ON THE PRINCIPAL STATE METHOD
FOR RUNLENGTH LIMITED SEQUENCES.

Tjalling Tjalkens
Eindhoven University of Technology,
P.O.Box 513, 5600 MB Eindhoven,
The Netherlands.

Abstract.
We present a detailed result on Franaszek's principal state method for the
generation of runlength constrained codes. We show that, whenever the
constraints $k$ and $d$ satisfy $k \geq 2d > 1$, the set of "principal states" is
$s_0, s_1, \ldots, s_{d-1}$. Thus there is no need for Franaszek's search algorithm
anymore. The counting technique used to obtain this result also shows that
"state independent decoding" can be achieved using not more than
three codewords per message and it allows us to compare the principal state
method with other practical schemes originating from the work of Tang and
Bahl and also allows us to use an efficient enumerative coding implementa-
tion of the encoder and decoder.

Introduction.
Shannon [4] considers the $(d,k)$-constrained channel, where the only posi-
tible binary sequences that can be transmitted over the channel are those
containing runs of zeros of length $d, \ldots, k$ ($d < k$). These channels can be
described by a state model where each state is indexed by the length of the
current run of zeros. Shannon defines the capacity of this channel as the
limit as $n \to \infty$ of the logarithm of the size of the set of all sequences of
length $n$ satisfying the $(d,k)$-constraint divided by $n$.

A runlength constrained code, $(d,k)$-constrained code, is a binary encoding
of information such that in the code sequence successive ones are separated
by at least $d$ zeros and at most $k$ zeros and this is well suited for use on a
$(d,k)$-constrained channel.

We shall consider fixed length codes for these purposes. Valid codewords
follow a possible path in this state model, starting at the state where the
previous codeword ended. So, a code for this state model contains sev-
eral codewords sets, each containing a variable number of words, where the
selected set depends on the previous codeword and is such that the concate-
nation of that codeword with any word in the set is permissible. Since we
consider fixed length codes the size of the code is determined by the smallest
set belonging to some state in the model.

Frazhnszek [3] noted that if we take a subset of all states in the model and
require the codewords to start and end in states of this subset then an
optimum subset exists. This subset is known as the set of principal states
and Franaszek described an algorithm to search for these principal states.

Another approach, presented by several authors, [1, 5, 6], is to use a single
set $S$ of codewords that satisfy the $(d,k)$-constraint internally. A special
sequence is put in between two codewords such that the $(d,k)$-constraint
remains satisfied between codewords.

The principal state method is an optimal code for systems that can be de-
scribed in the state model framework, and thus it is at least as efficient
as any of the glue methods, since the glue methods can also be described in
the state model framework.

The principal states.
We start with the definition of the building blocks or basic sets $U(m)$ for the
codewords given the $(d,k)$-constraint, containing all sequences that
start and end with a "one" and satisfy the $(d,k)$-constraint internally. Let
$U(m)$ denote the size of $U(m)$.

In the following we shall repeatedly use the shorthand notation $[a;b] \triangleq
\{a, a+1, \ldots, b\}$.

Let $S \subseteq [0;k]$ denote the set of permitted channel states, (not necessarily
the set of principal states). Consider the sets $V_S(n;i)$ containing all sequences
starting with a run of zeros and ending in a run of $r \in S$ zeros and satisfying
the $(d,k)$-constraint internally. Note that $V_S(n;i)$ can be described using
the basic sets as

$$V_S(n;i) = \bigcup_{i \in S} \{0^i\} \cup U(n-i-j) \times \{0^j\},$$

where $U \cup V$ indicates the set containing all concatenations of the sequences $z \in U$ with any sequence $y \in V$.

With these sets we can make Franaszek's state depending codeword sets
$W_S(n;i)$, i.e. the set of possible codewords of length $n$ starting in state
$i \in S$ and ending in any state $j \in S$. We have

$$W_S(n;i) = \bigcup_{0 \leq i \leq \min(n,k-1)} V_S(n;i).$$

Now we can formulate our goal and the result:

Given $n$, $k$, and $d$, (with the restriction $n \geq k \geq 2d > 1$), find the set $S^* \subseteq [0;k]$ such that:

$$W_{S^*}(n) \supseteq \bigcup_{0 \leq i \leq \min(n,k-1)} V_{S^*}(n;i).$$

Message mapping for state independent decoding

Partition the $W_{S^*}$ messages into sets $M_i$ of size $M_i = |W_{S^*}(n;i+d)|$, where
$i = 0, 1, \ldots, k-1$. Let $r$ be the number of trailing zeros in the previous
codeword. We distinguish between the following cases:

$d = 1$ and $r = 0$: The messages in the set $M_i$ are assigned to the set
$X_{S^*}(n; i+1)$.

d = 1$ and $r \geq 1$: The set $M_i$ is assigned to $X_{S^*}(n; i+1)$ and
$M_{i-1} \cup \ldots M_{i-r}$ are assigned to $X_{S^*}(n; 0)$.

d > 1$ and $r < d$: For all $i = 0, \ldots, k-d-r-1$, $d-r$ we assign the set $M_i$ the
codewords from $X_{S^*}(n;i+1)$ respectively. For $i = k-d-r+1, \ldots, k-d$ we
assign to the set $M_i$ the codewords from $X_{S^*}(n;i+2d-k+1)$ respectively.

d > 1$ and $r \geq d$: The sets $M_{i-1} \cup \ldots M_{i-r}$ are assigned to $X_{S^*}(n; 0)$
and $X_{S^*}(n; 1)$ in that order.

So, it is easy to see that every message is encoded into one of two or three
different codewords, depending on $r$.

Enumerative coding.

We shall briefly indicate the application of the well-known enumerative cod-
ing technique [2] to the generation of the $(d,k)$-constrained sequences.

First we determine the message subset $M_i$ of the message $m$ that we want
to transmit. Then, with the rules of the previous section we determine the set
$X_{S^*}(n; j)$ and the relative index $i(g_n)$ of our message in the set.

Finally we use the enumerative reconstruction to produce the codeword $g_n^* \in
X_{S^*}(n; j)$ from its index.

Let the codeword $g_n$ be given as $g_n = 0^{a_0}1^{a_1} \ldots 0^{a_r}$. So $a_0 = j$.

Although we will not need the (source) encoding algorithm, it is instructive to
see how the index can be computed recursively as

$$i(g_n; X_{S^*}(n; j)) = (i_0, \ldots, i_r),$$

where $i_0 = a_0$, $i_r = a_r$, and $a_{r-1} = j$.

Note that this computation produces a lexicographical ordering given the
symbol ordering $"0" \prec "1"$. Also note that in order to compute the index
we only need the $n+1$ numbers $|X_{S^*}(n; 0)|$ for $0 \leq p \leq n$.

Reconstructing $g_n$ involves producing the $a_0, \ldots, a_r$, and they can be found
recursively by the corresponding enumerative decoding algorithm.

References.
for encoding and decoding run-length-limited binary sequences," IEEE


