УДК 681.391

УПРОЩЕННЫЕ АЛГОРИТМЫ ДЕКОДИРОВАНИЯ КОДОВ С НИЗКОЙ ПЛОТНОСТЬЮ ПРОВЕРОК НА ЧЕТНОСТЬ, ОСНОВАННЫЕ НА АЛГОРИТМЕ РАСПРОСТРАНЕНИЯ ДОВЕРИЯ

Лихобабин Е.А., *научный сотрудник кафедры телекоммуникаций и основ радиотехники Рязанского государственного радиотехнического университета, e-mail: tor@rsreu.ru*

Рассмотрен алгоритм распространения доверия для декодирования кодов с низкой плотностью проверок на четность (LDPC) и его упрощенные модификации. Представлена эффективность для кода N=1008, K=504 и скорость сходимости рассмотренных алгоритмов, а также сложность их реализации.

Ключевые слова: помехоустойчивое кодирование, декодирование, низкоплотностный код, LDPC.

Введение

Коды с низкой плотностью проверок на четность (low density parity check codes, LDPC) LDPC коды были предложены Р.Галлагером [1] еще в 1963 году, однако были забыты почти на 40 лет, в связи со сложностью реализации предложенного Галлагером итеративного алгоритма декодирования этих кодов. Новая волна интереса к LDPC кодам возникла в ответ на открытие, так называемых, турбо-кодов [2]. В настоящее время LDPC коды все шире применяются на практике, так стандарты DVB-T2, DVB-S2, DVB-C2, WiFi, WiMax, IEEE 802.15.3 уже используют эти коды.

Как было замечено выше, у алгоритма, предложенного Галлагером, есть один существенный недостаток – его реализация требует существенных вычислительных затрат. В связи с чем появилось множество попыток упростить этот алгоритм с минимально возможными потерями в эффективности декодирования. В данной статье приводится обзор наиболее интересных модификаций алгоритма Галлагера, особенности их реализации, а также сравнение эффективности и вычислительных затрат.

Графическое представление LDPC кода

Здесь и далее будем использовать следующие обозначения: N – число информационных бит в кодовом слове, M – число проверочных бит в кодовом слове. MxN – размерность проверочной матрицы LDPC кода, d_r – Хэммингов вес строки, d_c – Хэммингов вес столбца.

LDPC код представляется графом Таннера [4] (рис. 1). Узлы в графе Таннера называются информационными (или символьными) и проверочными узлами, которые мы обозначим как VN и CN, соответственно. Можно отметить, что граф будет содержать M проверочных узлов CN, по одному для каждого проверочного уравнения, и N информационных узлов VN, по одному для каждого символа кодового слова. Введем также следующие обозначения: $N(i)$, $M(j)$ – множества позиций единиц в i-ом столбце и j-ой строке матрицы $\mathbf{H}$ соответственно. $N(i)$-$\{j\}$ и $M(j)$-$\{i\}$ – множества позиций единиц в i-ом столбце и j-ой строке матрицы $\mathbf{H}$, за исключением j-ой и i-ой позиций соответственно.

С целью анализа эффективности реализации рассмотрим основные алгоритмы декодирования LDPC кодов.

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

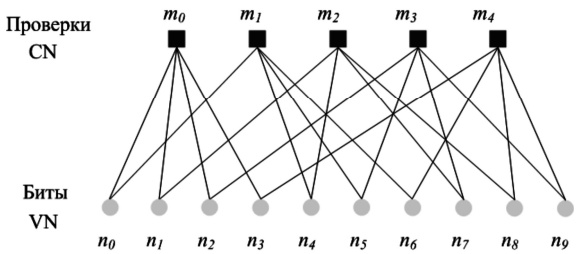

Рис. 1. Проверочная матрица и граф Таннера для линейного блокового кода (10,5), d_c=2, d_r=4.

Декодирование LDPC кодов

Помимо нового класса кодов Р.Галлагер в своей работе предложил близкий к оптимальному алгоритм декодирования, названный им алгоритмом «распространения доверия» (алгоритм РД, АРД, belief propagation – BP), также известного как алгоритм «сумма-произведение» (алгоритм СП, АСП, sum-product algorithm – SPA). Самым существенным недостатком предложенного Галлагером алгоритма является высокая сложность его реализации. Следует отметить тот факт, что АРД имеет вероятностное описание, однако, как указал Галлагер, возможна интерпретация в терминах логарифмических отношений правдоподобия (ЛОП, log likelihood ratios, LLR). Такой подход проще в реализации, за счет отсутствия необходимости в нормировании вероятностей на каждой итерации алгоритма, и алгоритм становится численно стабильнее. В связи с этим, в предложенном Галлагером вероятностном виде алгоритм практически не применяется, а используется его ЛОП-интерпретация. Здесь не будем отступать от сложившейся тенденции и, без потери общности, сосредоточимся на рассмотрении алгоритмов в терминах ЛОП. Алгоритм одинаково верен для любых каналов без памяти, поэтому наше рассмотрение будет максимально общим.

С выводом алгоритма АРД можно ознакомиться в [1, 3], а на русском языке в [5]. Приведем только результат вывода.

Алгоритм распространения доверия

1 шаг. Инициализация. Для всех узлов i инициализировать значения L_i, используя выражение

$$L_i = L(x_i \mid y_i) = \ln\left(\frac{p(x_i = 0 \mid y_i)}{p(x_i = 1 \mid y_i)}\right), \quad (1)$$

в зависимости от используемой модели канала связи. Затем для всех i и j, для которых h_{ij}=1, устанавливается $L_{i\to j} = L_i$.

2 шаг. Обновление проверочных узлов. Для всех проверочных узлов CN вычислить по формуле (2) исходящие сообщения

$$L_{j\to i} = \boxplus_{i'\in M(j)-\{i\}} L_{i'\to j}, \quad (2)$$

и передать их соответствующим информационным узлам VN.

3 шаг. Обновление битовых узлов. Вычислить сообщения $L_{i\to j}$, исходящие от информационных узлов VN

$$L_{i\to j} = L_i + \sum_{j'\in N(i)-\{j\}} L_{j'\to i},$$

и передать их соответствующим проверочным узлам.

4 шаг. Вычисление апостериорных ЛОП. Для всех *j=0,1...,N-1* вычислить

$$L_i^{total} = L_i + \sum_{j\in N(i)} L_{j\to i}.$$

5 шаг. Получение жестких решений. Для всех *i=0,1...,N-1* найти жесткие решения

$$\hat{v}_i = \begin{cases} 1, & L_i^{total} < 0, \\ 0, & L_i^{total} \ge 0. \end{cases}.$$

6 шаг. Проверка условия остановки. Вычислить синдром $\hat{v}H^T$. Если $\hat{v}H^T = 0$ или число итераций достигло максимума, вычисления прекращаются, а $\hat{v}$ считается результатом декодирования, в противном случае, вычисления продолжаются с шага 2.

В описании алгоритма $\boxplus$ – оператор суммы ЛОП (box plus, в зарубежной литературе).

$$\boxplus x_{i\to j} = 2\,\mathrm{th}^{-1}\left(\prod_{i'\in N(j)-\{i\}} \mathrm{th}\left(\frac{1}{2}x_{i\to j}\right)\right). \quad (3)$$

Реализация функций th и th^{-1} (АРД th) и в (3) является сложной вычислительной задачей, и Галлагер [1, 3] предложил упрощенный вариант вычисления (далее АРД) этого выражения

$$\boxplus_{i'\in N(j)-\{i\}} L_{i'\to j} = \prod_{i'\in N(j)-\{i\}} \alpha_{i'j} \cdot \phi\left(\sum_{i'\in N(j)-\{i\}} \phi(\beta_{i'j})\right), \quad (4)$$

где

$$L_{i\to j} = \alpha_{i\to j}\beta_{i\to j},$$
$$\alpha_{i\to j} = sign(L_{i\to j}),$$
$$\beta_{i\to j} = \left|L_{i\to j}\right|,$$
$$\phi(x) = -\ln[\mathrm{th}(x/2)] = \ln\left(\frac{e^x+1}{e^x-1}\right), \quad (5)$$

используя тот факт, что $\phi^{-1}(x) = \phi(x)$ при x>0. График функции $\phi(x)$ приведен на рис. 2.

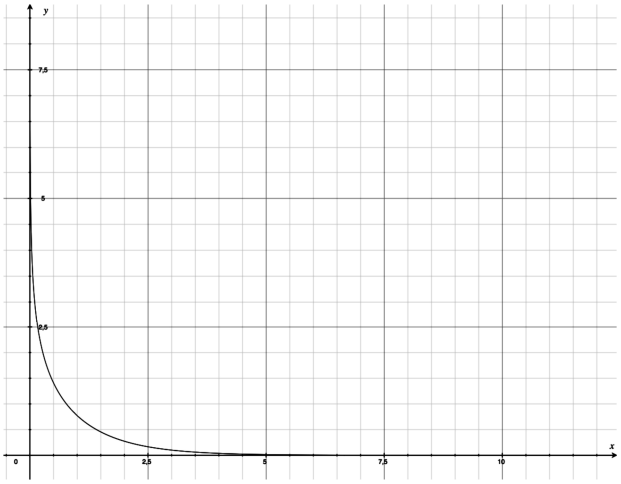

Рис. 2. График функции $\phi(x) = -\ln[\mathrm{th}(x/2)]$

Вычисление функции $\phi(x)$ может быть реализовано с помощью таблицы. Однако, для низких значений вероятности битовой ошибки, у декодера, основанного на табличном способе реализации функции $\phi(x)$, обычно начинает проявляться так называемый эффект “дна” (error floor)[3]. Одна из возможных альтернатив табулированию – использование кусочно-линейной аппроксимации этой функции [2, 3], другая альтернатива будут рассмотрены ниже.

Упрощенные алгоритмы

Основным недостатком АРД является его вычислительная сложность, а именно второй шаг алгоритма – обновление проверочных узлов, поскольку требует вычисления специальных функций. Поскольку алгоритм итеративный, ситуация еще более усугубляется, так как все шаги алгоритма выполняются многократно. Естественным направлением исследований стали попытки упрощения этого шага с наименьшими потерями в производительности. На сегодняшний день известно более десятка различных модификаций АРД [3, 5, 6, 7, 8]. Рассмотрим наиболее интересные из них.

Алгоритм Ричардсона-Новичкова

Реализация функции $\varphi(x)$ (5) табличным способом приемлема для некоторых программных реализаций алгоритма, но при реализации на большинстве аппаратных платформ такой подход не годится. Причина в том, что благодаря широкому динамическому диапазону (рис. 2) эта функция сложна для аппроксимации даже очень большой таблицей. При аппаратной реализации это свойство проявляется в существенном падении эффективности декодирования, особенно в области близкой к «дну».

Однако, несмотря на эти соображения, в [9] описан алгоритм, названный алгоритмом Ричардсона-Новичкова (далее АРН), предлагающий аппроксимацию именно этой функции, дружественную для аппаратной реализации. Выражение для расчета сообщений от проверочных узлов имеет вид:

$$| L_{j\to i} |= \phi^{-1}\left(\sum_{i'\in N(j)-\{i\}} \phi(\beta_{i'j})\right). \quad (6)$$

Учитывая тот факт, что для больших x справедливо $\log(1\pm e^{-x})\approx \pm e^{-x}$, получим

$$\phi(x)=\log\left(\frac{1+e^{-\beta_{i'j}}}{1-e^{-\beta_{i'j}}}\right)= \quad (7)$$

$$=\log(1+e^{-\beta_{i'j}})-\log(1-e^{-\beta_{i'j}})\approx 2e^{-\beta_{i'j}}$$

$\hat{\phi}(\beta_{i'j})=2e^{-\beta_{i'j}}$ – является достаточно точной аппроксимацией $\phi(\beta_{i'j})$, для $\beta_{i'j}>>\ln(2)$.

При этом обратная функция

$$\hat{\phi}^{-1}(x)\approx -\ln\left(\frac{x}{2}\right). \quad (8)$$

Эта аппроксимация достаточно точна для $x<<\ln(2)$. Иллюстрация аппроксимации выражения (5) двумя приведенными функциями представлена на рис. 3.

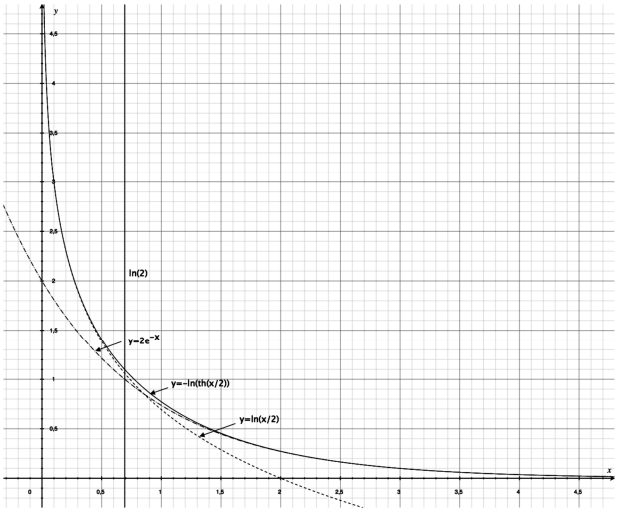

Рис. 3. Графики функций $\phi(x)=-\ln[\mathrm{th}(x/2)]$, $y(x)=2e^{-x}$ *и* $y(x)=-\ln(x/2)$

Подставляя (7) и (8) в (6), получим:

$$| L_{j\to i} |= -\ln\left(\sum_{i'\in N(j)-\{i\}} e^{-\beta_{i'j}}\right).$$

Поскольку $\hat{\phi}(\beta_{i'j})$ – грубая аппроксимация $\phi(\beta_{i'j})$ $\beta_{i'j}<<\ln(2)$, а $\hat{\phi}^{-1}(x)$ – грубая аппроксимация $\phi^{-1}(x)$ для $x>>\ln(2)$, логично, что при расчете сообщения от проверочного узла необходимо использовать корректировочные слагаемые C_1 и C_2.

Окончательно имеем:

$$| L_{j\to i} |= C_1-\ln\left(\sum_{i'\in N(j)-\{i\}} e^{-\beta_{i'j}}+C_2\right). \quad (9)$$

Дружественность алгоритма аппаратной реализации состоит в том, что путем достаточно простых преобразований в выражении (9) можно заменить натуральный логарифм и возведение в степень экспоненты, соответственно двоичным логарифмом и возведением в степень 2.

Алгоритм показывает хорошую эффективность для различных кодов, к недостаткам следует отнести то, что он защищен патентом [9].

Алгебра ЛОП

Обновление проверочных узлов можно представить в несколько ином виде, во многих случаях более удобном для реализации и позволяющем получить достаточно простые (с точки зрения реализации) аппроксимации. Но для этого нам потребуется следующий важный результат [10].

Для двух независимых случайных величин a_1 и a_2 с вероятностями $p(a_i=x)=p_x^{(l)}, x\in\{0,1\}$ и ЛОП $L_i=L(a_i)=\ln(p_0^{(i)}/p_1^{(i)})$, можно показать, что ЛОП их суммы $A_2=a_1+a_2$ равно:

$$\mathrm{L}(\mathrm{A}_2)=\ln\left(\frac{1+\mathrm{e}^{\mathrm{L}_1+\mathrm{L}_2}}{\mathrm{e}^{\mathrm{L}_1}+\mathrm{e}^{\mathrm{L}_2}}\right). \quad (10)$$

Этот факт может быть непосредственно применен для вычисления исходящего из проверочного узла CN сообщения в случае, если его степень равна двум d_c - 1=2, поскольку выход проверочного узла CN есть ЛОП суммы двоичных независимых случайных величин (далее будем обращаться к этому алгоритму как к АРД ЛОП).

Выражение (2) может быть вычислено повторным применением (10). Например, представляя операцию сложения ЛОП как функцию двух аргументов

$$L_{sum}(L_1,L_2)=L(A_2)=\ln\left(\frac{1+\mathrm{e}^{\mathrm{L}_1+\mathrm{L}_2}}{\mathrm{e}^{\mathrm{L}_1}+\mathrm{e}^{\mathrm{L}_2}}\right)=$$

$$=2\,\mathrm{th}^{-1}(\mathrm{th}(L_1/2)\,\mathrm{th}(L_2/2)),$$

сумма ЛОП $L_1\boxplus L_2\boxplus L_3\boxplus L_4$ может быть вычислена как:

$$L_1\boxplus L_2\boxplus L_3\boxplus L_4=L_{sum}(L_1,L_{sum}(L_2,L_{sum}(L_3,L_4))).$$

Таким образом, оператор (4) может быть представлен в виде:

$$\boxplus_{i'\in N(i)-\{j\}} L_{i'\to j}=\underset{\substack{i'\in N(i)-\{j\}-\{1\}\\ L^0_{j\to i}=L_{\{1\}\to j}}}{L_{sum}}(L_{j\to i},L_{i'\to j}),$$

где $L^0_{j\to i}$ - начальное значение $L_{j\to i}$

$L_{j\to i}=\boxplus_{i'\in N(i)-\{j\}} L_{i'\to j}$.

Алгоритмы, основанные на аппроксимации логарифмом Якоби

Рассматриваемые далее алгоритмы основываются на следующей численной аппроксимации выражения (10) при помощи логарифма Якоби [7, 11]

$$\ln(e^{x_1}+e^{x_2})=\max(x_1,x_2)+\ln(1+e^{-|x_1-x_2|}).$$

Применив логарифм Якоби к выражению (10) дважды, получим

$$\mathrm{L}_{\mathrm{sum}}(L_1,L_2)=\ln\left(\frac{1+\mathrm{e}^{\mathrm{L}_1+\mathrm{L}_2}}{\mathrm{e}^{\mathrm{L}_1}+\mathrm{e}^{\mathrm{L}_2}}\right)=$$

$$=\ln(1+\mathrm{e}^{\mathrm{L}_1+\mathrm{L}_2})-\ln(\mathrm{e}^{\mathrm{L}_1}+\mathrm{e}^{\mathrm{L}_2})=$$

$$=\max(0,L_1+L_2)+\ln(1+e^{-|L_1+L_2|})-$$

$$-\max(L_1,L_2)-\ln(1+e^{-|L-L_2|})$$

Можно показать [7], что

$$\max(0,L_1+L_2)+\ln(1+e^{-|L_1+L_2|})-\max(L_1,L_2)=$$

$$=sign(L_1)\cdot sign(L_2)\cdot\min(|L_1|,|L_2|).$$

Тогда окончательно имеем:

$$\begin{aligned}&\mathrm{L}_{\mathrm{sum}}(L_1,L_2)=\\&=sign(L_1)sign(L_2)\min(|L_1|,|L_2|)+g(L_1,L_2)=\\&=sign(L_1)sign(L_2)[\min(|L_1|,|L_2|)+g(|L_1|,|L_2|)]=\\&=sign(L_1)sign(L_2)\min{}^*(|L_1|,|L_2|),\end{aligned}\qquad(11)$$

где

$$g(L_1,L_2)=\ln(1+e^{-|L_1+L_2|})-\ln(1+e^{-|L_1-L_2|}),$$

$$\min{}^*(|L_1|,|L_2|)=$$

$$=\min(|L_1|,|L_2|)+g(|L_1|,|L_2|).$$

При этом $g(L_1,L_2)$ обычно представляют в виде:

$$g(L_1,L_2)=g(L_1')+g(L_2'),$$

где

$$g(L)=\ln(1+e^{-|L|}),$$

$$L_1'=L_1+L_2,$$

$$L_2'=L_1-L_2.$$

Функцию $g(L)$ можно вычислять непосредственно или же использовать один из упрощенных способов [7]. Вот некоторые из них.

Для самой распространенной реализации $g(L)$ – с использованием таблицы, соответствующие значения реализации приведены в табл. 1. Было показано [7], что при использовании таблицы из 8 значений функции $g(L)$, погрешность аппроксимации составляет не более 5 %, что достаточно для приемлемой точности декодирования.

Более точно реализацию функции $g(L)$ можно получить, используя кусочно-линейную аппроксимацию. Уравнения прямых приведены в табл. 2.

Однако можно пойти дальше с точки зрения упрощения алгоритма и воспользоваться аппроксимацией только одной прямой:

$$y(x)=\begin{cases}-|x|\cdot 2^{-2}+0.7, & |x|<2.8,\\ 0, & |x|\geq 2.8.\end{cases}$$

Несмотря на столь грубое приближение к $g(L)$, алгоритм все еще позволяет достичь достаточно хорошей эффективности. На рис. 4 приведены перечисленные варианты аппроксимации функции $g(L)$.

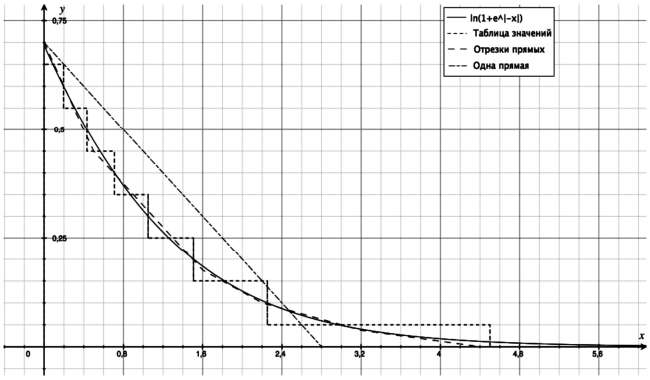

Рис. 4. Различные варианты аппроксимации функции

$$g(L)=\ln(1+e^{-|L|})$$

В [3] представлен другой вариант достаточно грубой аппроксимации, но уже функции $g(L_1,L_2)$ из выражения (11):

$$g(x,y)=$$

$$=\begin{cases}c & |x+y|<2 \quad и \quad |x-y|>2|x+y|,\\ -c & |x-y|<2 \quad и \quad |x+y|>2|x-y|,\\ 0, & в\ противном\ случае,\end{cases}\qquad(12)$$

при этом обычно c = 0.5.

Таблица1. Табличная реализация функции g(x)=log(1+e-|x|).

|x|	g(x)	|x|	g(x)
[0,0.196)	0.65	[1.05, 1.508)	0.25
[0.196,0.433)	0.55	[1.508, 2.252)	0.15
[0.433,0.71)	0.45	[2.252, 4.5)	0.05
[0.71,1.05)	0.35	[4.5, +∞)	0.0

Таблица 2. Кусочно-линейная реализация функции g(x)=log(1+e-|x|).

|x|	g(x)	|x|	g(x)
[0, 0.5)	-|x|*2^-1+0.7	[2.2, 3.2)	-|x|*2^-4+0.2375
[0.5, 1.6)	-|x|*2^-2+0.575	[3.2, 4.4)	-|x|*2^-5+0.1375
[1.6, 2.2)	-|x|*2^-3+0.375	[4.4, +∞)	0.0

Поверхности функций $g(L_1,L_2)$ и ее аппроксимация (12) приведены на рис. 5 и 6, соответственно.

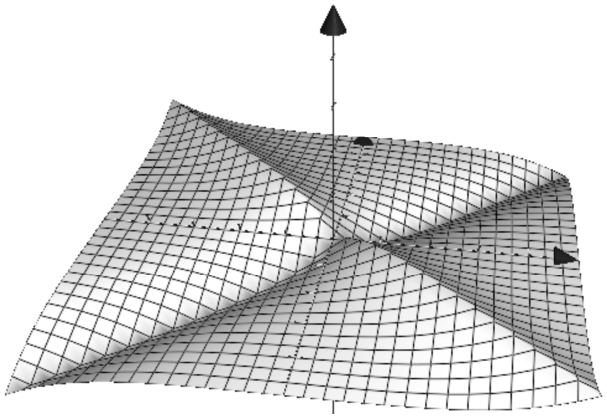

Рис. 5. Поверхность функции

$$g(L_1,L_2)=\ln(1+e^{-|L_1+L_2|})-\ln(1+e^{-|L_1-L_2|})$$

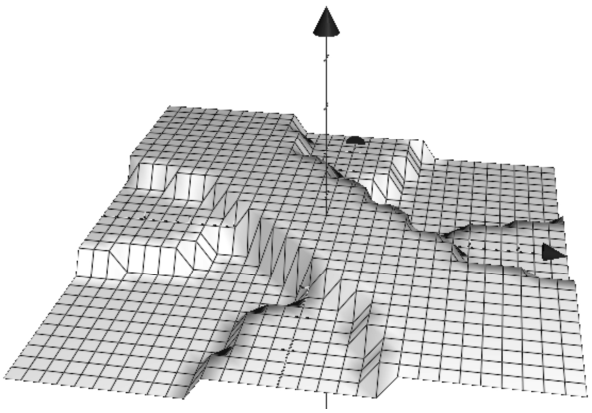

Рис. 6. Поверхность аппроксимации функции

$$g(L_1,L_2)=\ln(1+e^{-|L_1+L_2|})-\ln(1+e^{-|L_1-L_2|})$$

Эффективность работы и скорость сходимости всех рассмотренных алгоритмов приведена на рис. 7 и 8, соответственно. Алгоритмы РД, РД th, РД ЛОП, Якоби идентичны по эффективности работы и скорости сходимости и изображены на рисунках одной кривой, отмеченной как «АРД».

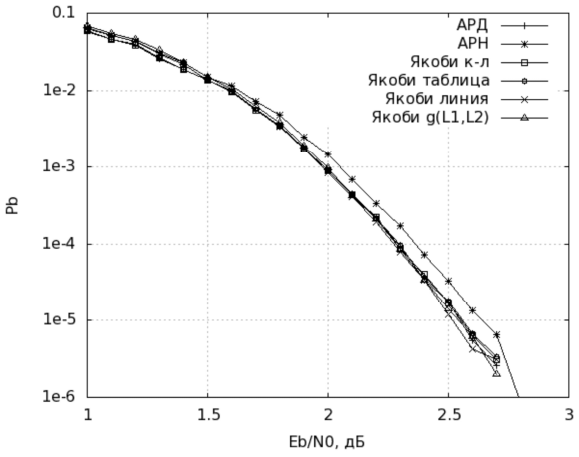

Рис. 7. Эффективность рассмотренных алгоритмов для кода (1008,504) для 100 итераций

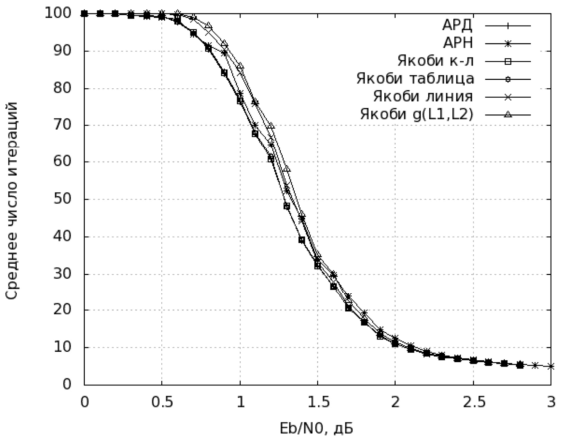

Рис. 8. Среднее число итераций, выполняемое алгоритмом для заданного отношения сигнал/шум

Оценка вычислительных затрат

Поскольку приведенные алгоритмы используют те или иные специальные функции, то, говорить о сложности того или иного алгоритма имеет смысл только для одной аппаратной платформы. Более того, часто бывает, что сложность выполнения операций зависит от режима работы – с фиксированной или с плавающей точкой. Поэтому здесь приведем вычислительные затраты на реализацию каждого из алгоритмов в общем виде, использующем в качестве параметров сложности не только затраты на реализацию специальных функций, используемых тем или иным алгоритмом, но и простейших операций, таких как сложение, умножение, деление.

Для усредненной оценки можно предположить, что все специальные функции реализованы таблицей и имеют одинаковую сложность. Но это достаточно грубое приближение, поскольку на реализацию разных функций могут потребоваться таблицы разных размеров, и более того, следует помнить, что функции th, th^{-1}, $\varphi(x)$ плохо аппроксимируются даже большими таблицами, что приведет к существенной деградации эффективности использующих их алгоритмов.

Вычислительные затраты на реализацию рассмотренных алгоритмов приведены в табл. 3. В случаях, когда число операций выполняемых для подсчета сообщения может меняться в зависимости от входных данных, приводится худший вариант.

Помимо специальных функций в таблице используются:

min – функция получения минимума двух аргументов,

LUT – обращение к таблице аппроксимации логарифма Якоби,

cmp – операция сравнения, выполняемая для определения того, какую аппроксимацию использовать для данных аргументов.

$[x]$ – сложность реализации функции x.

Указанные в таблице затраты можно дополнительно уменьшить, прибегнув к упрощению [3]. Суть его заключается в преобразовании выражения (2) к следующему виду:

Таблица 3. Вычислительные затраты на реализацию рассмотренных алгоритмов

Название	Синдром	Проверочные узлы	Информационные узлы	Жесткие решения
АРД th	M(d_r-1)	d_r M{(dr-1)+(d_r-1)[th]+[arcth]+[*]}	Ndc{1+(dc-1)}	N
АРД	·	dr M{(d_r-1)[φ]+(d_r-1)[+]+[φ]+[*]+ +(d_r-1)} φ=-ln(th(x/2)) φ=ln((ex+1)/(e^x-1))	·	·
АРН	·	d_r M{(d_r-1)[e^x]+[+]+[ln]+[+]}	·	·
АРД ЛОП	·	d_r M{3(d_r-1)+3[e^x](d_r-1)+ +(d_r-1)[/]+(d_r-1)[ln]}	·	·
Якоби	·	d_r M{2(d_r-1)+[min](d_r-1)+ +(d_r-1)[g(L1,L2)]} [g(L_1,L_2)]=2[ln]+5[+]+2[e^x]	·	·
Якоби таблица	·	·[g(L_1,L_2)]=2[LUT]+[+]	·	·
Якоби кусочно-линейный	·	·[g(L_1,L_2)]=2{3[cmp]+[*]+ +[+]}+[+]	·	·
Якоби прямая	·	·[g(L_1,L_2)]=2{[*]+[+]+ +[cmp]}+[+]	·	·
Якоби g(L_1,L_2)	·	·[g(L_1,L_2)]=4[cmp]+2[+]+2[*]	·	·

$$L_{j\to i} = \boxplus_{i'\in M(j)-\{i\}} L_{i'\to j} =$$

$$= \left(\boxplus_{i'\in M(j<i)} L_{i'\to j}\right) \boxplus \left(\boxplus_{i'\in M(j>i)} L_{i'\to j}\right),$$

где $M(j<i)$ – множество позиций единиц в j-ой строке матрицы **H**, с индексами меньшими i, $M(j>i)$ - множество позиций единиц в j-ой строке матрицы **H**, с индексами большими i.

Видно, что преобразование эквивалентное. Однако оно позволяет рассчитать и сохранить в памяти значения первой и второй сумм ЛОП для всех i за один проход по множеству $M(j)$. Первая сумма называется прямой, поскольку рассчитывается с первого элемента множества, а вторая обратной, поскольку рассчитывается с последнего элемента множества в обратном направлении.

Искомое значение ЛОП $L_{j\to i}$ находится как сумма ЛОП значений прямой и обратной сумм ЛОП для i.

Таким образом, для расчета всех сообщений от проверочного узла потребуется 3 прохода для каждой проверки, а не d_r, как в классическом случае. Очевидно, что это упрощение имеет смысл только для кодов с $d_r > 3$.

Также потребуются дополнительная память для хранения значений прямой и обратной сумм ЛОП, и затраты на сохранение и чтение необходимых значений оттуда. Запись осуществляется только один раз – при расчете значений сумм ЛОП, и составляет $|M(j)|$-1 на одну сумму ЛОП, где $|x|$ – мощность множества x. Для регулярного кода $|M(j)|=d_r$. Обращение к каждому элементу сумм ЛОП также осуществляется один раз для каждой суммы и равно $|M(j)|$. Итого $4|M(j)|$-2 для каждой проверки.

Заключение

Из анализа графиков видно, что эффективность всех алгоритмов, кроме АРН, сопоставима и отличается от АРД менее чем на 0,1 дБ. Для моделирования алгоритма АРН использовались значения поправочных коэффициентов C_1 = 0,1, C_2 = 0; его проигрыш АРД составляет порядка 0,1 дБ. Поскольку эффективность рассмотренных алгоритмов очень близка, то при выборе одного из них для реализации декодера низкоплотностного кода, следует руководствоваться, в первую очередь, сложностью реализации каждого из алгоритмов на используемой аппаратной платформе.

Стоит отметить, что для других кодов разница в эффективности может быть более существенной, что может оказаться критичным для выбора алгоритма.

Анализируя таблицу вычислительных затрат, становится очевидным тот факт, что все приведенные модификации алгоритмов нацелены на снижение сложности расчета сообщений от проверочных узлов, что достаточно логично, поскольку это самый сложный из всех шагов АРД.

Также следует отметить, что при сравнении вычислительной сложности алгоритмов необходимо принимать во внимание скорость сходимости рассматриваемых алгоритмов, в простейшем случае – это среднее число итераций при заданном отношении E_b/N_0.

Так видно, что АРД, АРД с аппроксимацией логарифма Якоби таблицей, АРД с кусочно-линейной аппроксимацией логарифма Якоби имеют практически одинаковую скорость сходимости, то есть требуют примерно одинакового числа итераций для декодирования на фиксированном отношении сигнал-шум. А вот оставшиеся алгоритмы несколько проигрывают им. Причем

проигрыш тем больше, чем меньше отношение сигнал-шум.

Исследования выполнены при поддержке гранта для ведущих научных школ НШ-242.2012.10.

Литература

1. Gallager R.G. Low-density parity-check codes // Cambridge, MA: M.I.T. Press, 1963.

2. Berrou C., Glavieux A, Thitimajshima P., «Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes», Proceedings of ICC'93, Geneva, Switzerland, pp. 1064-1070, May, 1993.

3. Ryan W.E., Lin S. Channel codes. Classical and modern // Cambridge, University Press, 2009.

4. Tanner R.M. A recursive approach to low complexity codes // IEEE Trans. Info. Theory, vol. IT-27. №.5. pp. 533-547. September 1981.

5. Кравченко А.Н. Снижение сложности декодирования низкоплотностного кода. «Цифровая обработка сигналов». 2010, № 2, С.35-41.

6. M.Fossorier, M.Mihaljevich, H.Imai, «Reduced complexity iterative decoding of low density parity check codes based on belief propagation», IEEE Trans. on Comm. vol. 47. № 5. pp. 673-680. May 1999.

7. Chen J., Dholakia A., Eleftheriou E., Fossorier M., Hu X.-Y., «Near optimal reduced-complexity decoding algorithms for LDPC codes», in Proc. IEEE International Symposium on Information Theory, Lausanne, Switzerland, July 2002.

8. Лихобабин Е., Дворкович А. Использование квазиоптимальных алгоритмов декодирования LDPC кодов в системе цифрового телевизионного вещания стандарта DVB-T2 // Труды РНТОРЕС имени А.С. Попова. Серия: Цифровая обработка сигналов и ее применения – М. Выпуск XII-1. С. 25-27. 2010.

9. Richardson T.and Novichkov V., «Node processors for use in parity check decoders», United State Patent 6,938,196 B2, August 30, 2005

10. Hegenauer J., Iterative decoding of binary block and Convolutional codes / IEEE Transactions on Information Theory, vol.42, №2, March 1996.

11. Шлома А.М., Бакулин М.Г., Крейнделин В.Б., Шумов А.П. Новые алгоритмы формирования и обработки сигналов в системах подвижной связи / Под редакцией профессора А.М. Шломы. – М.: Горячая линия – Телеком, 2008. – 344 с: ил.

12. http://www.inference.phy.cam.ac.uk/mackay/codes/data.html

REDUCED-COMPLEXITY BP-BASED DECODING ALGORITHMS FOR LDPC CODES

Likhobabin E.A.

Review of various reduced-complexity belief propagation (LLR-BP) based decoding algorithms for LDPC codes are presented. Complexity, effectiveness, and average number of iteration for each algorithm are shown.

Уважаемые авторы!

Редакция научно-технического журнала «Цифровая обработка сигналов» просит Вас соблюдать следующие требования к материалам, направляемым на публикацию:

1) Требования к текстовым материалам и сопроводительным документам:

1. *Текст - текстовый редактор Microsoft Word.*
2. *Таблицы и рисунки должны быть пронумерованы. На все рисунки, таблицы и библиографические данные указываются ссылки в тексте статьи.*
3. *Объем статьи до 12 стр. (шрифт 12). Для заказных обзорных работ объем может быть увеличен до 20 стр.*
4. *Название статьи на русском и английском языках.*
5. *Рукопись статьи сопровождается:*
   - *краткой аннотацией на русском и английском языках;*
   - *номером УДК;*
   - *сведениями об авторах (Ф.И.О., организация, должность, ученая степень, телефоны, электронная почта);*
   - *ключевыми словами;*
   - *актом экспертизы (при наличии в вашей организации экспертной комиссии).*

2) Требования к иллюстрациям:

- *Векторные (схемы, графики) - желательно использование графических редакторов Adobe Illustrator или Corel DRAW.*
- *Растровые (фотографии, рисунки) - М 1:1, разрешение не менее 300dpi, формат tiff, jpg.*

Уважаемые коллеги!

Для тех, кто не успел оформить подписку на второе полугодие 2013 года через ОАО «Роспечать», сохраняется возможность приобретения журналов непосредственно в редакции по адресу: 107031, г. Москва, Рождественка, 6\9\20, стр. 1, Российское научно-техническое общество радиотехники, электроники и связи им. А.С. Попова, или оформить Заказ в соответствии с требованиями, выставленными на сайте журнала: www.dspa.ru.

Справки по телефонам: (495) 621-71-08, 621-06-10.

Факс: (495) 621-06-10. *E-mail*: rntores@mail.ru