Декодирование Q–ных плетеных сверточных МПП–кодов

В. В.Зяблов

Институт проблем передачи информации, Российская академия наук, Москва, Россия

zyablov@iitp.ru

К. А. Кондрашов

Институт проблем передачи информации, Российская академия наук, Москва, Россия

k_kondrashov@iitp.ru

Аннотация

В настоящей работе рассматриваются два q–ных плетеных сверточных кода с малой плотностью проверок (П–СМПП) с двумя и четырьмя кодами–компонентами Рида–Соломона. Исследуются корректирующие способности при жестком декодировании мажоритарным алгоритмом и алгоритмом с введением стираний.

1. Введение

Сверточные МПП–коды [1] , обладая хорошей практической возможностью применения, постепенно начинают привлекать все больше и больше внимания. В данной работе исследуются возможности декодирования q-ных плетеных сверточных МПП (j,n_0)–кодов с (n_0,k_0)–кодами–компонентами Рида–Соломона, где j — это количество кодов–компонентов, входящих в плетеный код, а n_0,k_0 — их параметры. В [2] был предложен двоичный П–СМПП $(2,7)$–код с кодами–компонентами Хэмминга. Представленный код легко обобщается на случай недвоичного алфавита. При переходе к алфавиту большей размерности особый интерес представляет использование в качестве кодов–компонентов простых кодов с одной проверкой, что позволяет существенно упростить декодирование. При этом для получения хорошей корректирующей способности, очевидно, необходимо увеличить число кодов–компонентов. В [3] был предложен q-ный П–СМПП $(4,8)$–код, описана процедура кодирования и исследованы его дистанционные характеристики. Данная статья продолжает исследование корректирующих способностей П-СМПП $(2,7)$– и $(4,8)$–кодов.

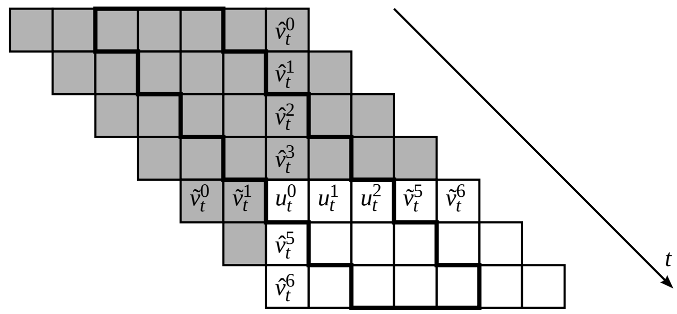


Рис. 1. Представление П–СМПП $(2,7)$–кода в виде двумерного массива. Серым выделены закодированные символы.

2. П–СМПП $(2,7)$–код

Плетеный сверточный МПП–код, построенный путем пересечения двух блоковых кодов–компонентов — горизонтального и вертикального — был впервые предложен в [2]. Кодовый массив состоит из трех полос. (Рис. 1). При кодировании символы информационной последовательности $\boldsymbol{u} = [\boldsymbol{u}_0\ \boldsymbol{u}_1\ \ldots\ \boldsymbol{u}_t\ \ldots]$ помещаются в центральную полосу. Проверочные символы кодовых слов вертикального кода–компонента помещаются в нижнюю полосу. Вместе с информационными символами они образуют информационную часть кодов горизонтального кода–компонента, проверочная часть которых помещается в верхнюю полосу и, в свою очередь, формирует информационную часть слов вертикального кода–компонента. Рассмотрим П–СМПП $(2,7)$–код с кодами–компонентами $(7,5)$–кодами Рида–Соломона. В произвольный момент времени t информационный блок $\boldsymbol{u}_t = \left[u_t^0\ u_t^1\ u_t^2\right]$ помещается в центральную полосу. Затем горизонтальный код–компонент кодирует блок $\left[\tilde{v}_t^0\ \tilde{v}_t^1\ u_t^0\ u_t^1\ u_t^2\right]$, выдавая $\left[\tilde{v}_t^0\ \tilde{v}_t^1\ u_t^0\ u_t^1\ u_t^2\ \tilde{v}_t^5\ \tilde{v}_t^6\right]$. Вертикальный код–

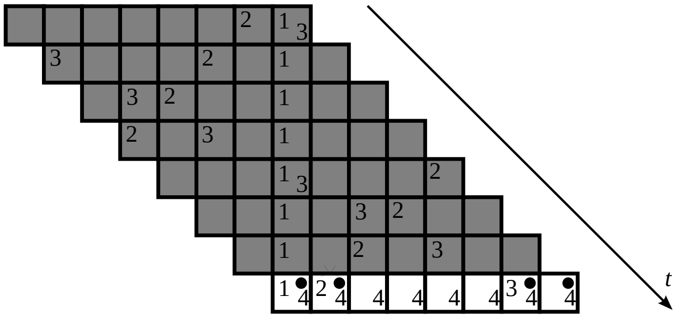


Рис. 2. Представление П–СМПП $(4,8)$–кода в виде двумерного массива. Индексами обозначена принадлежность символов к одному из четырех кодов–компонентов. Серым выделены закодированные символы. Круглым маркером отмечены позиции проверочных символов.

компонент кодирует блок $\left[\hat{v}_t^0\ \hat{v}_t^1\ \hat{v}_t^2\ \hat{v}_t^3\ u_t^0\right]$, выдавая $\left[\hat{v}_t^0\ \hat{v}_t^1\ \hat{v}_t^2\ \hat{v}_t^3\ u_t^0\ \hat{v}_t^5\ \hat{v}_t^6\right]$. Формируется кодовый блок $\boldsymbol{v}_t = \left[u_t^0\ u_t^1\ u_t^2\ \tilde{v}_t^5\ \tilde{v}_t^6\ \hat{v}_t^5\ \hat{v}_t^6\right]$. Результирующая скорость кода $R = \frac{3}{7}$.

3. П–СМПП $(4,8)$–код

В П–СМПП $(2,7)$–коде из [2] используются сильные коды–компоненты с хорошей корректирующей способностью, но их количество ограничено двумя. Увеличение числа кодов–компонентов ценой их упрощения было предложено в [3]. Представленный в [3] П–СМПП $(4,8)$–код состоит из четырех $(8,7)$–кодов Рида–Соломона с одной q-ной проверкой на четность (Рис. 2). Кодирование осуществляется схожим образом. В произвольный момент времени t первым шагом кодируются вертикальный и горизонтальные коды–компоненты. Затем информационный блок $\boldsymbol{u}_t$ и полученные проверочные символы кодируются горизонтальным кодом–компонентом. Скорость результирующего кода $R = \frac{1}{2}$.

4. Кодирование

Представленные коды принадлежат классу сверточных МПП–кодов и, наряду со схематичным описанием, могут быть заданы полубесконечной проверочной матрицей сверточного МПП–кода:

$$\boldsymbol{H}^{\mathrm{T}} = \begin{pmatrix} \boldsymbol{H}_0^{\mathrm{T}}(0) & \dots & \boldsymbol{H}_{m_s}^{\mathrm{T}}(m_s) & & \\ & \ddots & & \ddots & \\ & & \boldsymbol{H}_0^{\mathrm{T}}(t) & \dots & \boldsymbol{H}_{m_s}^{\mathrm{T}}(t+m_s) \\ & & \ddots & & \ddots \end{pmatrix}, \tag{1}$$

где подматрицы $\boldsymbol{H}_i^{\mathrm{T}}(t)$ определяются проверочными матрицами кодов–компонентов и имеют размерность $n_0 \times j(n_0 - k_0)$, а m_s — размер памяти кода. В любой момент времени t кодовая последовательность удовлетворяет условиям:

$$\boldsymbol{v}_t \boldsymbol{H}_0^{\mathrm{T}}(t) + \boldsymbol{v}_{t-1} \boldsymbol{H}_1^{\mathrm{T}}(t) + \dots + \boldsymbol{v}_{t-m_s} \boldsymbol{H}_{m_s}^{\mathrm{T}}(t) = 0, t \in \mathbb{Z} \tag{2}$$

и

$$\boldsymbol{v}_{[0,t-1]} \boldsymbol{H}_{[0,t+m_s-1]}^{\mathrm{T}} = \left[\boldsymbol{0}_{[0,t-1]} | \boldsymbol{s}_t\right], \tag{3}$$

где $\boldsymbol{s}_t = \left[\boldsymbol{s}_t^0 \boldsymbol{s}_t^1 \dots \boldsymbol{s}_t^{m_s-1}\right]$ — вектор частичных синдромов, обновляющийся по рекуррентному закону:

$$\boldsymbol{s}_t^i = \begin{cases} \boldsymbol{s}_{t-1}^{i+1} + \boldsymbol{v}_t \boldsymbol{H}_{i+1}^{\mathrm{T}}(t+i+1), & i = 0, \dots, m_s - 2 \\ \boldsymbol{v}_t \boldsymbol{H}_{i+1}^{\mathrm{T}}(t+i+1), & i = m_s - 1. \end{cases} \tag{4}$$

Алгоритм кодирования следует из (3). Кодовые блоки получаются в виде решения системы линейных уравнений

$$\boldsymbol{v}_{t+1} \boldsymbol{H}_0^{\mathrm{T}}(t+1) = -\boldsymbol{s}_t^0. \tag{5}$$

5. Дистанционные свойства кодов

Одной из важнейших характеристик кода, определяющих его корректирующие способности, является минимальное Хэммингово расстояние d_{min} между любыми двумя кодовыми словами. Для сверточных кодов, кодовые слова которых невозможно сравнивать на разных длинах, вводится аналог минимального кодового расстояния — свободное кодовое расстояние. Это минимальное расстояние между любыми кодовыми последовательностями.

$$d_{free} = \min_{\boldsymbol{v} \neq \boldsymbol{v}'} \left\{ d_{\mathrm{H}}(\boldsymbol{v}, \boldsymbol{v}') \right\}. \tag{6}$$

Для определения d_{free} удобно рассматривать активное расстояние кода d_j — минимальный вес кодовой последовательности, приводящей кодер в нулевое состояние после j информационных блоков.

$$d_j = \min\left\{\omega_{\mathrm{H}}(\boldsymbol{v}_{[1,j]})\right\} : \ \boldsymbol{v}_{[1,j]} \boldsymbol{H}_{[1,j+m_s-1]}^{\mathrm{T}} = \boldsymbol{0}. \tag{7}$$

Свободное и активные расстояния связаны отношением $d_{free} = \min_j \left\{d_j\right\}$. Активные расстояния для различных j мы будем искать среди решений системы линейных уравнений

$$x \boldsymbol{H}_{[1,j+m_s-1]}^{\mathrm{T}} = \boldsymbol{0}. \tag{8}$$

Найденные значения приведены на (Рис. 3.)

6. Декодирование

В данной работе мы рассматриваем два алгоритма жесткого декодирования: итеративный

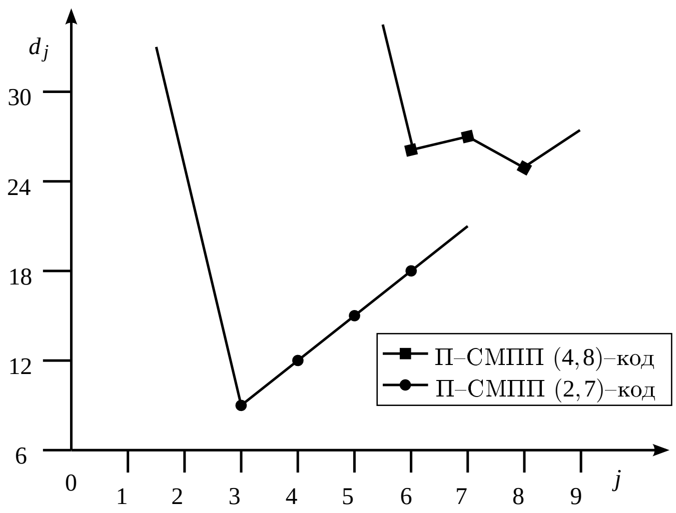


Рис. 3. Активные расстояния

мажоритарный алгоритм $\mathscr{A}_1$ и расширенный алгоритм $\mathscr{A}_2$ с введением стираний. Так как кодовое расстояние кодов–компонентов П–СМПП (4,8)–кода $d = 2$, они способны лишь обнаружить одну ошибку, но не исправить ее. Поэтому оба алгоритма $\mathscr{A}_1$ и $\mathscr{A}_2$ будут слегка различаться для П–СМПП (2,7)– и (4,8)–кодов. Мы опишем алгоритмы декодирования для П–СМПП (2,7)–кода, а затем уточним различия.

Рассмотрим мажоритарный алгоритм $\mathscr{A}_1$. Пусть на вход декодера на i-й итерации подается слово $\boldsymbol{r}^{(i)}$, где $\boldsymbol{r}^{(1)}$ – принятое из канала передачи искаженное кодовое слово. Тогда каждая итерация декодирования состоит из следующих шагов:

А л г о р и т м $\mathscr{A}_1$:

1. Для каждого кода–компонента k с помощью $\mathscr{D}^{(k)}$, где $\mathscr{D}^{(k)}$ – декодер кода–компонента k, декодируются все соответствующие ему слова из $\boldsymbol{r}^{(i)}$. Результаты запоминаются в $\boldsymbol{r}_k^{(i)}$.

2. Создается слово следующей итерации $\boldsymbol{r}^{(i+1)}$, символы $r_j^{(i+1)}$ которого определяются голосованием – большинством из $r_{k,j}^{(i+1)}$. Если выбор неоднозначен, то берется значение из входного слова $r_j^{(i)}$.

3. Вычисляется синдром $\boldsymbol{r}^{(i+1)}$. В зависимости от синдрома и номера итерации возможны четыре исхода: переход к следующей итерации, отказ от декодирования, ошибка декодирования, успех декодирования. Если синдром ненулевой и предел итераций не достигнут, осуществляется переход к следующей итерации. Отказ от декодирования происходит при ненулевом синдроме, если исчерпаны итерации или результирующее слово $\boldsymbol{r}^{(i+1)}$ не отличается от входного $\boldsymbol{r}^{(i)}$. Нулевой синдром завершает декодирование успехом или ошибкой, в зависимости от того, совпадает ли результирующее кодовое слово $\boldsymbol{r}^{(i+1)}$ с переданным.

Алгоритм $\mathscr{A}_2$ представляет собой модифицированный вариант алгоритма $\mathscr{A}_1$. Здесь, если после голосования выбор символа неоднозначен, символ заменяется новой буквой из расширенного алфавита – стиранием. Перед переходом к следующей итерации выполняется декодирование с исправлением стираний до полного их устранения.

А л г о р и т м $\mathscr{A}_2$:

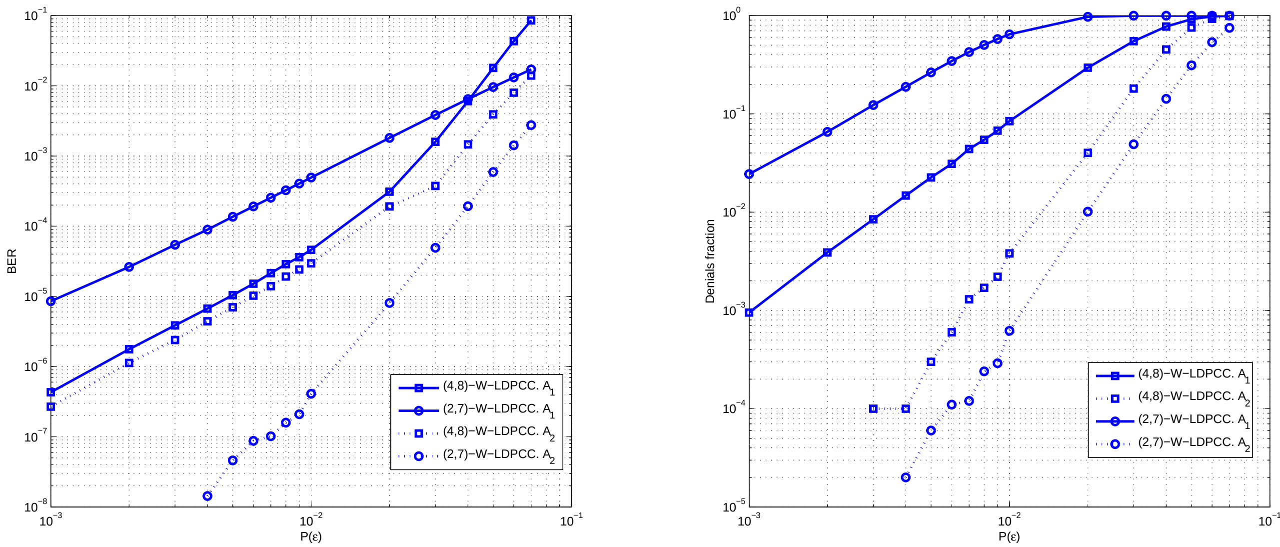
1. Для каждого кода–компонента k с помощью $\mathscr{D}^{(k)}$, где $\mathscr{D}^{(k)}$ – декодер кода–компонента k, декодируются все соответствующие ему слова из $\boldsymbol{r}^{(i)}$. Результаты запоминаются в $\boldsymbol{r}_k^{(i)}$.

2. Голосованием по $r_{k,j}^{(i+1)}$ определяются символы $r_j^{(i+1)}$ слова $\boldsymbol{r}^{(i+1)}$ следующей итерации. Если для некоторого символа выбор неоднозначен, его значение заменяется стиранием.

3. При наличии стираний результирующее слово $\boldsymbol{r}^{(i+1)}$ декодируется покомпонентно с помощью декодеров кодов–компонентов $\mathscr{E}^{(k)}$, исправляющих стирания. Решение об исправлении для каждого стертого символа принимается по результатам голосования среди кодов–компонентов. В голосовании участвуют коды–компоненты, $\mathscr{E}^{(k)}$ которых сумели исправить данный символ. Декодирование стираний повторяется до полного их устранения или превышения порога числа итераций. Если после декодирования количество стираний не уменьшилось, декодирование завершается отказом.

4. Вычисляется синдром $\boldsymbol{r}^{(i+1)}$. В зависимости от синдрома и номера итерации происходит переход к следующей итерации или завершение декодирования. Декодирование может завершиться отказом, ошибкой или успехом. Отказ от декодирования происходит при ненулевом синдроме, если исчерпаны итерации или результирующее слово $\boldsymbol{r}^{(i+1)}$ не отличается от результата предыдущей итерации $\boldsymbol{r}^{(i)}$. При нулевом синдроме декодирование завершается успехом или ошибкой, в зависимости от того, совпадает ли результирующее кодовое слово $\boldsymbol{r}^{(i+1)}$ с переданным.

В случае П-СМПП (4,8)–кода в обоих алгоритмах отпадает первый шаг. Так как коды–компоненты с $d = 2$ могут лишь обнаружить одну



(a) Вероятность ошибки на бит

(b) Доля отказов декодирования

Рис. 4. Результаты моделирования

ошибку, то процедура голосования преобразуется следующим образом. Для каждого символа $r_j^{(i+1)}$ вычисляются синдромы соответствующих кодам–компонентам слов, в которые входит данный символ. Решением от каждого кода–компонента будет значение, обращающее соответствующий синдром в ноль. Результирующее значение символа $r_j^{(i+1)}$ выбирается по большинству, а в случае неоднозначного выбора сохраняет значение $r_j^{(i)}$ или заменяется стиранием, в зависимости от алгоритма. Результаты моделирования приведены на Рис. 4.

Список литературы

[1] A. J. Felström and K. Sh. Zigangirov, Periodic time-varying convolutional codes wit low-density parity-check matrices, *IEEE Trans Inf. Theory*, vol. 45, no. 45, 2181–2190, 1999

[2] A. J. Felström, M. Lentmaier, D.V. Truhachev and K. Sh. Zigangirov, Braided block codes, *IEEE Trans Inf. Theory submission*, 2006

[3] V. V. Zyablov and K. A. Kondrashov, Two LDPCC-constructions *Information Technologies and Systems Workshop, Becasovo, Russia, 2009* 156–159.