

7th IEEE EAST-WEST DESIGN & TEST SYMPOSIUM

September 18-21, 2009 Moscow, Russia

An Algorithm for Testing Run-Length Constrained Channel Sequences

Oleg Kurmaev

Moscow Institute of Electronic Engineering

Outline

1. We define channel coding and channel sequences.
2. Enumerative schemes will be discussed; from the trivial case — the successive approximations method to more sophisticated Cover's enumerative scheme. The problems that may violate coding will be also shown.
3. We intend to introduce a method, which allows us to detect and tracing errors that were caused by enumeration.

Definition 1. Under NRZI encoding we understand mapping the source sequence x to bipolar sequence z , $z \in \{-1, 1\}$ such that

$$z_i = \begin{cases} z_{i-1}, & x_i = 0, \\ -z_{i-1}, & x_i = 1, \end{cases}$$

$z_0 = 1.$

Example 1. The NRZI rule.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
0	1	0	0	0	1	0	0
1	-1	-1	-1	-1	1	1	1
1	0	0	1	0	0	0	1
-1	-1	-1	1	1	1	1	-1

Example 2. Run-Length Constrained Binary Sequences:

1. **RLL (1,7)**

0 1 0 0 1 0 1 0 0 0 0 0 0 0 1 0

encoded,



1
-1 waveform (NRZI);

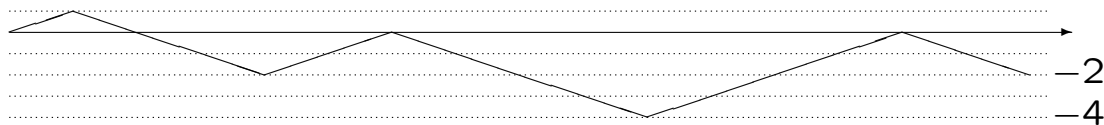
2. **RLL (1,7)**, charge: $\sigma = -2$, DSV: $\vartheta_1 = -4$, $\vartheta_2 = 1$.

0 1 0 0 1 0 1 0 0 0 1 0 0 0 1 0

encoded,



1
-1 waveform (NRZI),



charge, DSV;

Definition 2. A $dklr$ -limited binary sequence satisfies simultaneously the following conditions:

1. d -constraint – two ones are separated by a run of consecutive zeros of length at least d ;
2. k -constraint – any internal run of consecutive zeros is of length at most k ;
3. l -constraint – leading run of consecutive zeros is of length at most l ;
4. r -constraint – trailing run of consecutive zeros is of length at most r .

Example 3. $\{0, 1\}^n$, Fibonacci code (RLL (1, ∞)), and RLL (1, 3).

	2^n weight set				Fibonacci weights					RLL (1, 3) weights				
w	8	4	2	1	8	5	3	2	1	7	5	3	2	1
N	x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4	x_5	x_1	x_2	x_3	x_4	x_5
0	0	0	0	0	0	0	0	0	0					
1	0	0	0	1	0	0	0	0	1					
2	0	0	1	0	0	0	0	1	0	0	0	0	1	0
3	0	0	1	1	0	0	1	0	0	0	0	1	0	0
4	0	1	0	0	0	0	1	0	1	0	0	1	0	1
5	0	1	0	1	0	1	0	0	0	0	1	0	0	0
6	0	1	1	0	0	1	0	0	1	0	1	0	0	1
7	0	1	1	1	0	1	0	1	0	0	1	0	1	0
8	1	0	0	0	1	0	0	0	0	1	0	0	0	1
...				

Decoding and encoding algorithms.

```
 $N := 0;$      $a := 1;$   
for  $j := 1$  to  $n$  do  
  Get  $w(j, a);$   
  if  $x_j = 1$  then  
     $N := N + w(j, a);$   
     $a := 1;$   
  else  
     $a := a + 1;$   
  end if ... else  
end for.
```

```
 $a := 1;$   
for  $j := 1$  to  $n$  do  
  Get  $w(j, a);$   
  if  $N \geq w(j, a)$  then  
     $N := N - w(j, a);$   
     $x_j := 1;$      $a := 1;$   
  else  
     $x_j := 0;$      $a := a + 1;$   
  end if ... else  
end for.
```

Run-length counter a and corresponding operations are emphasized here.

An ambiguity of conventional methods.

1. 2^n weight set — guarantee a unique representation;
2. Fibonacci weights — an example of dual representation;

w	8	5	3	2	1
N	x_1	x_2	x_3	x_4	x_5
5	0	0	1	1	0
5	0	1	0	0	0

3. dk or dkr - weight set — conventional algorithms failure.

w	10	7	5	3	2	1
N	x_1	x_2	x_3	x_4	x_5	x_6
17-2	1	1	0	0	0	0
17-2	1	0	1	0	1	0

2 — is the threshold.

Cover's enumerative scheme

Let $\{0, 1\}^n$ denote the set of binary sequences of length n and let $\mathbf{x} = (x_1, x_2, \dots, x_n)$ denote a generic element of this set. Let $\mathcal{S} = \{\mathbf{x} \in \{0, 1\}^n : |\mathcal{S}| \leq 2^n\}$. Let \mathcal{S} be ordered lexicographically. The 1–1 lexicographic mapping $\mathcal{S} \rightarrow \{0, 1, 2, \dots, |\mathcal{S}| - 1\}$ can be obtained by

$$N(\mathbf{x}) = \sum_{i=1}^n x_i W(x_1, x_2, \dots, x_{i-1}, 0),$$

where $N(\mathbf{x})$ is the lexicographic index of $\mathbf{x} \in \mathcal{S}$ and $W(x_1, x_2, \dots, x_u)$ denote the number of elements in \mathcal{S} for which the first u coordinates are (x_1, x_2, \dots, x_u) .

The number of dkr - sequences,
which start with one

$$\hat{N}(n) = \sum_{i=d+1}^{k+1} \hat{N}(n-i), \quad n > d+k.$$

The number of $dklr$ - sequences

$$|\mathcal{S}| = \sum_{i=0}^{\min(n,l)} \hat{N}(n-i).$$

Example 4. Let $d = 2, k = 4, l = 1, r = 3$.

I_S	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	\hat{I}_S
0	0	1	0	0	0	0	1	0	
1	0	1	0	0	0	1	0	0	
2	0	1	0	0	1	0	0	0	
3	0	1	0	0	1	0	0	1	
4	1	0	0	0	0	1	0	0	0
5	1	0	0	0	1	0	0	0	1
6	1	0	0	0	1	0	0	1	2
7	1	0	0	1	0	0	0	1	3
8	1	0	0	1	0	0	1	0	4

Weight and charge constraints.

By $\nu = \sum_{i=1}^n x_i$ denote the weight of the sequence x .

Recall the NRZI rule; then

by $\sigma = \sum_{i=1}^n z_i$ denote the digital sum or charge of the sequence z .

Example 5. $d = 2, k = 4, l = 1, r = 3.$

N	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	ν	σ
0	0 1	1 -1	0 -1	0 -1	0 -1	0 -1	1 1	0 1	2	-2
1	0 1	1 -1	0 -1	0 -1	0 -1	1 1	0 1	0 1	2	0
2	0 1	1 -1	0 -1	0 -1	1 1	0 1	0 1	0 1	2	2
3	0 1	1 -1	0 -1	0 -1	1 1	0 1	0 1	1 -1	3	0
4	1 -1	0 -1	0 -1	0 -1	0 -1	1 1	0 1	0 1	2	-2
5	1 -1	0 -1	0 -1	0 -1	1 1	0 1	0 1	0 1	2	0
6	1 -1	0 -1	0 -1	0 -1	1 1	0 1	0 1	1 -1	3	-2
7	1 -1	0 -1	0 -1	1 1	0 1	0 1	0 1	1 -1	3	0
8	1 -1	0 -1	0 -1	1 1	0 1	0 1	1 -1	0 -1	3	-2

Example 6. Recurrence relations for dkr , charge constrained dkr , and DSV constrained dkr sequences:

$$1. \hat{N}_n = \sum_{j=d+1}^{k+1} \hat{N}_{n-j}, \quad n > d + k;$$

$$2. C_n^\sigma = \sum_{j=d+1}^{\min(n, k+1)} C_{n-j}^{-\sigma-j}, \quad d + 1 \leq n;$$

$$3. C_n^\sigma(\vartheta_1, \vartheta_2) = \sum_{j=d+1}^{\min(n, k+1, -\vartheta_1)} C_{n-j}^{-\sigma-j}(-\vartheta_2 - j, -\vartheta_1 - j),$$

$$d + 1 \leq n.$$

We are presented some examples of different recurrence relations with different initial conditions. All these equations are known in literature due to contributions coming from K. Immink, V. Braun, V. Levenshtein, V. Kolesnik, K. Abdel-Ghaffar, and some other authors.

But we can require more. . . Frequently it might cause errors.

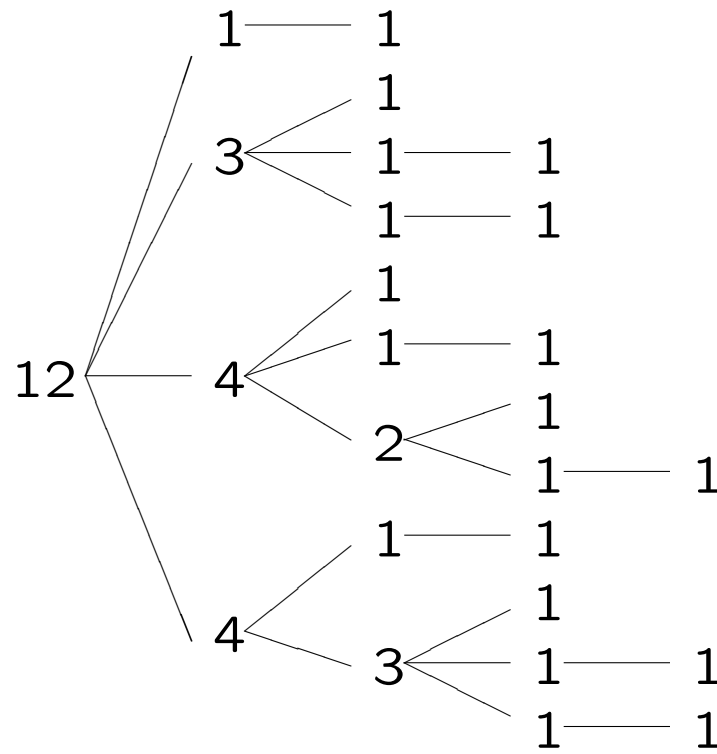
So, what can we do to avoid these things?

Example 7. An example of the tree. $d = 1$, $k = 4$, $r = 3$, charge: $\sigma = 0$ (**dc** - free), abs value of RDS does not exceed 3.

```

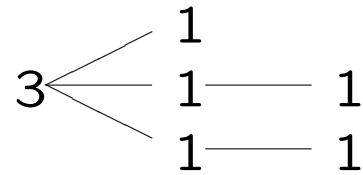
0  0 0 0 1 0 0 0 1
1  0 0 1 0 0 0 1 0
2  0 0 1 0 0 1 0 1
3  0 0 1 0 1 0 1 0
4  0 1 0 0 0 1 0 0
5  0 1 0 0 1 0 0 1
6  0 1 0 1 0 0 1 0
7  0 1 0 1 0 1 0 1
8  1 0 0 1 0 0 0 1
9  1 0 1 0 0 0 1 0
10 1 0 1 0 0 1 0 1
11 1 0 1 0 1 0 1 0

```



Example 8. The number of edges.

1	0	0	1	0	0	0	1	0
2	0	0	1	0	0	1	0	1
3	0	0	1	0	1	0	1	0



The number of edges, which are descendant of the vertex, equals difference of lengths of greatest and smallest runs of zeros; these runs are assigned to this vertex and begin with $u + 1$, therefore

$$N(x_1, \dots, x_{u-1}, 1) = \sum_{j=d(p)}^{k(p)} N(x_1, \dots, x_{u-1}, 10^j, 1).$$

$$\begin{aligned}
& N(x_1, \dots, x_{u-1}, 1) \\
&= \begin{cases} \sum_{j=d(\mathbf{p})+1}^{k(\mathbf{p})+1} N(x_1, \dots, x_{(u-1)+j}, 1), & 0 < u \leq n - k(\mathbf{p}), \\ \sum_{j=d(\mathbf{p})+1}^{r(\mathbf{p})+1} N(x_1, \dots, x_{(u-1)+j}, 1) + 1, & n - k(\mathbf{p}) < u \leq n - r(\mathbf{p}). \end{cases}
\end{aligned}$$

Here $d(\mathbf{p})$ is the length of smallest run of zeros, $k(\mathbf{p})$ is the length of greatest run of zeros, and $r(\mathbf{p})$ is the length of trailing run of zeros within the vertex P .

The second equation accounts ($+ 1$) the existence of trailing run of zeros.

The index of the first of lexicographically ordered sequences (for the sequences, which are assigned to the superior vertex) can be computed as

$$N(x_1, \dots, x_{u-1}, 0) = \sum_{j=1}^{(k(\mathbf{p})+1)-r(\mathbf{p}_{-1})} N(x_1, \dots, x_{u-1}, 0^j, 1),$$

where $r(\mathbf{p}_{-1})$ is the length of trailing run of zeros within the prefix $(x_1, \dots, x_{u-1}, 0)$, i.e., within the superior vertex P_{-1} .

Example 9. Mapping the set \mathcal{S} on the set of t .

N	x_j	t_j
0	0 0 0 1 0 0 0 1	0 0 0 1 0 0 0 1
1	0 0 1 0 0 0 1 0	0 0 1 0 0 0 1 0
2	0 0 1 0 0 1 0 1	0 0 2 0 0 1 0 1
3	0 0 1 0 1 0 1 0	0 0 3 0 1 0 1 0
4	0 1 0 0 0 1 0 0	0 1 0 0 0 1 0 0
5	0 1 0 0 1 0 0 1	0 2 0 0 1 0 0 1
6	0 1 0 1 0 0 1 0	0 3 0 1 0 0 1 0
7	0 1 0 1 0 1 0 1	0 4 0 2 0 1 0 1
8	1 0 0 1 0 0 0 1	1 0 0 1 0 0 0 1
9	1 0 1 0 0 0 1 0	2 0 1 0 0 0 1 0
10	1 0 1 0 0 1 0 1	3 0 2 0 0 1 0 1
11	1 0 1 0 1 0 1 0	4 0 3 0 1 0 1 0

Example 10.

Consider two different sets $\mathcal{S}_A, \mathcal{S}_B$ and the sequence $x(8) = (10010001)$ such that $x(8) \in \mathcal{S}_A$ and $x(8) \in \mathcal{S}_B$. If we assign t to x , then we obtain $t(8) = (10010001) \mapsto x(8) \in \mathcal{S}_A$ and $t(8) = (40020001) \mapsto x(8) \in \mathcal{S}_B$. Thus, the error has defined without checking eight previous sequences.

Finally, we present the algorithm using pseudo-code.

```
x :   array [1..n];
t :   array [0..1, 1..n];
Set t1,n := 0;
for N := 0 to | $\mathcal{S}$ | - 1 do
    Get x(N);
    s := 1;
    for j := 1 to n do
        tN mod 2,j := (st(N+1) mod 2,j + 1)xj;
        if tN mod 2,j = 1 then s := 0;
    end for
end for
```