ΠP periodica polytechnica

Electrical Engineering
53/1-2 (2009) 31–36
doi: 10.3311/pp.ee.2009-1-2.04
web: http://www.pp.bme.hu/ee
© Periodica Polytechnica 2009

RESEARCH ARTICLE

A joint coding concept for runlength and charge–limited channels

Péter Vámos

Received 2010-05-22

Abstract

By making the conventional (d, k) constraint time dependent as a function of the channel process, the wide sense RLL channel has been defined. With the help of the new concept several existing constraints can be described alternatively and many new ones can be constructed. A bit stuff algorithm is suggested for coding wide sense RLL channels. We determine the rate of the bit stuff algorithm as the function of the stuffing probability. We present a few examples for calculating the rate of different constrained codes complying with the newly introduced constraint.

Keywords

Runlength limited codes · constrained codes

Péter Vámos

Department of Telecommunications and Media Informatics, BME, H-1521 Budapest, Hungary

e-mail: vamos@tmit.bme.hu

1 Introduction

The sequence of independent identically distributed (i.i.d.) alternatives as coin tossing has been subject of scientific study since Jacob Bernoulli [1], and it has earned a special importance along with the development of digital communications and computer science where binary streams are ubiquitous. A binary sequence can be taken as consecutive strings of identical symbols, referred to as runs, and can be described by the sequence of their length:

Definition *A* run *is a substring of identical symbols. Let us define the transition times of the discrete process X as*

$$t_i = \min\{j > t_{i-1} | X_j \neq X_{j-1}\} \quad \textit{and} \quad t_0 = 0.$$

Then the runlengths are given as the differences $T_i = t_i - t_{i-1}$, and the process $T(X)$ is called as the runlength process *associated with X.*

In order of reliable data recovery the runlength is limited in the most channels. The upper bound, called *k constraint*, is set to ensure the reliable clock recovery [2]. It maximizes the number of consecutive identical symbols in $k + 1$:

$$T_i \leq T_{\max} = k + 1.$$

The lower bound is called *d constraint*. It minimizes the number of consecutive identical symbols in $d + 1$:

$$T_i \geq T_{\min} = d + 1.$$

The d constraint diminishes the intersymbol interference by enlarging the distances between the transitions [3]. It works as if the signalling rate were dropped by d, but for the transitions only, so it less reduces the capacity. The channels with input constraints above and the sequences complying with them are called runlength limited (RLL).

The queer definitions of d and k constraints have historical roots. It stems from that initially they limited only the length of 'zero' runs of the source sequence $X: X_i \in \{0, 1\}$, and then by a transformation using mod2 addition (exor) they formed the RLL channel sequence Y as

$$Y_i = X_i \oplus Y_{i-1}.$$

One can see that the above transformation called precoding [12] turns the 'zero' runs of length n of X into 'zero' and 'one' runs of length $n+1$ of Y alternately. The precoding balances the frequency of binary symbols in the channel sequence: $\Pr(Y=0) = \Pr(Y=1) = 1/2$ even when the source sequence is biased, i.e. $\Pr(X=0) \neq \Pr(X=1)$. However, the channel sequence becomes correlated for any biased sources. Using the symbols $\{+1, -1\}$ rather than $\{0, 1\}$, the channel sequence will have no discrete component at dc, that is why I will use that former convention. With $X_i, Y_i \in \{+1, -1\}$ the precoding can be defined as simple algebraic multiplication:

$$Y_i = X_i\, Y_{i-1}. \tag{1}$$

Besides the runlength, there is frequently set constraint for the accumulated charge [4, 5], or for some of its transforms [6, 7]. For the binary channel $Y:\ Y_i \in \{-1, +1\}$ such constraints can be described as

$$\left| \sum_{i=0}^{\infty} h_i\, Y_{n-i} \right| < c \qquad \text{for any } n \in \mathbf{Z}, \tag{2}$$

where $h_i \in \mathbf{R}$ are given constants. The constraints complying with (2) is referred to as *generalized charge constraint*, and those are set to satisfy some spectral requirement [8]. E.g. with $h_i \equiv 1$ we get the conventional charge constraint, which is the necessary and sufficient condition for the vanishing spectrum at dc [9].

In this paper we introduce the *wide-sense RLL channel*, a generalization of RLL constrained channel. It will be pointed out that many kinds of generalized charge constraints can be translated into the new concept, which makes possible the uniform handling of these constraints. We give a bit stuff algorithm for coding wide-sense RLL channels, and prove a theorem to compute the rate and estimate the channel capacity. Finally we present some examples of application.

2 The wide-sense RLL channel

Let us consider the conventional charge constraint when the accumulated charge is limited in the channel:

$$Q_n = \sum_{i=0}^{\infty} Y_{n-i} \qquad \text{and} \qquad |Q_n| \le c \ \text{ for any } n \in \mathbf{Z}. \tag{3}$$

The amount Q_n is known as running digital sum (RDS) as well. Since Q_n takes only integer values, it will not confine the generality, if we limit the range of c in (3) for the positive integers. If the charge is Q_n at a run's end, the next run should not be longer than $c + Q_n$, if the current run is positive, and $c - Q_n$, if it is negative:

$$T_{\max} = Y_n Q_n + c, \qquad \text{if} \quad Y_{n+1} \neq Y_n\,,$$

so the conventional charge constraint also limits the runlength. This limit varies in time, but depends on the history of the channel only. Using $c-1$ instead of c in (3), the definition of charge constraint will comply with (2). However, it is more convenient to allow equality in (3), since it is easier to express $T_{\max}$, and bound c itself makes the charge threshold of the bitstuff encoder. The (3) is the usual form of the charge (c) constraint [5]. The amount $Y_n Q_n$ we will refer to as rectified running digital sum (RRDS).

Alike $T_{\max}$, the lower bound $T_{\min}$ can be considered time dependent as well. E.g. *window-charge constraint* $\left|\sum_{i=0}^{w-1} Y_{n-i}\right| \le c$ sometimes forces to repeat the last input bit, imposing a temporary lower bound on the runlength [7, section 3.5 of 15].

With these properties several different channels can be constructed. In the following part we will consider the runlength limited channel in this wider sense:

Definition *A channel is RLL in the wide sense if the maximum and the minimum value of the runlength depends on the history of the channel:* $T_{\max} = k(Y) + 1$ *and* $T_{\min} = d(Y) + 1$. ($k(Y) > d(Y)$ *is always required*)

Since during a run the source sends identical bits, so the channel's state is predictable. Consequently, the upper and lower bound of the current run is always determined at the beginning of that run, i.e. by the output runlength process $T^{m-1} = (T_1, T_2, \ldots, T_{m-1})$ and they should remain constant during a run:

$$T_m - 1 \le k(Y^n) = k(Y^{n+1}) = \ldots = k(Y^{n+T_m-1}) = k(T^{m-1})$$

and

$$T_m - 1 \ge d(Y^n) = d(Y^{n+1}) = \ldots = d(Y^{n+T_m-1}) = d(T^{m-1}).$$

3 Coding wide-sense RLL channels

Due to the dependence on the channel history, coding for a wide-sense RLL channel can be easily performed by a bit stuff algorithm [10, 11]. To set the coding rule let us define the function $\mathrm{run}(Y^n)$, which keeps trace of the momentary length of the current output run:

$$\mathrm{run}(Y^n) = \begin{cases} 0, & \text{if} \quad Y_n \neq Y_{n-1}; \\ \mathrm{run}(Y^{n-1}) + 1, & \text{if} \quad Y_n = Y_{n-1}. \end{cases}$$

Then the coding rule for the $X:\ X_i \in \{-1, +1\}$ input sequence reads as

$$Y_{n+1} = \begin{cases} -Y_n, & \text{if} \quad \mathrm{run}(Y^n) = k(Y^n); \\ Y_n, & \text{if} \quad \mathrm{run}(Y^n) < d(Y^n); \\ X_{m+1}\, Y_n, & \text{(no stuffing)} \quad \text{otherwise.} \end{cases} \tag{4}$$

The indices of the input (X) and output (Y) sequences are different because of the previously stuffed bits: $n = m + s_n$, where s_n stands for the number of stuffed bits till Y_n. The coder always closes the current run by inserting a bit with opposite sign whenever the length of the current run can exceed the limit set by the RLL constraint in the next step if the run were continued. And similarly, the coder will insert $d(Y^n)$ bits at the beginning of each run ensuring the minimum runlength. The coding rule $Y_{n+1} = X_{m+1}\,Y_n$ applied for the no stuffing case performs a precoding defined by (1). It ensures that the output will not have discrete component at dc in case of biased input when the probability of "–1" and "+1" bits are not equal.

The decoding can be performed in two steps. First we invert precoding by

$$X'_{n+1} = Y_{n+1}\,Y_n,$$

resulting in a stream which contains the original sequence X padded with stuff bits. To recover the original sequence, we should remove the stuffed bits from X':

– if $\operatorname{run}(Y^n) = k(Y^n)$, remove X'_{n+1};

– if $\operatorname{run}(Y^n) = 0 \;\; (Y_n \neq Y_{n-1})$, remove $X'_{n+1} \ldots X'_{n+d(Y^n)}$.

The decoding process is demonstrated in Fig. 1 with constant parameters $d=1$ and $k=3$.

$Y:\ \ldots+1+1+1-1-1-1-1+1+1+1-1-1+1+1+1+1-1\ldots$

$X':\ \ldots\underline{+1}+\mathit{1}-\boldsymbol{\mathit{1}}\underline{+1+1}+\mathit{1}\underline{-1+1}+\mathit{1}\underline{-1}+\mathit{1}-\boldsymbol{\mathit{1}}\underline{+1+1}+\mathit{1}\underline{-1}\ldots$

$X:\ \ldots+1+1+1-1+1-1+1+1-1\ldots$

Fig. 1. The decoding process for $(d,k) = (1,3)$. The underlined bits form the original sequence, while italic denotes the stuffed bits. The bolded bits are stuffed to start a new run to satisfy the k constraint. The indexing grows from right to left.

Many kinds of generalized charge constraint can be implemented with similar bit stuff algorithm [8], so for those there exists a corresponding $(\,d(Y),\ k(Y)\,)$ wide-sense RLL constraint which imposes the same channel constraint. The reverse statement does not hold: e.g. for d constraint there is no corresponding generalized charge constraint.

About the input process we suppose that it is i.i.d. with $\Pr(X=+1) = p$ and $\Pr(X=-1) = 1-p = q$. Hence – taking the precoding into the account – we are coding an i.i.d. runlength process with geometric distribution.

Theorem *The rate of a bit stuff encoder coding an i.i.d. source for a wide-sense RLL channel is*

$$R = \frac{1-s}{1+q\mathrm{E}[d(T)]-ps}\,, \tag{5}$$

where s is the probability of that a run is closed by stuffing.

Proof: Initially we are going to prove the theorem for the case when $d(T) \equiv 0$. Since runlengths are one at least, and we have no information whether that first bit is stuffed or not, so we are considering the runs one bit shifted, excluding the first bit and including the closing bit which starts a new run with opposite sign (Fig. 2). Accordingly, let T_{in} denote the remaining length of the current "+1" run with the closing "-1" at the input when a new output run has started. Using that T_{in} and $T_{\max}$ are independent, for the stuffing probability we can write:

$$\text{input:}\quad \ldots\ldots\ldots{+1}\underbrace{\ldots\ldots\ldots\ldots{+1}\;{-1}}_{T_{\text{in}}}\ldots\ldots$$

$$\text{output:}\quad \ldots{-1}\;{+1}\;\underbrace{{+1}\ldots\ldots\ldots\ldots{+1}\;{-1}}_{T_{\text{out}}}\;{-1}\ldots$$

Fig. 2. The runs at the coder's input and output.

$$\begin{aligned}
s &= \sum_{i=1}^{\infty} \Pr(T_{\text{in}}=i)\Pr(T_{\max}\leq i)\\
&= \sum_{i=1}^{\infty} \Pr(T_{\text{in}}=i)[1-\Pr(T_{\max}>i)]\\
&= 1-\sum_{i=1}^{\infty} \Pr(T_{\text{in}}=i)\Pr(T_{\max}>i).
\end{aligned}$$

Which also gives the average number of stuffed bits during an output run, since $d(T) = 0$: $\overline{N}_{\text{stuff}} = s$. While the average number of input bits during an output run reads as

$$\overline{N}_{\text{in}} = \sum_{i=1}^{\infty} i\,[\Pr(T_{\text{in}}=i)\Pr(T_{\max}>i) + \Pr(T_{\text{in}}>i)\Pr(T_{\max}=i+1)].$$

Using that

$$\Pr(T_{\text{in}}=i) \;=\; \Pr(T_{\text{in}}\geq i) - \Pr(T_{\text{in}}\geq i+1)$$

and

$$\Pr(T_{\max}=i) \;=\; \Pr(T_{\max}>i-1) - \Pr(T_{\max}>i),$$

we have

$$\begin{aligned}
\overline{N}_{\text{in}} = &\sum_{i=1}^{\infty} i\,[\Pr(T_{\text{in}}\geq i) - \Pr(T_{\text{in}}\geq i+1)]\,\Pr(T_{\max}>i)\\
&+\sum_{i=1}^{\infty} i\,[\Pr(T_{\max}>i) - \Pr(T_{\max}>i+1)]\,\Pr(T_{\text{in}}\geq i+1)\\
= &\sum_{i=1}^{\infty} i\;\Pr(T_{\text{in}}\geq i)\Pr(T_{\max}>i)\\
&-\sum_{i=1}^{\infty} i\;\Pr(T_{\text{in}}\geq i+1)\Pr(T_{\max}>i+1)\\
= &\sum_{i=1}^{\infty} \Pr(T_{\text{in}}\geq i)\Pr(T_{\max}>i).
\end{aligned}$$

Since the input sequence is i.i.d. $\Pr(T_{\text{in}}=i) = q\,\Pr(T_{\text{in}}\geq i)$, so $\overline{N}_{\text{in}}$ can be expressed with the stuffing probability:

$$\overline{N}_{\text{in}} = (1-s)/q. \tag{6}$$

Then the rate reads as

$$R = \frac{\overline{N}_{\text{in}}}{\overline{N}_{\text{out}}} = \frac{\overline{N}_{\text{in}}}{\overline{N}_{\text{in}} + \overline{N}_{\text{stuff}}} = \frac{1-s}{1-ps}\,. \tag{7}$$

When $d(T) \not\equiv 0$ each run is padded with $d(T)$ bits, so the average number of stuffed bits will be

$$\overline{N}_{\text{stuff}} = \mathrm{E}[d(T)] + s. \tag{8}$$

Let us remove these padding bits from the output sequence. Then the maximal runlength will be $T_{\max} = k(T) - d(T) + 1$, which is also independent from T_{in}, so (6) still remains valid. Substituting (8) into (7) we get (5). ■

Applying the above theorem we can give a good estimation for the channel capacity. Let $R(p)$ and $s(p)$ denote the rate and the stuffing probability as a function of the input distribution, and $h(p)$ the entropy function $h(p) = -p\log p - (1-p)\log(1-p)$. With these notations we can constitute a lower bound for the channel capacity:

$$C \geq \max_p \; h(p)R(p) \geq h(\tfrac{1}{2})\,R(\tfrac{1}{2}) = \frac{2\,[1 - s(\frac{1}{2})]}{2 + \mathrm{E}[d(T)] - s(\frac{1}{2})}\,.$$

In general, when $\mathrm{E}[d(T)] \ll 1$ the bit stuff encoder performs a very dense mapping resulting in a high coding efficiency, i.e. the channel capacity can be estimated with $R(\frac{1}{2})$ [13].

4 Some examples for application of the theorem

In this section we present a few examples for calculating the rate of different constrained codes complying with the wide sense RLL channel criteria.

4.1 Conventional (d, k) constrained channel

Coding conventional (d, k) constrained channel the bit stuff encoder makes stuffings when the input run is longer than $k-d$, so the stuffing probability is $s = \Pr(T_{\text{in}} > k-d) = p^{k-d}$, while $d(T)$ is constant: $\mathrm{E}[d(T)] = d$. Then the rate reads as

$$R = \frac{1 - p^{k-d}}{1 + qd - p^{k-d+1}}\,.$$

4.2 Simultaneously RLL and charge constrained channel

In these channels not only the runlength, but the accumulated charge is limited as well. To monitor the accumulated charge, we are using the rectified running digital sum (RRDS):

$$C_n = Y_n \sum_{i=0}^{\infty} Y_{n-i} \leq c \tag{9}$$

It is convenient since the RRDS is always increasing during a run, so it is enough to bind upward its value, and it makes easier to trace the maximal runlength $T_{\max}$ as well. If the RRDS is C at a run's end, then for the next run

$$T_{\max} = \min(C + c,\; k + 1)$$

The bit stuff algorithm coding simultaneously (d, k, c) constrained channel has been studied by Bender and Wolf in [10]. They gave the coder's state transition probability matrix $\mathbf{Q}$:

$$\begin{bmatrix}
0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 1 \\
0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & q & p \\
\vdots & \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots & \vdots \\
0 & 0 & \cdots & 0 & 0 & \cdots & 0 & q & \cdots & p^{k-d-1}q & p^{k-d} \\
0 & 0 & \cdots & 0 & 0 & \cdots & q & pq & \cdots & p^{k-d} & 0 \\
\vdots & \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots & \vdots \\
0 & q & \cdots & p^{k-d-1}q & p^{k-d} & \cdots & 0 & 0 & \cdots & 0 & 0 \\
q & pq & \cdots & p^{k-d} & 0 & \cdots & 0 & 0 & \cdots & 0 & 0
\end{bmatrix}$$

The states correspond to the rectified RDS at the runs' end defined by (9): $C \in \{-c+d+1,\, -c+d+2,\, \ldots,\, c\}$, while each transition corresponds to an output run. Consequently, the components of the stationary distribution vector $\pi = \pi\mathbf{Q}$ give the probabilities of the run's end charge state: $\Pr(C = -c+d+i) = \pi_i$. Stuffing is performed whenever either the accumulated charge reaches the threshold c, or the input run is longer than $k-d$, so the stuffing probability is given as

$$s = \Pr(C = c) + p^{k-d}\left[1 - \sum_{i=1}^{k-d+1} \Pr(C = -c + d + i)\right],$$

while $d(T)$ is constant again. The rate calculated by (5) exactly corresponds with the one given in [10].

4.3 Average runlength constrained (ARC) channel

The ARC channel has been introduced by Heegard et al. in [14]. Beside the conventional (d, k) constraint they set an upper bound for the average of the runlength process T:

$$\frac{1}{n}\sum_{i=1}^{n} T_i \leq a, \qquad (d + 1 < a < k + 1).$$

It is carried out by limiting upward the accumulated deviation of the runlength along any cycle of the channel's state transition graph:

$$\sum (T_i - a) \leq b. \tag{10}$$

To satisfy the above constraint, we should keep trace the accumulated deviation of runlengths, but only in positive direction. For this purpose, let us define the process D as

$$D_0 = 0 \quad \text{and} \quad D_i = \max(0,\; D_{i-1} + T_i - a).$$

If process D is limited as $D_i \leq b$, $(i = 1, 2, \ldots)$, the constraint (10) is satisfied:

$$\sum_{m+1}^{n} (T_i - a) \leq D_n - D_m \leq b.$$

The bound on D limits the runlength as well:

$$T_i \leq a + D_i - D_{i-1} \leq a + b - D_{i-1}\,,$$

so, together with k constraint, for $T_{\max}$ we have:

$$T_{\max} = \min(a + b + D,\ k + 1).$$

The adjacency matrix of the ARC channel is given in [14]. Taking into account that the bit stuff algorithm coding for ARC channel pads each run with d bits, and it should start a new run whenever either the accumulated deviation reaches b or the input run is longer than $k-d$, it is not too difficult to construct the coder's state transition probability matrix from the adjacency matrix:

$$\begin{bmatrix}
\sum_{i=0}^{a-d-1} p^i q & p^{a-d}q & \cdots & p^{k-d} & 0 & \cdots & 0 & 0 & \cdots & \wr & \cdots & 0 & 0 \\
\sum_{i=0}^{a-d-2} p^i q & p^{a-d-1}q & \cdots & p^{k-d-1}q & p^{k-d} & \cdots & 0 & 0 & \cdots & \wr & \cdots & 0 & 0 \\
\vdots & \vdots & & & & & \vdots & \vdots & & \wr & & \vdots & \vdots \\
q & pq & \cdots & \cdots & \cdots & \cdots & p^{k-d} & 0 & \cdots & \wr & \cdots & 0 & 0 \\
0 & q & \cdots & \cdots & \cdots & \cdots & p^{k-d-1}q & p^{k-d} & \cdots & \wr & \cdots & 0 & 0 \\
\vdots & \vdots & & \sim & \sim & \sim & \sim\ \sim & \sim & \sim & \sim & & \vdots & \vdots \\
0 & 0 & \cdots & \wr & \cdots & q & pq & \cdots & \cdots & \cdots & \cdots & p^{k-d} & 0 \\
0 & 0 & \cdots & \wr & \cdots & 0 & p & \cdots & \cdots & \cdots & \cdots & p^{k-d-1}q & p^{k-d} \\
\vdots & \vdots & & \wr & & \vdots & \vdots & & & & & \vdots & \vdots \\
0 & 0 & \cdots & \wr & \cdots & 0 & 0 & \cdots & q & pq & \cdots & p^{a-d-1}q & p^{a-d} \\
0 & 0 & \cdots & \wr & \cdots & 0 & 0 & \cdots & 0 & q & \cdots & p^{a-d-2}q & p^{a-d-1}
\end{bmatrix}$$

The states correspond to the accumulated deviation $D = 0, 1, 2, \ldots, b$; while each transition corresponds to an output run. The stuffing probability can be expressed with the help of the components of the stationary distribution $\pi_i = \Pr(D = i - 1)$:

$$s = \Pr(D = b) + p^{k-d} \sum_{i=0}^{a+b-k-2} \Pr(D = i),$$

while d is constant in the case.

4.4 Simultaneously RLL and α-charge constrained channel

The *α-charge constraint* is a kind of generalized charge constraint when the channel sequence is bound on the output of the IIR low-pass filter $H(z) = 1/(1 - \alpha z^{-1})$:

$$W_n = Y_n \sum_{i=0}^{\infty} \alpha^i Y_{n-i} < c, \qquad (11)$$
$$(0 < \alpha < 1 \text{ and } \tfrac{1+2\alpha}{1+\alpha} < c < \tfrac{1}{1-\alpha}).$$

The amount W we refer to as *rectified weighted running digital sum* (RWRDS). Since the applied filter enhances the low frequency components of the channel sequence, keeping the RWRDS low it will mostly affect those components, resulting in a code spectrum with a suppression at low frequencies [8, 15].

The α-charge constraint limits the runlength in itself. If the RWRDS is W at a run's end, then according to (11), for the next runlength T we can write:

$$(1 + \alpha^T)/(1 - \alpha) - W\alpha^T < c.$$

Expressing T we have

$$T < \frac{1}{\log \alpha}\left[\log(1 - c + \alpha c) - \log(1 + W - \alpha W)\right].$$

That is, taking the explicit k constraint into account:

$$T_{\max} = \min\left\{k + 1,\ \left\lceil \frac{1}{\log \alpha}\ \log \frac{1 - (1 - \alpha)c}{1 + (1 - \alpha)W} - 1 \right\rceil\right\}.$$

The process W has a continuous distribution for most α, so it can be described as a Markov process rather than a Markov chain. The whole constrained system and the corresponding bit stuff encoder structure is studied in chapter 4 of [15]. The applied bit stuff algorithm inserts extra bits whenever either the RWRDS reaches or exceeds the threshold $c_0 = (c - 1)/\alpha$ or the input run is longer than $k-d$.

Let $G(x) = \Pr(W_n < x \mid Y_n \neq Y_{n+1})$, i.e. the distribution of RWRDS at the end of the runs. Then for $G(x)$ we can write the following functional equation:

$$G(x) = \begin{cases}
0, & \text{if } x \le x_{\mathrm{m}}; \\
1 - \sum_{i=d+1}^{k} q p^{i-(d+1)}\, G\Big(-\alpha^{-i}x - \frac{1-\alpha^{-i}}{1-\alpha}\Big) - & \\
-p^{k-d}\, G\Big(-\alpha^{-(k+1)}x - \frac{1-\alpha^{-(k+1)}}{1-\alpha}\Big), & \text{if } x_{\mathrm{m}} < x \le c_0; \\
1 - \sum_{i=d+1}^{k+1} p^{i-(d+1)}\, G\Big(-\alpha^{-i}x - \frac{1-\alpha^{-i}}{1-\alpha}\Big) + & \\
+ \sum_{i=d+1}^{k} p^{i-d}\, G\Big(-\alpha^{-i}c_0 - \frac{1-\alpha^{-i}}{1-\alpha}\Big), & \text{if } c_0 < x \le c; \\
1, & \text{if } x > c;
\end{cases}$$

where $x_{\mathrm{m}} = (1 - \alpha^{d+1})/(1 - \alpha) - \alpha^{d+1}c$, the lowest possible value of the RWRDS at the end of a run. With the help of the distribution function we can give the stuffing probability of the bit stuffing algorithm:

$$s = 1 - G(c_0) + p^{k-d}\left[1 - G\left(-\alpha^{-(k+1)}c_0 - \tfrac{1-\alpha^{-(k+1)}}{1-\alpha}\right)\right].$$

Then the rate can be calculated by (5).

References

1 **Bernoulli J**, *Ars Conjectandi.* Basel, 1713.

2 **Kautz W H**, *Fibonacci codes for synchronization control*, IEEE Trans. Inform. Theory, **IT-11**, (April, 1965), 284–292, DOI 10.1109/TIT.1965.1053772.

3 **Tang D T, Bahl L R**, *Block codes for a class of constrained noiseless channels*, Inform. and Control, **17**, (1970), 436–461, DOI 10.1016/S0019-9958(70)90369-4.

4 **Chien T M**, *Upper bound on the efficiency of dc-constrained codes*, Bell Syst. Tech. J. **49**, (Dec. 1970), 2267–2287.

5 **Patel AM**, *Zero-modulation encoding in magnetic recording*, IBM J. Res. Develop. **19**, (July 1975), 366–378, DOI 10.1147/rd.194.0366.

6 **Marcus B, Siegel P**, *On codes with spectral nulls at rational submultiples of the symbol frequency*, IEEE Trans. Inform. Theory, **IT-33**, (July 1987), 557–568, DOI 10.1109/TIT.1987.1057334.

7 **Waldman H, Pingarilho C**, *Spectral shaping codes*, Proc. IEEE Symp. on Inform. Theory, Trondheim, Norway, June, 1994, 209, DOI 10.1109/ISIT.1994.394759, (to appear in print).

8 **Vámos P, Osváth L, Telek M**, *A new method for spectral shaping coding*, Proc. IEEE Winter 1998 Inform. Theory Workshop, San Diego, Calif. Feb. 1998.

9 **Pierobon G L**, *Codes for zero spectral density at zero frequency*, IEEE Trans. Inform. Theory, **IT-30**, (March, 1984), 435–439, DOI 10.1109/TIT.1984.1056858.

10 **Bender P E, Wolf J K**, *A universal algorithm for generating optimal and nearly optimal run-length-limited charge-constrained binary sequences*, Proc. IEEE Symp. on Inform. Theory, San Antonio, Texas, Jan. 1993, 6, DOI 10.1109/ISIT.1993.748321, (to appear in print).

11 **Aviran s, Sigel P H, Wolf J K**, *An improvement to the bit suffing algorithm*, IEEE Trans. Inform. Theory, **IT-51**, (Aug, 2005), 2885–2891.

12 **Immink K**, *Coding Techniques for Digital Recorders*, London: Prentice Hall, 1991.

13 **Vámos P**, *On the distribution of waiting time for runs of given length*, Proc. IEEE Symp. on Inform. Theory, Trondheim, Norway, June, 1994, 183, DOI 10.1109/ISIT.1994.394789, (to appear in print).

14 **Heegard C D, Marcus B H, Siegel P H**, *Variable-length state splitting with application to average runlength-constrained (ARC) codes*, IEEE Trans. Inform. Theory, **IT-37**, (May, 1991), 759–777, DOI 10.1109/18.79946.

15 **Vámos P**, *Adaptive constrained coding*, Budapest University of Technology and Economics, 2002, available at `http://alpha.tmit.bme.hu/~vamos/Thesis.pdf`. PhD thesis.