the split input), we require that

$$r_{M+1} + r_M = R_M. \tag{18}$$

For notational convenience we also define $r_{[M-1]} = R_{[M-1]}$.

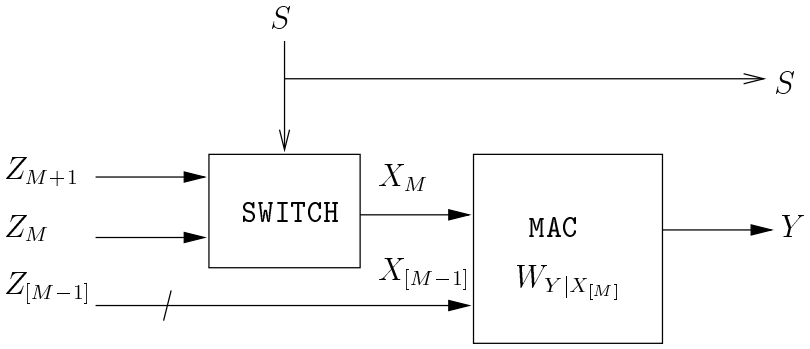


Figure 7: M-ary MAC with split input.

Next we assign

$$r_{M+1} \stackrel{!}{=} I(Z_{M+1}; YS) = (1-\lambda) I(X_M; Y), \tag{19}$$

where λ is the probability that the switch connects Z_M to X_M. We will determine the actual value of λ later. Regardless of the value for λ, this choice for r_{M+1} allows virtual input $M+1$ to be decoded on its own, independently of other users. This follows from the channel coding theorem for (point-to-point) discrete memoryless channels and the first equality in (19).

The reader may wonder if the remaining inputs $Z_{[M]}$ whose assigned rate is $r_{[M]}$ (and depends on λ through the Mth component) is decodable regardless of λ. The answer is no but in the interest of continuity we will come back to this later. We will choose λ carefully to ensure that (i) the remaining inputs are decodable and (ii) that $r_{[M]}$ be decodable in two parts as in the "lucky" case. Without (ii) we would be back to the original situation of having to decode M inputs jointly. The next Lemma guarantees the existence of a λ that fulfills (i) and (ii).

To decode $Z_{[M]}$ we temporarily use an hypothetical genie who knows Z_{M+1}. As done previously we can then remove the genie by decoding Z_{M+1} first and using the (possibly wrong) estimate of Z_{M+1} without affecting the overall error probability. With the genie, the channel seen by the decoder of $Z_{[M]}$ is the discrete memoryless multiple access channel $W_{YSZ_{M+1}|Z_{[M]}}$ defined by

$$W_{YSZ_{M+1}|Z_{[M]}} = P_{Z_{M+1}} W_{YS|Z_{[M]}}.$$

The existence of a λ that fulfills (i) and (ii) above is guaranteed by the following Lemma.

Lemma 1 Let $Z_{[M]}$, $W_{YSZ_{M+1}|Z_{[M]}}$, and $r_{[M]}$ be defined as above. There exists a $\lambda \in (0,1)$ such that $r_{[M]}$ is in the boundary of $\mathcal{D}[W_{YSZ_{M+1}|Z_{[M]}}, P_{X_1}, P_{X_2}, \ldots, P_{X_M}]$.

Proof It is sufficient to show that

$$r(\mathcal{L}) \geq I(Z_{\mathcal{L}}; YSZ_{M+1}) \tag{20}$$

holds for all $\mathcal{L} \subseteq [M]$ and that equality holds for $\mathcal{L} = [M]$ and for some $\mathcal{A} \subset [M]$. First assume that $M \in \mathcal{L}$. Then, regardless of λ,

$$\begin{aligned} r(\mathcal{L}) &= R(\mathcal{L}) - r_{M+1} \\ &\geq I(X_{\mathcal{L}}; Y) - I(Z_{M+1}; YS) \\ &= I(Z_{\mathcal{L}\cup\{M+1\}}; YS) - I(Z_{M+1}; YS) \\ &= I(Z_{\mathcal{L}}; YSZ_{M+1}) \end{aligned}$$

with equality if $\mathcal{L} = [M]$. Now we consider the case $M \notin \mathcal{L}$, i.e., when $\mathcal{L} \subseteq [M-1]$. For this case $r(\mathcal{L}) = R(\mathcal{L})$ so that

$$r(\mathcal{A}) > I(X_{\mathcal{A}} Y) \quad \forall \mathcal{A} \subseteq [M-1], \tag{21}$$

where equality is not possible since by assumption R is not a lucky rate of the original channel. Moreover,

$$r(\mathcal{A}) < I(X_{\mathcal{A}}; YX_M) \tag{22}$$

for some $\mathcal{A} \subseteq [M-1]$. Indeed for $\mathcal{A} = [M-1]$ we obtain

$$r(\mathcal{A}) = R(\mathcal{A}) = R([M-1]) < I(X_{[M-1]}; YX_M) = I(X_{\mathcal{A}}; YX_M)$$

where the inequality has to hold since $R \in \mathcal{R}$ and it has to be strict since equality would again imply that R is a lucky rate of the original channel with splitting set $[M-1]$.

Let $\mathcal{F} \subseteq 2^{[M-1]}$ be the set of subsets $\mathcal{A}$ for which (22) holds. For each $\mathcal{A} \in \mathcal{F}$, combining (21) and (22) we obtain

$$I(X_{\mathcal{A}}; Y) < r(\mathcal{A}) < I(X_{\mathcal{A}}; YX_M).$$

Hence there exists a $\lambda \in (0,1)$, denoted $\lambda_{\mathcal{A}}$ to be explicit about its association to $\mathcal{A}$, such that

$$r(\mathcal{A}) \stackrel{!}{=} (1-\lambda_{\mathcal{A}}) I(X_{\mathcal{A}}; Y) + \lambda_{\mathcal{A}} I(X_{\mathcal{A}}; YX_M) = I(Z_{\mathcal{A}}; YSZ_{M+1}). \tag{23}$$

Among all $\mathcal{A} \in \mathcal{F}$ we pick an $\mathcal{A}$ for which $\lambda_{\mathcal{A}}$ is smallest. Since the right side of (23) is non-decreasing in λ, choosing the smallest λ ensures that (20) holds true for all $\mathcal{L} \in \mathcal{F}$, with equality for $\mathcal{L} = \mathcal{A}$. If $\mathcal{L} \notin \mathcal{F}$ then $r(\mathcal{L}) \geq I(X_{\mathcal{L}}; YX_M)$ by definition of $\mathcal{F}$. But $I(X_{\mathcal{L}}; YX_M) = I(Z_{\mathcal{L}}; YX_M) = I(Z_{\mathcal{L}}; YSZ_M Z_{M+1}) \geq I(Z_{\mathcal{L}}; YSZ_{M+1})$. □

Arguing as we did earlier, instead of decoding $Z_{\{M+1\}}$, $Z_{\mathcal{A}}$, and $Z_{\mathcal{A}^c}$ where $\mathcal{A}^c = [M] \backslash \mathcal{A}$ independently and in parallel with the decoder for $\mathcal{A}$ helped by a genie who knows Z_{M+1}

and the decoder of $\mathcal{A}^c$ helped by a genie who knows both Z_{M+1} and $Z_{\mathcal{A}}$, we can decode serially without genie and incur in the exact same probability of error.

The following is a "high level" view of what we have done so far in this section. If R is not in the boundary of $\mathcal{D}$ then by splitting X_M into Z_M and Z_{M+1} we create a degree of freedom in our ability to choose λ while letting $r_{M+1} = (1-\lambda)I(X_M;Y)$. By varying λ, $r_{[M+1]}$ moves on a straight line between $(R_{[M-1]}, R_M - I(X_M;Y), I(X_M;Y))$ and $(R_{[M-1]}, R_M, 0)$. Notice that when $\lambda = 0$ r_{M+1} is set to $I(X_M;Y)$ which is, in general, larger than R_M. If this is the case then r_M is negative which is sufficient to preclude $r_{[M]}$ from being in the $\mathcal{R}$ of the corresponding channel. But this is a technical detail which is avoided by letting λ vary in the interval $\Lambda = [\lambda_0, 1]$, where λ_0 is the λ for which $r_{M+1} = R_M$ (see also 19). With this restriction the trajectory of $r_{[M+1]}$ is always in $\mathcal{D}[W_{YS|Z_{[M+1]}}; P_{X_1}, P_{X_2}, \ldots, P_{X_M}, P_{X_M}]$. Intuitively, this is the case since $r_{[M+1]}$ corresponds to the original $R_{[M]}$ which is an achievable rate tuple. On the other hand, when $\lambda \in \Lambda$ the tuple $r_{[M]}$ moves on a straight line between $(R_{[M-1]}, 0)$ and $(R_{[M-1]}, R_M)$. While the starting point is in $\mathcal{D}(W_{YSZ_{M+1}|Z_{[M]}}; P_{X_1}, \ldots, P_{X_M})$, the ending point is outside in general (and on the boundary otherwise). Hence, there exists a $\lambda \in \Lambda$ for which $r_{[M]}$ is on the boundary of $\mathcal{D}(W_{YSZ_{M+1}|Z_{[M]}}; P_{X_1}, \ldots, P_{X_M})$. This makes $r_{[M]}$ a "lucky rate," decodable in two steps.

Now we reiterate the procedure. With the genie, the constituent decoders for Z_{M+1}, $Z_{\mathcal{A}}$, and $Z_{\mathcal{A}^c}$ see three *independent* multiple-access channels with 1, $|\mathcal{A}|$, and $|\mathcal{A}^c|$ inputs, respectively. The above procedure can be applied without change to any of these subchannels. [6] At the end the genies are removed invoking the usual argument.

A question of interest is: In the worst-case scenario, how many virtual inputs do we need to ensure that each of them is decodable independently? This question is answered by the Corollary following next Lemma.

Lemma 2 For a given channel with M inputs, it suffices to split $M-1$ times to guarantee that the resulting virtual inputs are decodable one by one.

Proof By induction on M. With $M = 1$ no split is needed and the claim is true. Assume that the claim is true with less than M inputs and consider a situation with M inputs. We split input M as done above and we create the situation depicted in Fig. 8. Let $m = |\mathcal{A}|$ and $M - m = |\mathcal{A}^c|$. Since both m and $M-m$ are less than M, we can apply induction to each of the three groups of inputs. The resulting number of splits is at most

$$1 + 0 + (m-1) + (M-m-1) = M-1,$$

[6] Even a virtual input that can be decoded on its own can further be split to create lower rate inputs as explained for point-to-point channels in Section II.

where the first term accounts for the split required to create the situation of Fig. 8. □

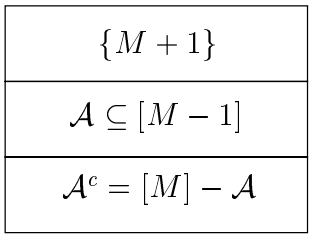


Figure 8: Mnemonic representing the situation after a split.

Since at each split we create a new input, splitting $M-1$ times creates $2M-1$ virtual inputs. This is the number of encoders and decoders needed to achieve any desired rate tuple in the asynchronous capacity region. We summarize:

Corollary 1 By splitting at most $M-1$ of the M inputs of a multiple-access channel it is possible to create a new multiple-access channel with at most $2M-1$ virtual input that can be encoded by means of a code for a point to point channel and can decoded one by one sequentially. □

By the time we are done with the construction explained in this section, possibly carried out to the point that each virtual user is decoded independently, original input i has been split into some number M_i of virtual inputs. Fig. 9(a) shows an example where input i is split 3 times, resulting in the equivalent 4-way split of Fig. 9(b). We can label virtual inputs with elements of the set $\{(i,j) : i \in [M], j \in [M_i]\}$. The final decoding order is described by a permutation K of this set. The fraction of time that virtual input (i,j) is connected to input i is described by the probability mass function $P_{S_i}(s)$ of the random variable S_i with support set $[M_i]$. Using vector notation we define $S = (S_1, \ldots, S_M)$ and $P_S = \prod P_{S_i}$. The resulting rate tuple r depends on K and P_S.

In this section we assumed that R was given and showed that it is possible to achieve it by means of at most $2M-1$ point-to-point encoder/decoder pairs. In a more pragmatic approach one may not start with a fixed rate tuple R in mind. Rather, a network provider may dictate the splits, the associated λs, and the decoding order, and one would like to know the resulting rate R as a function of K and P_S. In the next section we derive a closed form for the resulting rate as a function of K and P_S.

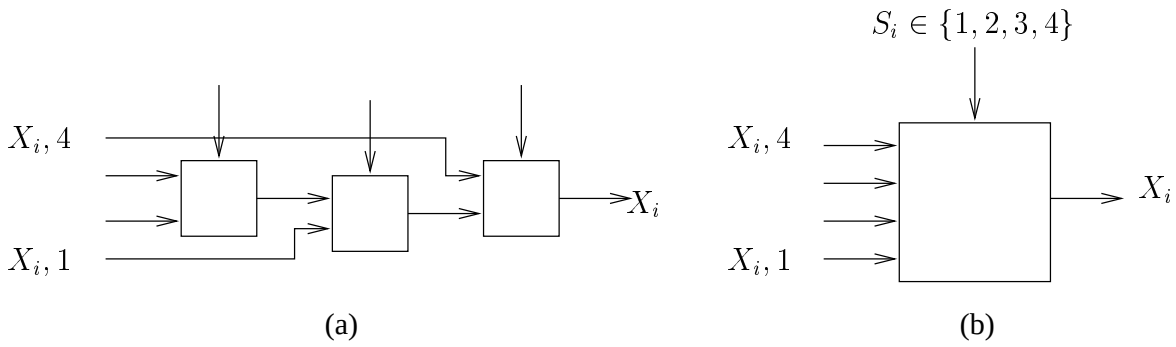


Figure 9: Example of three 2-way splits represented as a single 4-way split.

IV Splitting by Switching: From System Parameters to Rate Vectors

In this section, as in the previous one, the channel input distribution is an arbitrary but fixed product distribution. Channel input i (with assigned input distribution P_{X_i}) has been split M_i ways by means of a switch controlled by a sequence of i.i.d. copies of a random variable $S_i \in [M_i]$ with distribution P_{S_i}. When $S_i = j$ the switch connects virtual input $X_{(i,j)}$ to (original) input X_i. Each of the M_i virtual inputs corresponding to original input i are assigned the distribution P_{X_i}. The random M tuple $S = (S_1, \ldots, S_M)$ describing the position of each switch has distribution $P_S = \prod P_{S_i}$. The decoding order is described by K.

For each decoding order K and probability P_S, there is a rate tuple, denoted $\Psi = \Psi[W; P_{[M]}; K; P_S]$, which dominates all rate tuples consistent with the specified parameters. The primary goal of this section is to describe this rate tuple Ψ.

At any time, each original channel input has exactly one virtual input attached to it. We will call this the *active* virtual input. The decoding order K orders virtual inputs. However, *exactly one* virtual input per original input is active at any given time (and S specifies which). Hence, K and S also determine a decoding order on the *original* inputs. Let the M-tuple $v = v(K, S)$ be this decoding order. Let $\mathcal{L}(v, i)$ be the set of indices to to the left of i in v. The inputs $X_{\mathcal{L}(v,i)}$ are those decoded prior to X_i.

Example 1 If $K = (21, 31, 11, 32, 22)$ and $s = (1, 2, 1)$, then the active inputs are the bold entries in $K = (21, \mathbf{31}, \mathbf{11}, 32, \mathbf{22})$. Hence $v = (3, 1, 2)$, meaning that (original) input 3 is decoded first, input 1 second, and input 2 third. $\mathcal{L}(v, 1) = \{3\}, \mathcal{L}(v, 2) = \{3, 1\}$, and $\mathcal{L}(v, 3) = \emptyset$. Notice that $s \neq s'$ does not necessarily imply $v(K, s) \neq v(K, s')$. For instance if $K = (21, 31, 22, 32, 11)$ then both $s = (1, 1, 1)$ and $s = (1, 2, 2)$ lead to $v = (2, 3, 1)$.

Now we precoded to determine Ψ. We assume that each constituent decoder is helped by the corresponding genie since we know that we can remove the genie in the usual way without affecting the error probability.

Consider for a moment fixing $S = s$. Then, letting $v = v(K, s)$, the active constituent decoder of user i observes Y and $X_{\mathcal{L}(v,i)}$. Hence, the active encoder of user i communicates to the corresponding decoder through channel $W_{YX_{\mathcal{L}(v,i)}|X_i}$. The mutual information of this channel is $I(X_i; YX_{\mathcal{L}(v,i)})$. The mutual information of the channel seen by all other encoder/decoder pairs of user i is 0. Using the indicator function $1_{\mathcal{A}}(j)$

$$1_{\mathcal{A}}(j) = \begin{cases} 1 & j \in \mathcal{A} \\ 0 & \text{otherwise,} \end{cases}$$

the mutual information of the channel seen be encoder/decoder pair (i, j) is $I(X_i; YX_{\mathcal{L}(v,i)})1_{\{s_i\}}(j)$.

Now let $v = v(K, S)$ be a random decoding order, selected via the i.i.d. random variable S. This means that the encoder/decoder pair of user i sees a randomly selected channel. As described in Appendix A, the resulting channel has the average mutual information

$$\sum_s P_S(s) I(X_i; YX_{\mathcal{L}(v(K,s),i)}) 1_{\{s_i\}}(j).$$

Adding over all $j \in [M_i]$ we obtain

$$\Psi_i = \sum_s P_S(s) I(X_i; YX_{\mathcal{L}(v(K,s),i)}) \sum_{j \in [M_i]} 1_{\{s_i\}}(j) = \sum_s P_S(s) I(X_i; YX_{\mathcal{L}(v(K,s),i)}). \tag{24}$$

Passing to M-tuple notation, defining

$$\Psi = \begin{pmatrix} \Psi_1 \\ \Psi_2 \\ \vdots \\ \Psi_M \end{pmatrix} \qquad \text{and} \qquad I_v = \begin{pmatrix} I(X_1; YX_{\mathcal{L}(v,1)}) \\ I(X_2; YX_{\mathcal{L}(v,2)}) \\ \vdots \\ I(X_M; YX_{\mathcal{L}(v,M)}) \end{pmatrix},$$

we obtain

$$\Psi = \sum_s P_S(s) I_{v(K,s)}.$$

I_v is the vertex of $\mathcal{R}[W; \prod P_{X_i}]$ when the decoding order is v. Since $I_{v(K,s)}$ depends on K and s only through v, we can also write

$$\Psi = \sum_v P_V(v) I_v,$$

where we defined

$$P_V(v) = \sum_{s:v(K,s)=v} P_S(s).$$

We summarize:

Theorem 1 *Let $W : \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_M \to \mathcal{Y}$ be a discrete memoryless multiple-access channel equipped with: a product input distribution $\prod P_{X_i}$; an M_i way switch at input i, $i \in [M]$ controlled by a random variable $S_i \in [M_i]$ distributed according to P_{S_i}; a decoding order K. Then*

$$\Psi = \sum_s P_S(s) I_{v(K,s)} = \sum_v P_V(v) I_v. \tag{25}$$

□

The above theorem generalizes straightforwardly to memoryless channels with arbitrary alphabets.

We give two examples to gain some intuition on what happens as we vary P_S keeping the decoding order K fixed.

Example 2 *(Two Users)* Consider a two user channel and an arbitrary product distribution $P_{X_1} P_{X_2}$ on its inputs. We arbitrarily choose $K = (21, 11, 22)$ and define $P_{S_2}(1) = \lambda$. This choice implies the existence of two codebooks for user 2, used a fraction λ and $1 - \lambda$ of time, respectively, and one codebook for user 1. The relationship between s, v_s, and the resulting P_V are given in the following table:

s	$(1,1)$	$(1,2)$
v	$(2,1)$	$(1,2)$
$P_V(v)$	λ	$1-\lambda$

For this case the right side of (25) is

$$\Psi = \sum_v P_V(v) I_v = \lambda I_{(2,1)} + (1-\lambda) I_{(1,2)}$$

By varying $\lambda \in [0, 1]$ we sweep out the dominant face $\mathcal{D}$ of $\mathcal{R}$ shown in Fig. 10. ◇

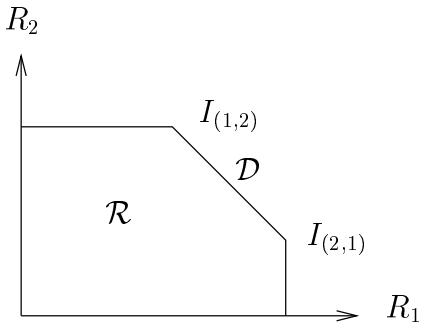


Figure 10: A two user example.

Example 3 (Three Users) Consider a three user channel and an arbitrary product distribution $P_{X_1}P_{X_2}P_{X_3}$ on its inputs. Except for special cases in which a user does not interfere with the other two (see e.g. [8]), the dominant face $\mathcal{D}$ of $\mathcal{R}[W; \prod P_{X_i}]$ is an hexagon embedded in R^3 as shown in Fig. 11(a) where vertices are labeled by the corresponding input decoding order v (see also Fig. 3). Hence $\mathcal{D}$ is a two dimensional convex polytope. To label vertices it is convenient to keep in mind that if the label starts with i, $i \in [M]$, then user i is the first decoded and therefore his rate is smaller than for any other vertex. If the label ends with i then the rate of user i is as large as possible. Now pick arbitrarily $K = (11, 21, 31, 22, 12)$, implying two codebooks per user for users 1 and 2, and a single codebook for user 3. With this choice and defining $P_{S_i}(1) = \lambda_i$, $i = 1, 2$ and $P_{S_3}(1) = 1$, the relationship between s, v, and P_V becomes:

s	$(1,1,1)$	$(2,1,1)$	$(1,2,1)$	$(2,2,1)$
v	$(1,2,3)$	$(2,3,1)$	$(1,3,2)$	$(3,2,1)$
$P_V(v_s)$	$\lambda_1\lambda_2$	$(1-\lambda_1)\lambda_2$	$\lambda_1(1-\lambda_2)$	$(1-\lambda_1)(1-\lambda_2)$

Using the above table the right side of (25) becomes

$$\begin{aligned}\sum_v P_V(v)I_v &= \lambda_1\lambda_2 I_{(1,2,3)} + \lambda_1(1-\lambda_2)I_{(1,3,2)} + (1-\lambda_1)\lambda_2 I_{(2,3,1)} + (1-\lambda_1)(1-\lambda_2)I_{(3,2,1)} \\ &= (1-\lambda_2)A + \lambda_2 B,\end{aligned}$$

where we defined

$$\begin{aligned}A &= (1-\lambda_1)I_{(3,2,1)} + \lambda_1 I_{(1,3,2)} \\ B &= (1-\lambda_1)I_{(2,3,1)} + \lambda_1 I_{(1,2,3)},\end{aligned}$$

By varying λ_1 and λ_2 in $[0, 1]$ we sweep out the shaded region in Fig. 11(b).

One can easily verify that with $K = (21, 11, 22, 31, 12)$ and $K = (11, 31, 21, 12, 22)$ we obtain the triangle above and below the shaded area in Fig. 11(b), respectively.

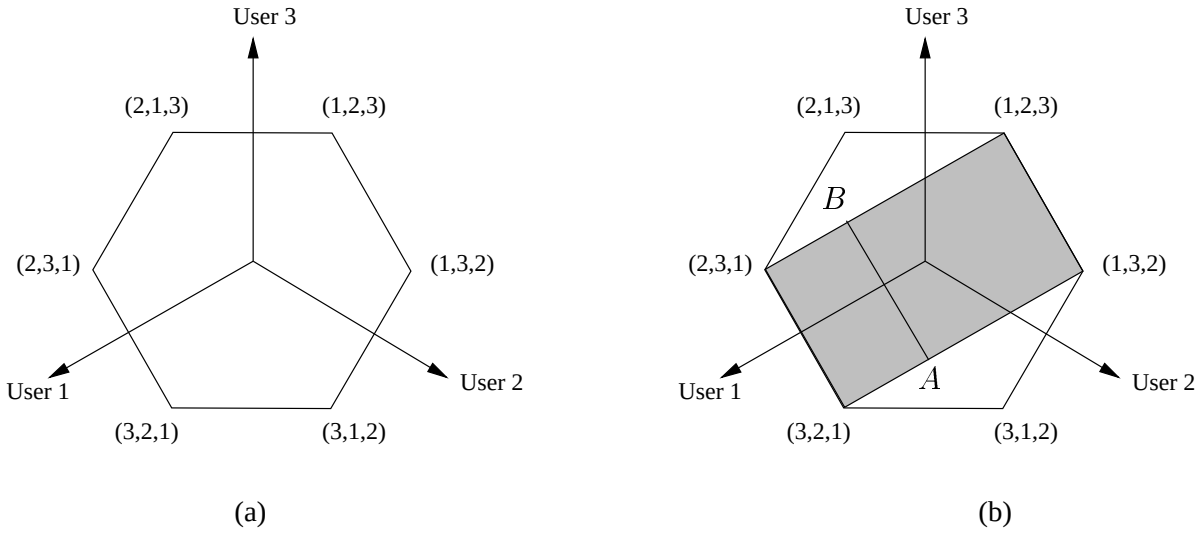


Figure 11: Dominant face.

More specifically, the following observations can easily be verified. Let $\Psi(K)$ be the achievable region corresponding to a given K, i.e., the image of Ψ as a function of λ_1 and λ_2 when K is fixed. The achievable region is bounded by the image of Ψ when one of the λs is held fixed to either 0 or 1. For instance the upper boundary of the shaded region in Fig. 11(b) is the image of $\Psi(K)$ as a function of λ_1 when $\lambda_2 = 1$. When $\lambda_2 = 1$, the entry 22 in K is irrelevant. Hence we can think of the boundary under consideration as of the image of $\Psi(K')$ where K' is K with entry 22 removed. Now let K'' be K' with the entry 22 inserted to the left. The image of $\Psi(K'')$ when $\lambda_2 = 1$ is also the image of $\Psi(K')$. Hence the image of $\Psi(K)$ and that of $\Psi(K'')$ share a common boundary (namely the image of $\Psi(K')$). This example shows how to change a K so that the new achievable region neighbors the old one. In principle, this tells us how to change K repeatedly so as to cover the entire dominant face. $\diamond$

V Generalized Time-Sharing

There is a tight connection between a switch and a corresponding serializer. In this brief section we show that last Section's results apply also for serializers. That will prove that in both cases the same choice for K and P_S leads to the same rate tuple Ψ and, indirectly, that the results of Section III also apply for generalized time-sharing.

It is sufficient to show that the conclusion reached in (24), namely that

$$\Psi_i = \sum_s P_S(s) I(X_i; Y X_{\mathcal{L}(v(K,s),i)}), \tag{26}$$

holds also for a serializer. The rest of Section IV then follows by passing to vector notation. As we did then, consider for a moment fixing $S = s$. Letting $v = v(K, s)$, the active constituent decoder of user i observes Y and $X_{\mathcal{L}(v,i)}$. Hence, the active encoder of user i communicates to the corresponding decoder through channel $W_{YX_{\mathcal{L}(v,i)}|X_i}$. The mutual information of this channel is $I(X_i; YX_{\mathcal{L}(v,i)})$. Now let S be random and let us average over the possible values of S *conditioned on* $S_i = j$. Under this condition the channel seen by encoder/decoder pair (i, j) is the same (at all times) for both the switch and the serializer. This channel is a DMC of mutual information

$$\sum_s P_{S|S_i}(s|j) I(X_i; YX_{\mathcal{L}(v(K,s),i)}).$$

The difference between a switch and a serializer enters into play when $S_i \neq j$. In the former case, the mutual information of the channel seen by $X_{(i,j)}$ is 0. In the latter, the clock of $X_{(i,j)}$ is not advanced. In both cases, the contribution to the average rate of information going through virtual input (i, j) is the same and equal to the desired result (26).

We conclude this section on generalized time-sharing by pointing out two differences with respect to the usual form of time thought in information theory as a method to approach any point in the capacity region. The first difference is that the usual time-sharing requires synchronization (even if it is done to achieve a point in the asynchronous capacity region). The second is that the usual time-sharing requires up to M^2 codebooks. This is so since any point in the dominant face is in the convex hull of at most M vertices, and to approach any vertex one needs M codebooks, one for each user. On the other hand generalized time-sharing requires no synchronization and at most $2M - 1$ codebooks.

VI Conclusion

We have introduced time-splitting multiple-access. The idea is to split one or more inputs to a multiple-access channel[7] to create multiple virtual inputs per original input. Each virtual input is fed with symbols from an encoder. The virtual inputs take turns in using the original input. We have discussed two versions, called splitting by switching and generalized time-sharing, respectively. They differ by what we do with codeword symbols of a virtual input when such inputs do not have access to the original input. In the former case we advance the clock and loose the symbols. In the latter we freeze the clock (i.e. store the symbols) until they can be sent through the channel.

In all cases successive decoding is assumed. This implies the existence of a a constituent decoder for each virtual input and an ordering telling in which sequence virtual inputs are

[7] point-to-point channels are included as as special case.

decoded. A constituent decoder receives, as side information, the estimates of all already decoded virtual inputs.

The advantage of successive decoding is that one avoids the complexity of having to decode all inputs simultaneously. The complexity of such a joint decoder could be the main reason for the slow progress in the implementation of multiple-access techniques that are optimal in the usual information theoretic sense, i.e., in the sense of allowing one to approach any rate in the capacity region of the channel at hand. It is the daunting complexity of an optimal[8] joint detector that has lead to the new research field of multiuser detectors [20]. It should be noticed that multiuser detectors are the multiple-access counterpart of hard decoders used in point-to-point channels: such detectors decide on a symbol-by-symbol basis neglecting coding. The performance of a multiuser detector is evaluated based on the error probability. Being faced with the unmanageable complexity of the optimal decoder, an alternative to multiuser detectors is that of suboptimal multiuser decoders. The goal now is no longer that of guaranteeing certain error probabilities but it of guaranteeing that the rate tuple of interest be inside the capacity region of the channel that incorporates our engineering choices. As we have seen this can always be guaranteed via input splitting (as done in this paper or as in [8]) and successive decoding. The P_e requirements can then be fulfilled with sufficiently powerful point-to-point codes. It should also be pointed out that trading an optimal decoder with a suboptimal decoder that does not reduce the capacity region is an old trick that goes back to Shannon's original paper. Indeed Shannon's typical sequence decoders is the best example of a suboptimal yet capacity achieving decoder (see e.g. [21, Section 8] for an in-depth approach using typical sequence decoders.)

What about the impact of the side information on the decoding complexity? The decoding complexity depends, of course, on the particular codes used by each virtual input. If one uses a Viterbi decoder, then the complexity does not depend on the channel on which the code is used (we are assuming memoryless channels). Nor does the decoding complexity of the Viterbi decoder (measured as usual by the number of add select compare operations) depend on the amount of side information received from the other decoders. The side information merely changes the metric used to label trellis transitions. In this sense the decoding complexity is linear in the number of virtual inputs which, as we have seen, never needs to be more than $2M-1$, while M is the number of original inputs.

Time-splitting multiple-access is closely related to rate-splitting multiple-access [10, 22, 8]. There are (at least) two distinguishing features of time-splitting multiple-access. The first is the simple relationship between the design parameter P_S and the resulting rate (see (25)). In particular, the portion of time that a virtual input is active acts linearly on the resulting rate (see (25) and also Examples 2 and 3). The second is that time-splitting multiple-access does not significantly change the statistics of the original channel. This is particularly true for generalized time-sharing and can be best seen considering a single-user

[8] In the sense of minimizing the error probability.

noiseless binary channel. As described in Subsection II-A, generalized time-sharing creates two parallel channels which are identical to the original channel. Hence no coding is needed before or after splitting. This is not the case with the splitting function used in [8] (see the Noiseless Binary Channel example (Example 7) of [8]).

APPENDIX

A Review of Key Facts

(One more careful pass is needed).

When we do stripping (on a multiple access channel), we say that we consider other users as "noise." The meaning of this is clear when the channel is the AWGN channel and codewords are generated randomly from a Gaussian distribution. For the AWGN channel it makes sense to view other users as noise even if we remove the assumption that the code is random. This is so since one can show that to approach the capacity of the point-to-point channel seen by each user, the process obtained by the transmitted codeword and the additive white Gaussian noise is that of an i.i.d. Gaussian random variable.

For a general discrete memoryless multiple-access channel, what other users do to us (the user of interest) is to select the point to point channel that we "see" at each instant in time. Hence, viewing other users as noise means transmitting over a channel in which the random event that maps inputs to outputs has been additionally "randomized" by the selection made by other users. The purpose of this section is to investigate this phenomenon in more details and derive the information theoretic limits associated with it. We use bold letters to denote n-length sequences.

Fact 1 *(The average DMC)* Assume that a single-user DMC is selected at each time from a parametric family of DMCs. Let the input and output alphabets be the same for all channels in the family and assume that the selection is i.i.d. To be specific, consider the family $\mathcal{W} := \{W_s(y|x) : s \in \mathcal{S}\}$ of channels and let

$$Pr\{\boldsymbol{Y} = \boldsymbol{y}|\boldsymbol{X} = \boldsymbol{x}, \boldsymbol{S} = \boldsymbol{s}\} = \prod_i W_{s_i}(y_i|x_i)$$

be the channel transition probability for n-length sequences when the ith input sees channel W_{s_i}. If the channel sequence is selected according to a product distribution $Q(\boldsymbol{s}) = \prod Q(s_i)$ which is independent of the channel input sequence X, then the channel transition probability becomes

$$\begin{aligned} Pr\{\boldsymbol{Y} = \boldsymbol{y}|\boldsymbol{X}\boldsymbol{x}\} &= \sum_{\boldsymbol{s}} Pr\{\boldsymbol{S} = \boldsymbol{s}|\boldsymbol{X} = \boldsymbol{x}\} Pr\{\boldsymbol{Y} = \boldsymbol{y}|\boldsymbol{X} = \boldsymbol{x}, \boldsymbol{S} = \boldsymbol{s}\} \\ &= \sum_{\boldsymbol{s}} \prod_i Q(s_i) W_{s_i}(y_i|x_i) = \prod_i \sum_{s_i} Q(s_i) W_{s_i}(y_i|x_i) \\ &= \prod W(y_i|x_i) \end{aligned}$$

where $W(y|x) = \sum_{s \in \mathcal{S}} W_s(y|x)Q(s)$. Hence the channel is a DMC with transition probability $W(y|x)$. What we see is the average DMC, averaged over the selection process. The coding theorem for DMCs asserts that the supremum of all achievable rates is $\max_P I(P, W)$. Observe that this is less than the average mutual information in general. This follows from the convexity of $I(P; W)$ with respect to W:

$$I(X;Y) = I(P;W) \leq \sum Q(s) I(P;W_s).$$

(Notice that it is not a problem if the constituent channels do not have the same output alphabet since we may always extend the output alphabet of each constituent channel to be the union of all output alphabets.)

Fact 2 *(Receiver Side Information)* Now consider the same situation but assume that the i.i.d. random sequence $\boldsymbol{S}$ is *known* at the receiver (but *unknown* at the transmitter). Now the channel is characterized by

$$\begin{aligned} Pr\{\boldsymbol{Y} = \boldsymbol{y}, \boldsymbol{S} = \boldsymbol{s} | \boldsymbol{X} = \boldsymbol{x}\} &= Pr\{\boldsymbol{S} = \boldsymbol{s} | \boldsymbol{X} = \boldsymbol{x}\} Pr\{\boldsymbol{Y} = \boldsymbol{y} | \boldsymbol{S} = \boldsymbol{s}, \boldsymbol{X} = \boldsymbol{x}\} \\ &= Q(\boldsymbol{s}) Pr\{\boldsymbol{Y} = \boldsymbol{y} | \boldsymbol{X} = \boldsymbol{x}, \boldsymbol{S} = \boldsymbol{s}\} \\ &= \prod Q(s_i) W_{s_i}(y_i|x_i) \\ &= \prod Q(s_i) W(y_i|x_i, s_i) \\ &= \prod W(y_i, s_i|x_i), \end{aligned}$$

where we defined $W(y_i|x_i, s_i) = W_{s_i}(y_i|x_i)$ and $W(y_i, s_i|x_i) = Q(s_i)W(y_i|x_i, s_i)$. Again, this is a DMC. The mutual information between input and output is determined by

$$I(X; YS) = I(X; S) + I(X; Y|S) = I(X; Y|S) = \sum I(P; W_s) Q(s) \tag{27}$$

where we used the fact that X and S are independent random variables and thus $I(X; S) = 0$. The key point is that if the channel-selecting random variable is available at the receiver, then the receiver sees the constituent DMCs rather than just the average DMC and the supremum of the achievable rates becomes the average mutual information.

Fact 3 *(Transmitter and Receiver Side Information)* A third and best situation (in terms of maximizing the throughput) arises when the transmitter is also informed of the channel parameter. In this case the transmitter sees a number $|\mathcal{S}|$ of channels and gets to use channel W_s with probability $Q(s)$. On channel W_s one can achieve $\max_{P_s} I(P_s; W_s)$. Hence the maximal achievable rate becomes

$$\sum_s Q(s) \max_{P_s} I(P_s; W_s). \tag{28}$$

To approach this rate one in general needs $|\mathcal{S}|$ codebooks. If the maximizing distribution P_s is the same for all s, or if we are forced to use constant composition codes and the

composition has to be the same for all codebooks, then knowing the channel selection parameter s at the transmitter does not increase the rate at which reliable transmission is possible. The above expression may be rewritten in various forms

$$\begin{aligned}
&\sum_s Q(s) \max_{P_s} I(P_s; W_s) \\
&\qquad = \sum_s Q(s) \max_{P(x|s)} I(X;Y|S=s) \\
&\qquad = \max_{P(x|s)} \sum_s Q(s) P(x|s) I(X=x;Y|S=s) \\
&\qquad = \max_{P(x|s)} I(X;Y|S).
\end{aligned}$$

Notice that in both of the last two situations the quantity of interest is $I(X;Y|S)$: when only the receiver knows the channel selection then we maximize over $P(x)$ whereas if the transmitter and the receiver know the channel selection then we maximize over $P(x|s)$.

References

[1] B. Rimoldi, "Generalized time sharing for multiple access channels," in *IEEE International Symposium on Information Theory*, (Ulm, Germany), p. 26, June 29 - July 4 1997.

[2] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error correcting coding and decoding: turbo-codes," in *Proceeding of ICC '93*, (Geneva), pp. 1607–1070, May 1993.

[3] T. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message-passing decoding," tech. rep., Bell Labs, Lucent Technologies, 1998.

[4] R. G. Gallager, *Low-density parity-check codes*. Cambridge, Massachusetts: M.I.T Press, 1963.

[5] J. L. Massey and P. Mathys, "The collision channel without feedback," *IEEE Trans. Inform. Theory*, vol. IT–31, pp. 192–204, Mar. 1985.

[6] S. C. Chang and E. J. Weldon, "Coding for T-user multiple-access channels," *IEEE Trans. Inform. Theory*, vol. IT–25, pp. 684–691, Nov. 1979.

[7] P. Mathys, "A class of codes for a T active users out of N multiple-access communication system," *IEEE Trans. Inform. Theory*, vol. IT–36, pp. 1206–1219, Nov. 1990.

[8] A. Grant, B. Rimoldi, R. Urbanke, and P. Whiting, "Rate-splitting multiple access for discrete memoryless channels," *IEEE Trans. Inform. Theory*, To appear.

[9] A. J. Grant and L. K. Rasmussen, "A new coding method for the asynchronous multiple access channel," in *Proceedings of the 33d Allerton Conference on Communication, Control, and Computing*, (Monticello, IL), 1995.

[10] B. Rimoldi and R. Urbanke, "A rate-splitting approach to the Gaussian multiple-access channel," *IEEE Trans. Inform. Theory*, vol. IT–42, pp. 364–375, Mar. 1996.

[11] E. M. Yeh and R. G. Gallager, "Achieving the multiple access capacity region via projective time sharing," in *Proceedings 1998 IEEE International Symposium on Information Theory*, (M.I.T, Cambridge, MA USA), p. 213, 16-21 August 1998.

[12] L. Duan, B. Rimoldi, and R. Urbanke, "Approaching the AWGN channel capacity without active shaping," in *IEEE International Symposium on Information Theory*, (Ulm, Germany), p. 374, June 29 - July 4 1997.

[13] J. G. D. Forney, "Approaching the capacity of the AWGN channel with coset codes and multilevel coset codes," *IEEE Trans. Inform. Theory*, 1996. Submitted.

[14] Y. Kofman, E. Zehavi, and S. S. (Shitz), "Performance analysis of a multilevel coded modulation system," *IEEE Trans. Commun.*, vol. 42, pp. 299–312, Feb. 1994.

[15] J. Huber and U. Wachsmann, "Capacities of equivalent channels in multilevel coding schemes," *Elect. Lett.*, vol. 30, pp. 557–558, Mar 1994.

[16] G. S. Poltyrev, "Coding for channel with asynchroneous multiple access," *Probl. Peredachi Informatsii*, vol. 19, pp. 12–21, 1983.

[17] J. Y. N. Hui and P. A. Humblet, "The capacity region of the totally asynchronous multiple-access channel," *IEEE Trans. Inform. Theory*, vol. IT–31, pp. 207–216, Mar. 1985.

[18] R. Ahlswede, "Multi-way communication channels," in *Proc. 2nd Int. Symp. Inform. Theory*, (Tsahkadsor, Armenian S.S.R., 1971), pp. 23–52, 1973. Hungarian Academy of Science.

[19] H. Liao, *Multiple Access Channels*. PhD thesis, Department of Electrical Engineering, University of Hawai, 1972.

[20] S. Verdu, *Multiuser Detection*. Cambridge University Press, 1998.

[21] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

[22] A. Grant, B. Rimoldi, R. Urbanke, and P. Whiting, "On single-user coding for the discrete memoryless multiple-access channel," in *Proceedings 1995 IEEE International Symposium on Information Theory*, (Whistler, B.C. Canada), p. 448, September 17-22, 1995.