Synchronization Strategies for RFI Channels

R. J. McEliece
Communications Systems Research Section

H. van Tilborg
California Institute of Technology

S. Tung
Cornell University

In this article we define an RFI channel to be a multiple-access channel in which no sender can know when any other starts, and study the problem of determining the relative phases of the senders at the receiver. Along the way we prove a new result about binary deBruijn sequences.

I. Introduction

As the electromagnetic spectrum becomes increasingly crowded, it becomes more and more important to study the possibilities of cooperation between individuals who must share the same channel. In the information-theory literature, there is a class of channel models designed for such studies called *multiple-access* channels. Unfortunately, a basic assumption about these channels is that all senders begin transmission simultaneously. Recently (Ref. 2) we have begun to study multiple-access channels without making this assumption; we call these channels RFI (radio frequency interference) channels. We now describe the simplest kind of RFI channel, the two-input, one-output, discrete memoryless RFI channel.

Such a channel has two inputs $x^{(1)} \epsilon A_1$, $x^{(2)} \epsilon A_2$, and one output $y \epsilon B$, where A_1, A_2, and B are finite sets. The transition probability $p(y|x^{(1)}, x^{(2)})$ represents the probability that y will be the output, given that $x^{(1)}$ and $x^{(2)}$ are the two inputs. Figure 1 gives the appropriate block diagram. The sources are assumed independent. Note that since the channel has only one output, there is no loss in generality in assuming a single decoder, although there may actually be two receivers located in physically different places.

We assume that the two senders may agree ahead of time on the strategies they will use, but that neither can know when the other will begin transmission. In Ref. 2 we assumed that the receiver can always determine when each sender begins transmission either by a separate synch channel or by the design of suitable synchronization prefixes. In this paper we shall study the second method of obtaining joint synchronization.

II. An Example

Consider the two-input, one-output RFI channel described in Table 1. The two inputs are labeled X_1 and X_2, and the corresponding output, Y. Notice that if $X_1 = X_2$, then $Y = X_1 = X_2$; but if $X_1 \neq X_2$, the output Y is the special erasure symbol "?". It can be shown that the capacity region of this channel is given by Fig. 2. This means that for any pair of rates (R_1, R_2) lying in the shaded region of Fig. 2, and any $\epsilon > 0$, there exists a coded communication scheme for the channel such that sender No. 1 has rate $> R_1 - \epsilon$, sender No. 2 has rate $> R_2 - \epsilon$, and the overall error probability is less than ϵ. And while it is not necessary to assume the two senders begin transmission simultaneously, it is necessary to assume that the receiver can determine the relative phases of the two senders, in order to prove the theorem. This section will describe a simple and generalizable technique for establishing synchronization at the receiver for this channel.

We assume sender No. 1 begins his transmission with the prefix P_1, and sender No. 2 begins with P_2, where

$$P_1 = 1\,1\,0\,0\,1\,0\,1\,1\,1\,1\,0\,1\,0\,0\,0\,0$$

$$P_2 = 1\,0\,1\,1\,0\,1\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0$$

We further assume that after he sends P_1, sender No. 1 sends his information in blocks of 11 bits followed by 4-bit blocks of zeros called *windows*; ditto for No. 2. The overall transmission strategies are depicted in Fig. 3. We shall now show that if these strategies are used, then the receiver will be able to synchronize with both senders on the basis of the received sequence.

First we assume that the two senders begin transmission nearly simultaneously, say within three bits times of each other. Then the receiver will be able to synchronize with both senders on the basis of the first three received symbols, for there are only five possibilities and they all yield different initial received sequences:

P_1	110···	110···	110···	11···	1···
P_2	1···	10···	101···	101···	101···
Received	???···	?10···	1??···	??1···	?01···

(We assume that when the channel is not in use, zeros are being transmitted; by "the first three received symbols" we mean beginning with the first nonzero one.) Hence if the first three symbols are in the set {???, ?10, 1??, ??1, ?01}, synchronization is immediately established. The only other possibilities for the first three symbols are ??0, signifying that No. 1 has begun transmission, but giving no information about No. 2, or ?0?, signifying that No. 2 has begun, but giving no information about No. 1. In either of these two cases, it is more difficult but still possible to establish synchronization with the remaining sender.

Let us suppose that the first three symbols received are ??0. Synchronization with No. 1 is now established, but nothing is known about No. 2. The idea is to look through the 4-bit "windows" in No. 1's transmitted stream in order to locate No. 2. For a while, perhaps, these windows will be blank, indicating that No. 2 still has not begun transmission. But when the 16-bit prefix is finally sent, a 4-bit substring of it will appear through the window. And the prefix P_2 is a *deBruijn sequence*, which means that each of its 16 4-bit substrings taken cyclically are distinct; thus the first nonzero window will enable the receiver to synchronize with No. 2.

For example, suppose the first nonzero 4-bit window in the received stream is 000?. This must correspond to the substring 0001 of P_2, and means that No. 2 began its transmission right at the end of the window. Similarly if ??0? is received, the substring is 1101; in which case No. 2 began transmission two bits prior to the window.

If sender No. 2 began transmission first, the procedure is the same, because the prefix P_1 is also a deBruijn sequence. Thus we have shown that joint synchronization can be established in every case. In the next section we will show that this strategy can be generalized.

III. The Generalization

In this section we will generalize only to binary, two-input, noiseless RFI channels, leaving further generalizations to a later paper. Up to obvious equivalence there are only five nontrivial such channels (Ref. 2), as given in Table 2. In the notation of Table 2, the channel discussed in Section II is channel C.

There are two problems to be solved in generalizing the approach of Section II. First, we must design the initial portion of the prefixes so that nearly simultaneous starts can be detected; and second, we must show that these initial portions can be extended to deBruijn sequences.

By trial and error we have found such pre-prefixes for each of the five channels in Table 2, as given in Table 3. The reader is invited to verify for himself that each pair of pre-prefixes has the property that, if the senders start nearly simultaneously, in the sense that their pre-prefixes overlap, then joint synchronization can be established on the basis of the first few received symbols.

The general strategy will be to extend a pre-prefix of length n to a deBruijn sequence of length $N = 2^m$ ending in $m + 1$ zeros, and then to insert windows of length $m + 1$ between data blocks of length $2^m - m - 1$. The analysis of Section II can then be extended to show that joint synchronization can be established. However, it is not obvious that an arbitrary string of length n can be extended to such a deBruijn sequence. In the next section we will show that any binary string of length n can be extended to a deBruijn sequence of length 2^m which ends with m zeros, provided only that $m \geqslant n + 1$. Since the "overhead" imposed on the transmission scheme by the inclusion of the windows is $(m + 1)/(2^m - 1)$, this result shows that the decrease in data rate required to establish joint synchronization can be made arbitrarily small.

IV. A Result about deBruijn Sequences

A *deBruijn sequence* of length 2^m is a sequence of 2^m 0s and 1s, such that when the sequence is viewed cyclically, each of the 2^m substrings of length m are distinct. These sequences exist for all values of $m \geqslant 1$; we list a few short ones below:

m	one deBruijn sequence of length m
1	0 1
2	0 0 1 1
3	0 0 0 1 0 1 1 1
4	0 0 0 0 1 0 0 1 1 0 1 0 1 1 1 1

There is a useful graphical description of deBruijn sequences which we now give.

Let G_m be the directed graph whose vertices are the 2^{m-1} binary strings of length $m - 1$. In G_m there is a directed edge going from $\mathbf{v} = v_1 v_2 \cdots v_{m-1}$ to $\mathbf{v}' = v_1' v_2' \cdots v_{m-1}'$ iff there is a binary string $\mathbf{e} = e_0 e_1 \cdots e_{m-1}$ of length m such that $\mathbf{v} = e_0 e_1 \cdots e_{m-2}$, $\mathbf{v}' = e_1 e_2 \cdots e_{m-1}$. This means that the edge set of G_m can be identified with the set of binary strings of length m. The graph G_3 is illustrated in Fig. 4. Any binary string $\mathbf{s} = s_1 s_2 \cdots s_M$ of length $M \geqslant m$ can be viewed as a closed path (circuit) of length M in G_m, viz. the path passing successively through the edges $(s_1 s_2 \cdots s_M)$, $(s_2 \cdots s_{m+1})$, $\cdots$, $(s_{M-m+1} \cdots s_M)$, $(s_{M-m+2} \cdots s\, s_1)$, $\cdots$, $(s_M s_1 \cdots s_{m-1})$. In particular, s is a deBruijn sequence if $M = 2^m$ and the corresponding circuit uses each edge of G_m exactly once. A circuit in a graph using each edge exactly once is called an *Euler circuit*; thus there is a one-to-one correspondence between cyclically distinct deBruijn sequences and Euler paths in G_m.

The theorem we wish to prove in this section is that given any binary string s of length $m - 1$ (or less) beginning with a 1, there exists a deBruijn sequence of length 2^m, beginning with s, and ending with a string of m zeros. Alternatively, since a deBruijn sequence is viewed cyclically, our assertion is that, given any such s, there is a deBruijn sequence beginning with 0^ms. In terms of the graph G_m, our result is as follows.

Theorem: Any path $P = [\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_{m-1}]$ of length $m - 1$ in G_m, with $\mathbf{v}_0 = (00 \cdots 0)$ and $\mathbf{v}_1 = (00 \cdots 01)$, can be completed to an Euler circuit in G_m.

Proof: Let $\mathbf{E}$ be the binary string of length $2m - 2$ obtained by concatenating $\mathbf{v}_0$ with $\mathbf{v}_{m-1}$. Then the vertices in the path P are the m substrings of $\mathbf{E}$ of length $m - 1$, and the edges of P are the $m - 1$ substrings of $\mathbf{E}$ of length m. $\mathbf{E}$ will have the general form illustrated in Eq. (1) for $m = 6$.

$$\mathbf{E} = (\overset{\longleftarrow m-1 \longrightarrow}{0\ 0\ 0\ 0\ 0}\ \overset{\longleftarrow m-1 \longrightarrow}{1\ x\ x\ x\ x}) \tag{1}$$

From Eq. (1) it is clear that the m vertices and the $m - 1$ edges in P are all distinct. We now form a new graph G_m' by removing the $m - 1$ edges of P from G_m and replacing them with a new edge $\mathbf{e}'$ which joins $\mathbf{v}_0$ to $\mathbf{v}_{m-1}$ directly. This construction is illustrated in Fig. 5 with $m = 4$, $P = [(000), (001), (011), (110)]$.

The graph G_m' is clearly *balanced*, i.e. every vertex has the same number of edges going in as coming out. (This number is two except for the vertices $\mathbf{v}_1, \cdots, \mathbf{v}_{m-2}$; for these vertices it is 1.)

The graph G_m' is also *connected*. To see this, observe that for any vertex $\mathbf{v}$ in G_m', there is always a path of length $m - 1$ from $\mathbf{w} = (111 \cdots 1)$ to $\mathbf{v}$, where edges are the $m - 1$ substrings of the string $\mathbf{F}$ formed by concatenating $\mathbf{w}$ and $\mathbf{v}$. For example in Fig. 3 if $\mathbf{v} = 101$, $\mathbf{F} = 111101$, and the path is (111), (111), (110), (101). Note that these edges are not among those deleted in the construction of G_m', since the first bit in each edge is 1, whereas the first bit in each deleted edge is 0.

Since G_m' is balanced and connected, by a theorem of I. J. Good (see Ref. 1, Sec. 2.3.4.2, for example) G_m' possesses an Euler circuit, which must perforce contain the dummy edge $\mathbf{e}'$. In Fig. 3 such a path is (110), (100), (001), (010), (101), (011), (111), (111), (110), (101), (010), (100), (000), (000), (110), the last edge corresponding to the dummy edge e′. If now we replace the dummy edge e′ with the original path P in G_m, the result is an Euler circuit in G_m, which has the desired property. For example, the path so constructed in Fig. 3 is 0001100101111010, which yields the prefix P_1 = 1100101111010000 of Section II. QED.

References

1. Knuth, D. E. *The Art of Computer Programming. Vol. 1, Fundamental Algorithms.* Addison-Wesley, Reading, Mass., 1968.
2. McEliece, R. J. and Rubin, A. L. "Timesharing without Synchronization," in the *Proceedings of the 1976 International Telemetering Conference*, McGregory Werner, Los Angeles, 1976.

Table 1. A simple RFI channel

X_1	X_2	Y
0	0	0
0	1	?
1	0	?
1	1	1

Table 2. The five binary, two input, noiseless RFI channels

		Y				
X_1	X_2	A	B	C	D	E
0	0	0	0	0	0	0
0	1	1	1	2	1	1
1	0	2	2	2	1	1
1	1	3	2	1	0	1

Table 3. Pairs of pre-prefixes for establishing synchronization for nearly simultaneous starts

Channel	Pre-prefix 1	Pre-prefix 2
A	1	1
B	1000	1011
C	110	101
D	11011	10101
E	100011000	111000000

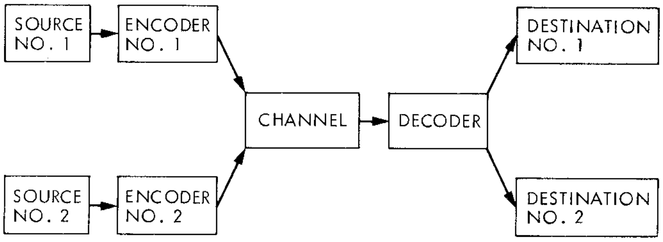

Fig. 1. An RFI channel

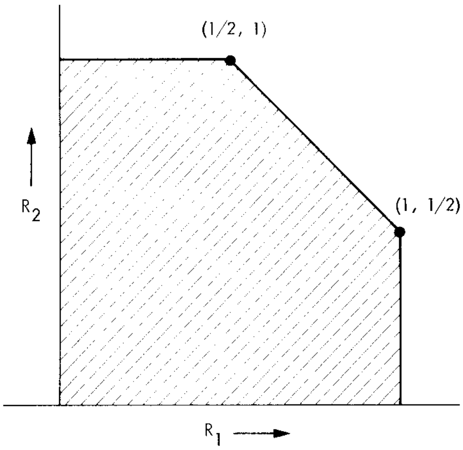

Fig. 2. The capacity region of the RFI channel described in Table 1

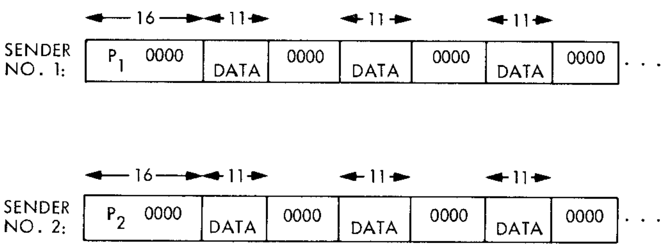

Fig. 3. The transmission strategies

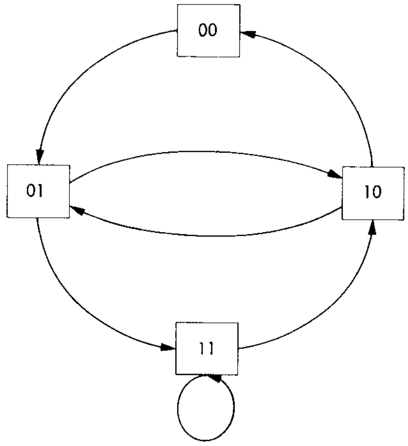

Fig. 4. The graph G_3

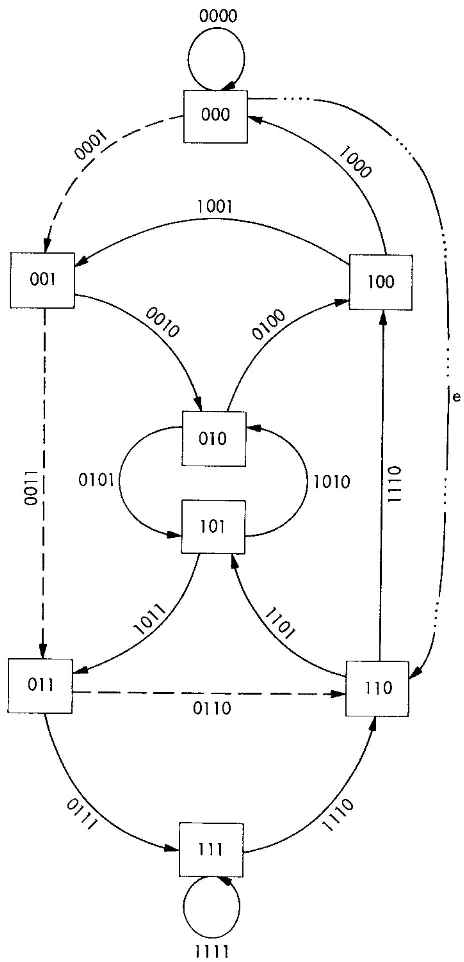

Fig. 5. The graph G'_m, with m = 4