整数リング上において置換多項式を使用するターボ符号のためのインタリーバ

1 置換多項式の探索

置換多項式に基づいてインタリーバの場合は、良い係数をうまく選べば $\Delta(x,t)=0$ の線にかなり近い点がなくなる。置換多項式に基づいてインタリーバはそれぞれの性能が違う。フレームサイズと要素符号が与えられたら、最適置換多項式を見つけようとするのである。固定フレームサイズが与えたら、残りの変数は多項式の次数と係数である。この論文では

$$P(x)=bx^2+ax+c$$

のようの二次多項式に着目する。一つ目の理由は可能な限り低い複雑さを持ちたいためである。置換多項式の中で最も簡単な種類は

$$P(x)=ax+c$$

(線形インタリーバ) の形を持つ一次多項式である。しかし、論文 [6] で示されるように、線形インタリーバは悪い入力重み 4 エラーイベント性質を持つため、中間から長いフレームの場合では高いエラーフローを起こす。そのため、二次多項式が注目される。二番目の理由は二次多項式の分析が相対的に簡単だからである。定数項 c はインタリーブされたシーケンスの周期的回転にちょうど対応する。境界効果を無視したら、連結されたシステムの性能に影響を与えず、置換多項式の条件と関係ないので、0 とし、

$$P(x)=bx^2+ax$$

のような多項式を考える。

多項式の良い係数を選ぶために、ターボ符号のエラーイベントの部分集合の最低距離を基準とする。その部分集合は入力重み 2m のエラーイベントである。そのエラーイベントは簡単に見つけて数えることができるからである。もちろん、そのエラーイベントはすべてのあり得るエラーイベントの場合を示さないが、置換多項式に基づいたインタリーバの構造のため、特に短いフレーメサイズのとき、そのエラーイベントの多重度は高く、通常の場合、TC の性能に影響を与える。この論文ではそれぞれの RSC 符号の要素符号が (tail-biting trellis)[16] を使っている仮定を使う。trellis の末尾は trellis の先頭と直接に繋いでいて、(flushing bits) を使わない。(tail-biting trellis) では要素

符号におけるエラーイベントの循環けた送りもエラーイベントである。そして、trellis の末尾の近くから始まるエラーイベントが trellis の先頭にかかることが可能です。そのため、エラーイベントが mod N といえる。その仮定を使うと、終端にする境界効果を無視することができる。残念ながら、RSC 符号を要素符号とするので、常に (tail-biting trellis) が存在するわけではない [17]。終端を使わなければならない。あるエラーイベントは終端でなくなる。そして、終端でエラーイベントを起こすときもある。ですから、エラーイベントを mod N で探せば、エラーを導かける。しかし、終端で壊された mod N エラーイベントの割合が少ないし、終端にすって生じるの重みエラーイベントの多重度はたいてい低いので、短いフレームサイズのとき、mod N エラーイベントは性能を左右する。うえに、エラーイベントを mod N で数えてもかまわない。

1.1　入力重み **2m** エラーイベント

長いランダムインタリーバは均一なインタリーバで近似できる。均一なインタリーバというのは、与えられた入力を同じ確率で出力ポジションに並び替える確率的なデバイスのことである。[14]。均一なインタリーバモデルを使って最高 SNR 領域での復号性能は入力重み２エラーイベントに左右される。入力重み２エラーイベントに対して最小距離は (minimum effective distance)d_{ef} と呼び [2]、要素符号が良いエラーフロー性能をえるために、設計条件として、使われている。置換多項式は線形インタリーバの一般化のように考えることができる。線形インタリーバと同じように、置換多項式によく起きるエラーイベントは入力重み　エラーイベントである $(m = 1, 2, ...)$ しかし、多項式の係数の a と b をうまく選べばそのようなエラーイベントを制御することができる。要素符号が与えたら、良い性能をもつ二つの係数が見つけられる。代表的な入力重み m エラーイベントは図３に示される。それぞれの入力重みエラーイベントは最初と最後のポジションを示す整数の組で表される。二つの要素符号は同じ RSC 符号を使用するので、すべての t_i と s_i は畳み込み符号の (cycle length)τ の倍数である。この論文では、(cycle length)τ というのは入力シーケンスが [1,0,0,0......] のとき、符号器の出力の周期である。

例　要素符号 = $\frac{1+D^2}{1+D+D^2}$,8 進数で 5/7
入力 =[1,0,0,0,...] 出力 =[1,1,1,0,1,1,0,1,1,0]
周期 =[1,1,0], (cycle length)$\tau = 3$
(cycle length) ＝最低入力重み２エラーイベントの距離- １。エラーパタンを以下の長さ２mのベクトルのように定義する。

$$[t_1, t_2, ..., t_m, s_1, s_2, .., s_m]$$

入力重み２エラーイベントで、以下のm式が書ける。

$$P(x_2) - P(x_1) = s_1 \tag{3.1}$$

$$P(x_3) - P(x_1 + t_1) = s_2 \tag{3.2}$$

$$P(x_4) - P(x_2 + t_2) = s_3 \tag{3.3}$$

$$P(x_m + t_m) - P(x_{m-1} + t_{m-1}) = s_m \tag{3.m}$$

t_i,s_i の値は τ の小さい倍数,$x_i = \ 0, 1, ...N-1$
エラーイベントの見つける方法を簡単にするために、式 3 を分析しやすい形に変換する。すると以下のようになります。

$$\begin{aligned} s_1 - s_2 + s_3 - s_4... = 2b(&x_1t_1 - x_2t_2 + x_3t_3 - x_4t_4...) \\ &+ b((t_1)^2 - (t_2)^2 + (t_3)^2 - (t_4)^2...) \\ &+ a(t_1 - t_2 + t_3 - t_4...) \end{aligned} \tag{4}$$

または、もっと簡単な形で

$$\begin{aligned} &\sum_{i=1}^{m}(-1)^{i-1}s_i = 2b\sum_{i=1}^{m}(-1)^{i-1}x_it_i \\ &\qquad\qquad + b\sum_{i=1}^{m}(-1)^{i-1}{t_i}^2 + a\sum_{i=1}^{m}(-1)^{i-1}t_i \end{aligned} \tag{5}$$

図 3 のような入力重み $2m$ エラーイベントが現れるために、式 5 は式 3 の残りの m-1 式と一緒に使わなければなりません。この論文には、入力重み 2 mエラーイベントを見つけるために式 3-5 を使われる。あるエラーパターンが与えられ、最初のエラーイベントが重ならなかったら、エラーイベントのハミング距離を一意に決定できる。例で説明する。
要素符号は 5/7 の RSC 符号とする。その符号の $\tau = 3$。t_i と s_i は τ の倍数なので $k\tau$ の一般形を持つ。入力は、$1 + D^{k\tau}$ とする。出力は以下のようになる。

$$\begin{aligned} &(1 + D^{3k})\frac{1 + D^2}{1 + D + D^2} \\ &\qquad = (1 + D^3 + D^(2.3) + ... + D^{3(k-1)}) \\ &\qquad \times (1 + D^3)\frac{1 + D^2}{1 + D + D^2} \\ &\qquad = (1 + D^3 + D^(2.3) + ... + D^{3(k-1)}) \\ &\qquad \times (1 + D + D^2 + D^3) \end{aligned} \tag{6}$$

入力の $1 + D^3$ にたいして出力シーケンスの重みは $w_0 = 2$ になる。(最初と最後の 1 を入れずに) そのうえ、入力の $1 + D^3k$ にたいして出力シーケンスの重みは $2 + w_0k$ になる。エラーエベントの全出力重みは以下のようになる。

$$6m + \left(\frac{\sum |t_i|}{\tau} + \frac{\sum |s_i|}{\tau}\right)w_0 \tag{7}$$

この論文では式 (7) を使ってエラーパターンのハミング距離を計算する。

1.2 効果的な自由距離 (d_{ef}) を使用して、良いインタリーバを探索する。

決定論インタリーバでは大きな d_{ef} が良い性能を保証するわけではないが、小さい d_{ef} だと通常、悪い性能になる関係がある。このように悪い置換多項式を選ばないように、d_{ef} を基準とする。置換多項式に基づいてインタリーバを使う場合、多項式の係数をうまく選べば、ある要素符号によく起きる重み2エラーイベントが防止できる。そうすると、それより大きい入力重み２エラーイベントをなくすことができる。1 番目の要素符号に起きる入力重み２エラーイベントの長さを $t+1$ とする。そうすると t は τ の倍数で、t のオーダーは o_t とする。２番目の要素符号に起きる入力重み２エラーイベントの長さ引く 1 は以下のようになる。

$$\Delta(x,t)\ P(x+t) - P(x) = 2btx + bt^2 + at = c_1x + bt^2 + at \tag{8}$$

性質 2.9 でより x の係数は $c_1 = 2bt$ であり, オーダーは $o_{c1} = o_2 + o_b + o_t$ である。$x \in \{0,1,2,...,N-1\}$ のとき、式 (8) での第一項は $k \cdot p_N^{o_{c1}}, k = \{0,1,2,...,p_N^{(o_N - o_{c1})}\}$
それぞれの値は $p_N^{o_{c1}}$ 回をとる。x に従って c_1x の図を描くと $p_N^{(o_N - o_{c1})}$ の水平線が出る。$bt^2 + at$ は水平線のオフセットを与える。短い入力重み２エラーイベントを防止するために、 t が τ の小さい倍数の場合、τ の倍数である $\Delta(x,t)$ もを0から離れてほしい。このためには、ベクトル o_{c1} を大きくしたい。o_{c1} はもう大きいため、0 から最初の線着目する。0 からの距離は以下のように書ける。

$$s = \pm\Delta(x,t) mod p_N^{o_{c1}} = (bt^2 + at) mod p_N^{o_{c1}} \tag{9}$$

a,b,τ が与えられたとき、$\mathbf{L}_{(a,b,\tau)}$ は以下のように定義して、良いインタリーバを選ぶ基準とする。

$$\mathbf{L}_{(a,b,\tau)}\ min\ (|s| + |t|)$$

要素符号が与えられたとき、$\mathbf{L}_(a,b,\tau)$ から d_{ef} を計算することができる。良い a と b を探索するとき、範囲を制限したらよい。以下の補題で a と b の範囲を制限することができる。

補題 4.1
入力重み２エラーイベントの解析では、b を $b_1 \cdot b_0$=$b_1 \cdot p_N^{o_{b1}}$ のようにかけば b1 を 1 とすることができる。

Proof. $b_1 = 1$ と仮定すると、b_1 と N は互いに素である。ある置換多項式 $P_1(x) = p_N^{o_b}x^2 + at)$ が与えたら、(9) は

$$s_1 = p_N^{o_b}t^2 + at\ mod\ p_N^{o_b+o_t+o_2}$$

もう一つの置換多項式 $P_2(x) = b_1p_N^{o_b}x^2 + at)$ が与えたら、(9) は

$$s_2 = b_1p_N^{o_b}t^2 + at\ mod\ p_N^{o_b+o_t+o_2}$$

$s_2 - s_1$ を計算すると以下の式が出る。

$$s_2 - s_1 = (b_1 - 1)p_N o_b t^2 + at\ mod\ p_N^{o_b+o_t+o_2} \tag{10}$$

もし 2 は N の因数ならば、b_1 と N は互いに素であるので b_1 は奇数で、$b_1 - 1$ は偶数である。式 (10) の右辺のオーダーは少なくとも $o_2 + o_b + 2o_t$。2 は N の因数でないとき、式 (10) の右辺のオーダーは少なくとも $o_b + 2o_t$ であり、mod $o_b + o_t$ で計算する。両方の場合に

$$s_2 - s_1 = 0\ mod\ p_N^{o_b+o_t+o_2}$$

$P_1(x)$ と $P_2(x)$ の入力重み 2 エラーイベントの位置以外は同じ入力重み 2 エラーイベントを持っている。この観点から、$P_1(x)$ と $P_2(x)$ は均しいである。 □

補題 4.2
入力重み 2 エラーイベントの解析では、$b = b_1 \cdot p_N^{o_{b1}}$ があたえられたとき、a は $1 \leq a \leq p_N^{o_{b1}}$ となる a だけ考えば十分である。

Proof. 補題 4.1 の結果より $b = p_N^{o_b}$。$a_0 = a\ mod\ p_N^{o_b+o_2}$ とする。すると、$a = a_0 + lp_N^{o_b+o_2}$。

$$\begin{aligned} s &= \pm(bt^2 + (a_0 + lp_N^{o_b+o_2})t)\ mod\ p_N^{o_b+o_t+o_2} \\ &= \pm bt^2 + (a_0^{o_b+o_2})t\ mod\ p_N^{o_b+o_t+o_2} \end{aligned} \tag{11}$$

これは $\mathbf{L}_(a, b, \tau) = \mathbf{L}_(a_0, b, \tau)$ を意味する。2 は N の因数でないとき、上記の証明は十分である。もし 2 は N の因数ならば、一般性を失わずに、上の証明で $1 \leq a < p_N^{o_b+o_2}$ を仮定できる。$a_0 = p_N^{o_b+o_2} - a$ とすると、

$$\begin{aligned} s &= \pm(bt^2 + (a_0 t)\ mod\ p_N^{o_b+o_2+o_t} \\ s &= \pm(bt^2 + (p_N^{o_b+o_t+o_2} - a)t)\ mod\ p_N^{o_b+o_2+o_t} \\ &= \pm(b(-t)^2 + (a_0(-t))\ mod\ p_N^{o_b+o_2+o_t} \end{aligned} \tag{12}$$

また、$\mathbf{L}_(a, b, \tau) = \mathbf{L}_(a_0, b, \tau)$ □

7/5 と 5/7 要素符号の場合の結果をテーブル 1 に示す。

$$\tau(7/5) = 2, \tau(5/7) = 3, N = 2^n, p_N = [2], o_N = [n], o_b = [4]\, b = 16$$

o_b が与えられたら、d_{ef} で良い a と b を選ぶのは十分可能であるように思える。しかし、d_{ef} のみでは o_b を選ぶ十分な情報でない。例えば式 (9) を見ると、o_b が大きければもっと良い a を選ぶことができる。しかし、シミュレーションより、置換多項式の性能は o_b に従ってある値まで良くなって、その値を超えると性能が悪くなる。それを説明して、これより正確パラメータを選ぶ方法を見つけるために、より高い入力重みエラーイベントを調べなければならない。

1.3　より高い入力重みエラーイベント

良い置換多項式を探索すとき、入力重み２エラーイベントの d_{ef} で決める。d_{ef} を見つけるために、m が小さい値しか注目しない。大きい値はほとんど大きいハミング距離と関係があるからである。エラーイベントを見つける一つの方法はエラーパタン $[t_1, ..., t_m, s_1, ..., s_m]$ をきめて x_1 を計算する。
エラーパターンと x_1 が決めたら、残りの x_i は、式（３）の残りの $m-1$ 式で計算できる。最後に x_i, t_i, s_i の $3m$ 値をまだ使っていない（３）か（４）式に使って、エラーイベントが正しいかどうかを確かめる。

置換多項式に基づいてインタリーバは高度に構造化ので x_1 を 0 から $p_N^{(o_N-o_2-o_b)}-1$ から確認したら十分である。入力重み２ｍエラーイベントは周期的な構造を持つ。一番目の要素符号に起こるエラーイベントはすべての２ｍ末尾点を $p_N^{(o_N-o_2-o_b)}$ の倍数で循環的に動かしたら、ぜんたいの TC で、正しい入力重み２ｍエラーイベントをまた得る。ゆえに、エラーイベントの探索で x_1 を 0 から $p_N^{(o_N-o_2-o_b)}-1$ から確認したら十分である。$p_N^{(o_N-o_2-o_b)}$ が小さい場合、この方法は有能である。エラーパターンが与えられ、x_1 を見つけるとき、入力重み２ｍエラーイベントの制約を一つの式に変えたい。つまり、式（３）での $m-1$ 残りの式を式（５）で $i = 2, ...m$ をキャンセルする。以下の問題を解決できれば、(3) と (5) を一つの式にすることができる。
問題：N,a,b,s が与え、$P(y) - P(x) = s$ なら、x を y の関数とします。
上記の問題を解決する前、ほかの定義が必要。置換多項式 $P(x) = bx^2 + ax$ と定数した s が与えたら、シーケンス $\{y_i\}$ は以下のように定義します。

$$\begin{aligned} P(y_0) - P(0) &= s \\ P(y_1) - P(1) &= s \\ P(y_2) - P(2) &= s \\ P(y_3) - P(3) &= s \end{aligned} \tag{16}$$

そして、$\Delta_k(i)$ を再帰的に以下のように定義する。

$$\begin{aligned} \Delta_1(i) &= y_i - 1 \\ \Delta_2(i) &= \Delta_0(i+1) - \Delta_0(i) etc \end{aligned}$$

注意：y_i と Δ_k (i) は a,b と s に関する関数である。
すると以下の定理が出ます。

定理 4.3
N,$P(x)$ と s が与えられたとき､すべての i で $\Delta(i)$ は同じオーダを持つ｡$\Delta(i)$ のオーダーは o_{Δ_k} と示したら､k>0 場合 $o_{\Delta_0} = o_s$｡そして､$o_{\Delta_k} = o_{\Delta_{k-1}} + o_b + o_{2k}$｡すべての k では $o_{\Delta_k} = ko_b + ko_2 + \sum_{n=1}^{k} o_n + o_s$ である。Δ_k のオーダーは k に従って厳密に増加ので、最終的に o_N よりおおきくなる。もし K は最大数で $ok \not\geq o_N$ であれば、$\Delta_{K+1}(i) = 0 \ mod \ N$。$\Delta_{K+1}(i)$ のていぎより $\Delta_K(i)$ は

すべての i で定数である。この結果は以下の系で要約された。系 4.4
N,P(x) と s が与え、K が一番大きい数で、$o_{\Delta_k} \not\geq o_N$ の場合、すべての i で $\Delta_K(i) = \Delta_K$ は定数である。注意：K, N, a, b と s の関数である。

系 4.4 での K を見つけられたら、すべての k>K 場合 $\Delta_K(i) = 0$。その上、もしすべての $0 \leq k \leq$ K で,$\Delta_K(0)$ が知られていたら、すべての i で $\Delta_K(i)$ の定義で $\Delta_K(i)$ を計算すくことがでる。わかりやすくするために、Δ_k を $\Delta_k(0)$ と代表する。s を指す必要があったら、$\Delta_k(s)$ を使う。
これで、$P(y) - P(x) = s$ の場合、x と y の関係を見つける道具を持っている。以下の定理で要約された。

定理 4.5 N,P(x) と s が与え、$P(y) - P(x) = s$ であるならば

$$y = x + \Delta_0(s) + \Delta_1(s) + \frac{(x(x-1)}{2!}\Delta_2(s) + \frac{(x(x-1)(x-2)}{3!}\Delta_3(s) + \tag{17}$$

x<k の場合 $\binom{x}{k} = 0$ と定義すると、(17) が以下のように書ける。

$$y = F(x, s) \triangleq x + \sum_{k=0}^{\infty} \binom{x}{k} \Delta_k(s) \tag{18}$$

F(x,s) を整数値の多項式をするために、N が与えたら、$D(k) = \prod_{i=2}^{k} \frac{i}{p_N^{o_i}}$ を定義する。$\frac{\Delta_k(s)}{k!}$ はいつも整数であるわけではないからである。

1.4 式を解くことで、エラーイベントを探索する。

式 (4) が正解であったら、エラーイベントを作ることがでる。定理 4.5 を式 (3) の残り $m - 1$ で使えば、以下の式が出る。

$$\begin{aligned} x_2 &= F(x_1, s_1) \\ x_3 &= F(x_1 + t_1, s_2) \\ x_4 &= F(x_2 + t_2, s_3) \\ x_m &= F(x_{m-2} + t_{m-2}, s_{m-1}) \end{aligned} \tag{20}$$

そして、(20) を (4) で使えば、一般的な x_1 に対する多項式になる。　一般の多項式では、解答、またはいくつの解答があるかを探すことの複雑さは高いである。しかし,m の値が小さい場合もっと簡単な方法がある。これから、$m = 1, 2, 3$ の場合を解く。

1.4.1　$m = 1$、入力重み 2 エラーイベント

上記の場合は、式 (20) を使わずに、式（4）は以下のようになる。

$$2bt_1x_1 + bt_1{}^2 + at_1 - s_1 = 0 \tag{21}$$

x_1 に対する線形多項式としたら、以下のようになります。

$$c_1x + c_0 \tag{22}$$

もし $o_{c0} \geq o_{c1}$ 場合のみ (22) の解答がある。その条件が満たされる場合、$p_N^{o_{c1}}$ で (22) を分割でき、結果は特別な解答を持つ一次多項式 mod $p_N^{o_N - o_{c1}}$ になる。多くの場合ではエラーイベントの位置よりエラーイベントのハミング距離とそのハミング距離の多重度に興味がある。ゆえに、式 (22) を解答するかわりに $o_{c0} \geq o_{c1}$ を確認する。満たされていたら、$p_N^{o_{c1}}$ の多重度を対応するスペクトラム線と足す。

1.4.2　$m = 2$、入力重み **4** エラーイベント

(20) 使って、(4) は以下のようになる。

$$2b[\,x_1t_1 - (x_1 \sum_{k=0}^{\infty} \binom{x_1}{k} \Delta_k(s_1))t_2]\, b(t_1^2 - t_1^2) + a(t_1 - t_2) - (s_1 - s_2) = 0 \tag{23}$$

x_1 に関する条項を収集すると以下のようになる。

$$\begin{aligned}(-2b\Delta_0(s_1)t_2 + b(t_1^2 - t_2^2) + a(t_1 - t_2) - (s_1 - s_2))&\\ +\ 2b(t_1 - t_2 - \Delta_1(s_1)t_2)x_1&\\ -\,2bt_2 \sum_{k=2}^{\infty} \frac{\Delta_k(s_1)}{k!} \prod_{m=0}^{k-1}(x_1 - m) = 0&\end{aligned} \tag{24}$$

式 (24) は分数係数を持つような x_1 に対する多項式です。s_1 が与えられ、系 4.4 での K を見つけることができる。(24) を D(K) と掛けたら、以下のようになります。

$$c_0 + c_1x_1 + c_2(x_1) = 0 \tag{25}$$

$$\begin{aligned}c_0 &= (-2b\Delta_0(s_1)t_2 + b(t_1^2 - t_2^2) + a(t_1 - t_2) - (s_1 - s_2))D(K)\\ c_1 &= 2b(t_1 - t_2 - \Delta_1(s_1)t_2)D(K)\\ c_2(x_1) &= -2bt_2 \sum_{k=2}^{\infty} \frac{\Delta_k(s_1)}{k!} \prod_{m=0}^{k-1}(x_1 - m) = 0.\end{aligned} \tag{26}$$

c_0 と c_1 は整数であり、$c_2(x_1)$ は x_1 に対する多項式である。式 (26) のオーダーは少なくとも、$3o_b + 3o_2 + o_s1 + o_{t2}$。系 2.5 で、$\frac{c_1x_1+c_2(x_1)}{p_N^{o_{c1}}}$ は置換多項式である。$t_1 \neq t_2$ 場合、系 2.5 を使うために、以下の条件がある。

$$o_{c1} \ll 3o_b + 3o_2 + o_{s1} + o_{t2} \tag{27}$$

エラーパターン $[t_1, t_2, s_1, s_2]$ が与えられたとき、式 (27) の条件があっているかどうかを確認する。そして、入力重み 2 エラーイベントと同じように $o_{c0} \geq o_{c1}$ の確認しかしない。正しければ、$p_N^{o_{c1}}$ 同じエラーパタンを持っているエラーイベントがある。もしスペクトラムに着目されたら、$\Delta_0(s_1)$ と $\Delta_1(s_1)$ を解くことになる。

1.4.3 $m = 3$、入力重み 6 エラーイベント

この場合はの (25) は入力重み 4 エラーイベントと同じ形になる。

$$c_1 = 2b(t_1 - t_2 + t_3 - \Delta_1(s_1)t_2 + \Delta_1(s_2)t_3)D$$

$D = D(max(K(s_1), K(s_2)))$ そして、$c_2(x_1)$ の係数のオーダーは少なくとも $min(3o_b + 3o_2 + o_{s_1} + o_{t_1}, 3o_b + 3o_2 + o_{s_2} + o_{t_3})$。系 2.5 を使用する条件は

$$o_{c1} = min(3o_b + 3o_2 + o_{s_1} + o_{t_1}, 3o_b + 3o_2 + o_{s_2} + o_{t_3}) \tag{28}$$

あっていたら、$\frac{c_1x_1+c_2(x_1)}{p_N^{o_{c1}}}$ は置換多項式である。もしスペクトラムに着目したら、$\Delta_0(s_1), \Delta_0(s_2), \Delta_1(s_1)$ と $\Delta_1(s_2)$ を解くことになる。

1.5 o_b の上界

入力重み 2 エラーイベントの分析より、N が与えたとき、o_b が大きければ良いインタリーバを選ぶことができ、よい性能を得る。しかし、シミュレーションより、置換多項式の性能は o_b に従ってある値まで良くなって、その値を超えると性能が悪くなる。定理 2.5 で o_b の上界を見つけることができる。

1.5.1 入力重み 4 エラーイベントでの o_b の上界

(25) から始まる。条件 (27) があっていて、$o_{c0} \geq o_{c1}$ のとき、(25) で x_1 の解答があるなら、与えたエラーパターンにたいして x_1 から始まるエラーイベントがある。0 から N-1 のすべて x_1 が解答である特別な場合がある。$t_1 = t_2 = t, s_1 = s_2 = s$ のとき、(26) にある c_0 と c_1 は以下のようになる。

$$c_0 = -2b\Delta_0(s)tD(K)$$
$$c_1 = -2b\Delta_1(s)tD(K)$$

$o_{c0} \geq o_{c1}$ ということが簡単に見える。$o_{c0} = o_N$ のとき、(25) は全 0 の多項式になる。

1.5.2 入力重み 6 エラーイベントでの o_b の上界

(25) から始まる。条件 (28) があっていたら、c_0 と c_1 は以下のようになる。

$$
\begin{aligned}
c_0 =&[\,2b[\,t_3\Delta_0(s_2) - t_2\Delta_0(s_1)] \,+ 2bt_3t_1\Delta_1(s_2) \\
&+ 2bt_1t_3 + b(t_1^2 - t_2^2 - t_3^2) \\
&+ a(t_1 - t_2 + t_3) - (s_1 - s_2 + s_3)]\, D
\end{aligned} \tag{30}
$$

$$
c_1 = [\,2b(t_1 - t_2 + t_3) + 2b(t_3\Delta_(s_2) - t_2\Delta(s_1))] \tag{31}
$$

$D = D(max\ (k(s_1), k(s_2)))$ で $c_2(x_1)$ の係数のオーダーは o_c1 より大きいすべての　がエラーパタンの解答である場合に興味ある。
[2t,t,-t,s,-s,2s] の形を持つエラーパタンが大変重要である。最小ハミング距離に対応するエラーパタンは $t = \tau$ と $s = \pm t$ のときである。上記のエラーパタンで $t_1 - t_2 + t_3 = s_1 - s_2 + s_3 = 0$ それで、c_0 と c_1 は以下のようになる。

$$
c_0 = \left[\,-2bt[\,\Delta_0(-s) + \Delta_0(s)] \,- 4bt^2\Delta_1(-s)\right]\, D \tag{32}
$$

$$
c_1 = -2bt(\Delta_1(-s) + \Delta_1(s))D \tag{33}
$$

ここから進めるために二つの補題が必要である。補題 4.6 $\Delta_0(-s) + \Delta_0(s)$ のオーダーは $o_b + o_2 + 2o_b$ 補題 4.7 $\Delta_1(-s) + \Delta_1(s)$ のオーダーは少なくとも $o_b + 2o_2 + 2o_b$

それで c_0 と c_1 のオーダーを計算することができる。

$$
o_{c0} \geq 2o_b + 2o_2 + 3o_\tau
$$

$$
o_{c1} \geq 2o_b + 3o_2 + 3o_\tau
$$

ベクトル o_N にあるそれぞれのメンバー $2o_b + 2o_2 + 3o_\tau$ に対応するメンバーより大きくないとき c_0 と c_1 は $0 mod N$ になって、すべての x_1 は式の解答である。o_b の上界とすることができる。

1.6 a と b を探索するときの範囲

補題 4.1 と補題 4.2 で、入力重み 2 エラーイベントのとき、o_b が決めたら $b = b_0 \cdot p_N^{o_b}$ として a は 1 から b_0 しか注目しない。残念ながら一般的な場合でも a の範囲の結果はだいたい同じである。

二次順列多項式に基づいたインタリーバで、a の範囲は 1 から 2b を注目しなければない。
入力重み 4 エラーイベントでは一般的に、o_b が与えたら、すべての b、そして $1 \leq a \leq 2b$ さがさなければならない。これは退屈である。しかし、ある条件で補題 4.1 よりの b を使うことができる。ゆえに、入力重み 2 エラーイベントのスペクトラムを使って、多項式を探索するとき、$b = p_N^{o_b}$ と $1 \leq a \leq 2b$ を着目する。

2 結果

フレームサイズ N と要素符号にが与えられたら、良い置換多項式に基づいてインタリーバを探すことは、多項式の a と b を計算することになる。最初に、o_b の値を決める。前の分析で $p_N^{o_b}$ を大きくしなければならないですが、特別入力重み 4 エラーイベントと入力重み 6 エラーイベントで成約を拘束しなければならない。o_b が決めたら、$b = p_N^{o_b}$ として定理 4.8 の範囲ですべての a を計算する。

6 種類の要素符号が選ばれて、テーブル 2 に書かれている。フレームサイズを $N = 2^n$ とし,N のベースを $p_N = 2$ になり、N のオーダーはスカラーになる。$N = 2^8$ の場合、要素符号に対して最良な置換多項式そして、入力重み 2 エラーイベントに対する最低距離と多重度がテーブル 2 に書かれている。シムレーションで置換多項式に基づいてインタリーバを S ーランダムインタリーバと二次インタリーバと比べた結果は、図 6-11 で示される。置換多項式に基づいたインタリーバは常に二次インタリーバと S ーランダムインタリーバよりいい働をする。

要素符号を RC5/7 符号、フレームサイズ N を 1024 と 16384 とし、それぞれのインタリーバの最良置換多項式は $P(x) = 31x + 64x^2$ と $P(x) = 15x + 32x^2$ に基づく。シムレーションでの結果は図 12 と 13 に示される。長いフレームサイズの場合、置換多項式に基づいたインタリーバの性能は、二次インタリーバより良いですが、S ーランダムインタリーバほどよくないということがわかる。