Лекция 8

Ilya Yaroshevskiy

21 октября

Содержание

1 Субоптимальность полярных кодов

Алгоритм последовательно исключения

Program 1 Алгоритм последовательного исключения

1: **for** $i = 0, 1, \ldots, 2^m$ **do**

2: $\hat{u}_i = \begin{cases} 0 & , i \in \mathcal{F} \\ 0 & , L_m^{(i)}(y_0^{n-1}, \hat{u}_0^{i-1} > 0), i \notin \mathcal{F} \\ 1 & , L_m^{(i)}(y_0^{n-1}, \hat{u}_0^{i-1}) \leq 0, i \notin \mathcal{F} \end{cases}$

3: **end for**

При принятии решения относительно незамороженного символа не учитываются ограничения замораживания на символы $u_j, j \in \mathcal{F}, j > i$. Если принято неправильное решение $\hat{u}_i$, алгоритм последовательного исключения не способен далее его исправить. Минимальное расстояние слишком мало **[org:** Проблема: При оцениывнии каждого $\hat{u}_i$ не учитываем замораживание на будущие символы
]

2 Списочное кодирование

Не будет принимать окончательное решение относительно u_i на фазе i. На каждой фазе i будем рассматривать L векторов u_0^{i-1}, строить их возможные продолжения u_0^i и выбирать из них L наиболее вероятных. Вектора u_0^i задают пути в кодовом дереве Доделать Картинка

2.1 Приближенный алгоритм декодирования

$$W_\lambda^{(i)}(u_0^i|y_0^{2^\lambda-1}) = \frac{W_\lambda^{(i)}(y_0^{2^\lambda-1}, u_0^{i-1}|u_i)}{2W(y_0^{2^\lambda-1})} = \sum_{u_{i+1}^{2^\lambda-1}} W_\lambda^{(2^\lambda-1)}(u_0^{2^\lambda-1}|y_0^{2^\lambda-1})$$

$$W_\lambda^{(2i)}(u_0^{2i}|y_0^{2^\lambda-1}) = \sum_{u_{2i+1}d0}^{1} W_{\lambda-1}^{(i)}(u_{0,\text{even}}^{2i+1} + u_{0,\text{odd}}^{2i+1}|y_{0,\text{even}}^{2^\lambda-1}) W_{\lambda-1}^{(i)}(u_{0,\text{odd}}^{2i+1}|y_{0,\text{odd}}^{2^\lambda-1})$$

$$W_\lambda^{(2i+1)}(u_0^{2i+1}|y_0^{2^\lambda-1}) = W_{\lambda-1}^{(i)}(u_{0,\text{even}}^{2i+1} + u_{0,\text{odd}}^{2i+1}|y_{0,\text{even}}^{2^\lambda-1}) W_{\lambda-1}^{(i)}(u_{0,\text{odd}}^{2i+1}|y_{0,\text{odd}}^{2^\lambda-1})$$

Наиболее вероятное продолжение пути u_0^i (без учета замороженных символов $u_j = 0, j \in \mathcal{F}, j > i$)

$$\tilde{W}_\lambda^{(i)}(u_0^i|y_0^{2^\lambda-1}) = \max_{u_{i+1}^{2^\lambda-1}} W_\lambda^{(2^\lambda-1)}(u_0^{2^\lambda-1}|y_0^{2^\lambda-1})$$

$$\tilde{W}_\lambda^{(2i)}(u_0^{2i}|y_0^{2^\lambda-1}) = \max_{u_{2i+1}} \tilde{W}_{\lambda-1}^{(i)}(u_{0,\text{even}}^{2i+1} - u_{0,\text{odd}}^{2i+1}|y_{0,\text{even}}^{2^\lambda-1}) \tilde{W}_{\lambda-1}^{(i)}(u_{0,\text{odd}}^{2i+1}|y_{0,\text{odd}}^{2^\lambda-1})$$

$$\tilde{W}_\lambda^{(2i+1)}(u_0^{2i+1}|y_0^{2^\lambda-1}) = \tilde{W}_{\lambda-1}^{(i)}(u_{0,\text{even}}^{2i+1} + u_{0,\text{odd}}^{2i+1}|y_{0,\text{even}}^{2^\lambda-1}) \tilde{W}_{\lambda-1}^{(i)}(u_{0,\text{odd}}^{2i+1}|y_{0,\text{odd}}^{2^\lambda-1})$$

Определение. Вес пути в кодовом дереве

$$R_\lambda^{(i)}(u_0^i, y_0^{2^\lambda-1}) = \ln \tilde{W}_\lambda^{(i)}(u_0^i|y_0^{2^\lambda-1})$$

Модифицированное логарифмическое отношение правдоподобия

$$S_\lambda^{(i)}(u_0^{i-1}, y_0^{2^\lambda-1}) = R_\lambda^{(i)}(u_0^{i-1}.0, y_0^{2^\lambda-1}) - R_\lambda^{(i)}(u_0^{i-1}.1, y_0^{2^\lambda-1}) = \ln \frac{\tilde{W}_\lambda^{(i)}(u_0^{i-1}.0|y_0^{2^\lambda-1})}{\tilde{W}_\lambda^{(i)}(u_0^{i-1}.1|y_0^{2^\lambda-1})}$$

Если значение u_i соответствует наиболее вероятному продолжению пути u_0^{i-1}, то

$$R_m^{(i)}(u_0^i, y_0^{2^m-1}) = R_m^{(i-1)}(u_0^{i-1}, y_0^{2^m-1})$$

Если значение u_i не соответсвует наиболее вероятному продолжению пути u_0^{i-1}, то

$$R_m^{(i)}(u_0^i, y_0^{2^m-1}) = R_m^{(i-1)}(u_0^{i-1}, y_0^{2^m-1}) - (R_m^{(i)}(u_0^{i-1}.1 - u_i, y_0^{2^m-1}) - R_m^{(i)}(u_0^{(i-1)}.u_i, y_0^{2^m-1}))$$

$$R_m^{(i)}(u_0^i, y_0^{n-1}) = R_m^{(i-1)}(u_0^{i-1}, y_0^{n-1}) + \tau(S_m^{(i)}(u_0^{i-1}, y_0^{n-1}), u_i)$$

$$\tau(S, v) = \begin{cases} 0 & , \text{if } \operatorname{sgn}(S) = (-1)^v \\ -|S| & , \text{otherwise} \end{cases}$$

Модифицированные логарифмические отношения правдоподобия

$$\begin{aligned} S_\lambda^{(2i)}(u_0^{2i-1}|y_0^{2^\lambda-1}) &= max(J(0)+K(0), J(1)+K(1)) - max(J(1)+K(0), J(0)+K(1)) \\ &= max(J(0)-J(1)+K(0)-K(1), 0) - max(K(0)-K(1), J(0)-J(1)) \end{aligned}$$

где

$$J(c) = R_{\lambda-1}^{(i)}((v_{0,\text{even}}^{2i-1} \oplus v_{0,\text{odd}}^{2i-1}.c|y_{0,\text{even}}^{2^\lambda-1}), K(c) = R_{\lambda-1}^{(i)}(v_{0,\text{even}}^{2i-1}|y_{0,\text{even}}^{2^\lambda-1})$$

$$J(0) - J(1) = a = S_{\lambda-1}^{(i)}(v_{0,\text{even}}^{2i-1} \oplus v_{0,\text{odd}}^{2i-1}|y_{0,\text{even}}^{2^\lambda-1}), K(0) - K(1) = b = S_{\lambda-1}^{(i)}(v_{0,\text{odd}}^{2i-1}|y_{0,\text{odd}}^{2^\lambda-1})$$

2.2 Декодер min-sum

$$S_\lambda^{(2i)}(u_0^{2i-1}, y_0^{2^\lambda-1}) = Q(a,b) = \operatorname{sgn}(a)\operatorname{sgn}(b)\min(|a|,|b|)$$

$$S_\lambda^{(2i+1)}(u_0^{2i}, y_0^{2^\lambda-1}) = P(u_{2i}, a, b) = (-1)^{u_{2i}} a + b$$

$$a = S_{\lambda-1}^{(i)}(v_{0,\text{even}}^{2i-1} \oplus v_{0,\text{odd}}^{2i-1}|y_{0,\text{even}}^{2^\lambda-1}), b = S_{\lambda-1}^{(i)}(v_{0,\text{odd}}^{2i-1}|y_{0,\text{odd}}^{2^\lambda-1})$$

Program 2 Алгоритм последовательного исключения

1: **for** $i = 0, 1, \dots, 2^m - 1$ **do**

2: $\hat{u}_i = \begin{cases} 0 & , i \in \mathcal{F} \\ 0 & , S_m^{(i)}(y_0^{n-1}, \hat{u}_0^{i-1} > 0), i \notin \mathcal{F} \\ 1 & , S_m^{(i)}(y_0^{n-1}, \hat{u}_0^{i-1}) \le 0, i \notin \mathcal{F} \end{cases}$

3: **end for**

2.3 Списочный алгоритм Тала-Варди

- Пусть $V[l]$ – l-ый вектор u_0^{i-1}, рассматриваемый декодеромб $0 \le l < L$
- Пусть $R[l] = R_m^{(i-1)}(V[l], y_0^{n-1})$

Program 3 Списочный алгоритм Тала-Варди

for $i = 0, 1, \dots, 2^m - 1$ **do**
  Вычислить $s_l = S_m^{(i)}(V[l], y_0^{n-1}), 0 \le l < L$
  if $i \in \mathcal{F}$ **then**
    Дописать к $V[l]$ значение замороженного символа, $R[l] \leftarrow R[l] + \tau(s_l, (V[l])_i)$
  else
    Выбрать L пар (l, v) с наибольшим занчением r_{lv}, где $r_{lv} = R[l] + \tau(s_l, v), v \in \{0, 1\}, 0 \le l < L$
    Для i-той выбранной пары (l, v) построить продолженный путь
    $V'[i] \leftarrow V[i].v, R'[i] \leftarrow r_{lv}, 0 \le i < L$
    $V \leftarrow V', R \leftarrow R'$
  end if
end for

Из полученного списка выбрать наилучший путь:

- С наибольшим $R[l]$
- Удовлетворяющий некоторым дополнительным условиям

2.4 Частичные суммы

Полярный код получается путем рекурсивного применения конструкции Плоткина $c = (c_1 + c_0, c_1)$. Массивы частичных сумм C_λ размерности $2^{m-\lambda}, 0 \le \lambda \le m$.

- В массиве C_0 размерности 2^m будем формировать кодовое слово c
- В массиве C_1 размерности 2^{m-1} будем офрмировать кодове слово c_0
- c_0 также является кодовым словом некоторого полярного кода. Соответствующий вектор c_00 будем формировать в C_2
- ...
- По готовности кодовых слов будем применять преобразование Плоткина и высвобождать массивы под кодовые слова следующий компонентных кодов

$u_0 = 0$

2	1	0
0		

$u_1 = 1$

2	1	0
0	$1 = 0 + 1$	
	1	

$u_2 = 1$

2	1	0
1	1	
	1	

$u_3 = 0$

2	1	0
1	1	$0 + 1 + 1$
	1	$0 + 1$
		$0 + 1$
		0

Массив ЛОПП S_λ размерности $2^{m-\lambda}, 0 \le \lambda \le m$. S_0 содержит ЛОПП принятого вектора. На i-той итерации алгоритма последовательного исключения обновляются массивы $S_{m-l}, \dots, S_m$, где t – масимальная степень 2, делящая $i, t < m$

$$S_{m-t}[j] = P(C_{m-t-1}[j], Sm - y - 1[j], S_{m-t-1}[j + 2^t]), 0 \le j < 2^{m-t}$$

$$S_\lambda[j] = Q(S_{\lambda-1}[j], S_{\lambda-1}[i + 2^{m-\lambda}]), m - t < \lambda \le m$$

При $i = 0$ считается $t = m$, первая формула не используется

Замечание. Многие массивы частичных сумм и ЛОПП совпадают у различных путей. Копирование данных может быть исключено полностью

2.5 Полярные коды с CRC

Определение. Cyclic redundancy check – циклический код, обнаруживающий ошибки

Замечание. Систематическое кодирование циклического кода длины n с порождающим многочленом $g(x)$

$$c(x) = a(x)x^{n-k} + b(x), b(x) \equiv a(x)x^{n-k} \mod g(x)$$

Добавим к данным проверочные символы ($b(x)$ перед их кодированем полярным кодом. Удалим из списка, формируемого декодером Тала-Варди кодовые слова с направильным значением контрольной суммы.

2.6 Динамически замороженные символы

Классические полярные коды: замороженные символы $u_i = 0, i \in \mathcal{F}$. Обобщение: $u_i = \sum_{j=0}^{i-1} V_{s_i,j} u_j, i \in \mathcal{F}$

$$uV^T = 0$$

Program 4 Алгоритм последовательного исключения

1: **for** $i = 0, 1, \dots, 2^m - 1$ **do**

2: $\hat{u}_i = \begin{cases} \sum_{j=0}^{i-1} V_{s_i,j} \hat{u}_j & , i \in \mathcal{F} \\ \operatorname{argmax}_{u_i} W_m^{(i)}(y_0^{n-1}, \hat{u}_0^{i-1} | u_i & , i \notin \mathcal{F} \end{cases}$

3: **end for**

Вероятность ошибки (алгоритм ПИ) $P_{SC} \le \sum_{i \notin \mathcal{F}} Z_{m,i}$. Совпадает с таковой для классических полярных кодов с тем же $\mathcal{F}$. Непосредственное обощение сисочного алгоритма Тала-Варди.

Как выбрать матрицу V:
Рассмотрим $(n = 2^m, k)$ линейный блоковый код с проверочной матрицей H. Пусть $V' = HA_m^T$. Применим к V' элементарные операции над строками, так, чтобы последние ненулевые элементы строк расположились в различных столбцах j_i. Пусть $V = QHA_m^T$ – полученная матрица. Алгоритм последовательного исключения можно использовать для декодирования произвольных линейных блоковых кодов.

2.7 Декодирование линейных кодов методов ПИ и его аналоги

- В общем случае вероятность ошибки декодирования методом ПИ намного больше вероятности ошибки декодирования по максимуму правдоподобия
- Перестановка столбцов проверочной матрицы (переход к эквивалентному коду) может радикально изменить множество $\mathcal{F}$ и P_{SC}
- Для расширенных примитивных кодов БЧХ в узком смысле P_{SC} достаточно мала $\implies$ списочный алгоритм Тала-Варди с небольшим рамером списка может обеспечить хорошую корректирующую способность

2.7.1 Списочное декодирование полярных кодов и кодов БЧХ

- Полярные коды минимизируют вероятность ошибки $P_{SC}(L = 1)$
- При $L = 4$ обеспечивается декодирование полярных кодов по максимуму правдоподобия
- При $L = 256$ обеспечивается декодирование кодо БЧХ почти по максимуму правдоподобия
- Болшее минимальное расстояние обеспечивает лучшую корректирующую способность кодов БЧХ

2.8 Полярные коды в узком смысле

- Выберем линейный блоковый код (протокод) с достаточно большим минимальным расстоянием
- Удалим из него кодовые слова, препятствующие эффективному декодированию методом последовательного исключения

Определение. Рассмотрим канал $W(y|c)$ и $(n = 2^m, k', d)$ код $\mathcal{C}'$ над $GF(2)$, называемый протокодом. Пусть $\mathcal{F}'$ – множество номеров замороженных символов $\mathcal{C}'$. $(n, k, \geq d)$ полярным подкодом $\mathcal{C}$ в узком смысле кода $\mathcal{C}'$ называется множество векторов $c_0^{n-1} = u_0^{n-1} A_m$, где u_0^{n-1} одновременно удовлетворяет ограничениям замораживания кода $\mathcal{C}'$, а также дополнительным ограничениям $u_s = 0$ для $k' - k$ номеров $s \notin \mathcal{F}'$ с наибольшими вероятностями ошибки $P_{m,s}$.

Замечание. Расширенные примитивные коды БЧХ в узком смысле – хорошие протокоды

Матрица ограничений и матрица прекодирования:

$$c_0^{n-1} = u_0^{n-1} A_m, \quad u_0^{n-1} V^T = 0$$

$$u_0^{n-1} = xW, \quad WV^T = 0$$

- V – матрица ограничений (аналог проверочной матрицы)
- W – матрица прекодирования (аналог порождающей матрицы)

2.9 Полярные коды в широком смысле

- Выберем полярный код $(n = 2^m, n - r)$, эффективно декодируемый методом ПИ
- Удалим их него кодовые слова, оствтственные за высокую вероятность ошибки декодирования МП, построив его подкод размерности $k < n - r$.

Определение. Полярным кодом в широком смысле называется множество векторо

$$c = x\mathbb{W}A_m, x \in FG(2)^k$$

где матрица $\mathbb{W}$ имеет нулевые стобцы в позициях, соответствующих r нименее надежным подканалам $W_m^{(j)}$

3 Выводы

- Полярные коды достигают предела Шеннон, имеют простые процедуры построения, кодирования и декодирования
- Корректирующая способность полярных кодов и алгоритма последовательного исключения неудовлетворительная
- Улучшенные декодеры: списочный и последовательный алгоритмы, метод распространения доверия
- Полряные подкоды и полярные коды с CRC на длинах до нескольких тысяч имеют лучшую корректирующую способность и меньшую сложность декодирования (при использовании последовательного декодирования) по сравнению с LDPC и турбо-кодами
- Алгоритм последовательного исключения и его аналоги плохо распараллеливаются и имеют большую задержку