СЕКЦИЯ «СИСТЕМЫ И СЕТИ ИНФОКОММУНИКАЦИЙ» СТАТЬИ И ТЕЗИСЫ

УДК 621.391

СЛОЖНЕНИЕ СЛОЖНОСТИ ИТЕРАЦИИ ДЕКОДИРОВАНИЯ LDPC КОДА И ТУРБО КОДА

Абрамов И.О., Барабанов М.Ю., магистрант гр.167001

Белорусский государственный университет информатики и радиоэлектроники

г. Минск, Республика Беларусь

Астровский И.И. – канд. техн. наук

Аннотация. В данной статье проведена оценка вычислительной сложности одной итерации декодирования блокового турбо кода (32,26) x (32,26), декодируемого по алгоритму Чейза, и сравнение полученной вычислительной сложности с вычислительной сложностью итерации декодирования LDPC кода, декодируемого по алгоритму минимума суммы «Min-sum normalized».

Ключевые слова. Турбо код, LDPC код, сверточный код, сравнение сложности итерации декодирования

При создании турбо кодов использовалась парадигма каскадных кодов. Сущность турбо кодирования заключается в каскадировании двух и более составляющих кодов [2]. Соединение может быть как параллельным, так и последовательным, но в обоих случаях оно происходит через перемежитель.

Турбо коды, использующие в качестве составляющих кодов сверточные коды, называются Turbo Convolution Codes (TCC) или сверточными турбо кодами. Декодируют такие коды с помощью алгоритмов Soft Output Viterby Algorithm (SOVA), Log-MAP и его модификации Max-Log-MAP.

В отличие от сверточных турбо кодов, блоковые турбо коды произведения (Turbo Product Codes (TPC)) используют в качестве составляющих кодов линейные блоковые коды БЧХ или Хэмминга. Перемежение в этом случае не требуется, а декодирование таких кодов осуществляется по алгоритму Чейза.

Для сравнения с рассмотренным выше LDPC кодом выбран двумерный блоковый турбо код (32,26) x (32,26) со скоростью кодирования 0.66. В качестве составляющих кодов используется расширенный код Хэмминга (32,26).

Работу итеративного алгоритма декодирования блокового турбо кода можно разделить на несколько этапов, представленных на рисунке 1. Подсчет элементарных операций будет вестись для этапов 1-4, выполняемых итеративно.

Рисунок 1 – Этапы декодирования блокового турбо кода

Сложность этапов 1 и 2 одинакова и в случае рассматриваемого турбо кода сводится к 32-кратному повторению процедуры обработки строки. Расчет сложности ведется для длины списка гипотез при T=4.

Шаг 1. Перед поиском четырех наименее надежных символов в рассматриваемой строке ко всем значениям надежностей символов применяется операция взятия модуля числа. Всего $N_{|a|} = 32$ операции взятия модуля числа.

Затем происходит поиск 4 наименее надежных символов. Вначале посредствам $N_{/} = 31$ сравнения определяется наименее надежный символ. Затем посредствам $N_{/} = 62$ сравнений

определяется второй с конца по надежности символ (31 сравнение надежностей между собой и 31 проверка на то, что найденный символ не является наименьшим по надежности). Для поиска третьего и четвертого символов с конца потребуется $N_/ = 93$ и $N_/ = 124$ операции сравнения, соответственно.

Всего на данном шаге обработки строки используется $N_{|a|} = 32$ операции взятия модуля числа и $N_/ = 310$ операций сравнения.

Шаг 2. На данном шаге создается 16 гипотез. Для их создания изначально нужно преобразовать «мягкие» решения на входе алгоритма обработки строки в «жесткие» решения путем $N_/ = 32$ сравнений надежностей с нулем. Далее создаются 16 гипотез путем $N_\oplus = 64$ прибавлений по модулю 2 определенных значений ко всем возможным комбинациям позиций, содержащих наименее надежные символы.

Всего на данном шаге обработки строки используется $N_/ = 32$ операции сравнения и $N_\oplus = 64$ операции сложения по модулю 2.

Шаг 3. На данном шаге происходит декодирование сформированных гипотез. 31 символ перемножается по правилам матричного умножения на транспонированную проверочную матрицу кода Хэмминга из 5 строк и 31 столбца. Для этого используется $N_x = 155$ умножений и $N_+ = 150$ сложений по модулю 2. Получившийся синдром преобразуется в число в десятичной системе счисления путем $N_x = 5$ умножений и $N_+ = 4$ сложений. По полученному значению происходит поиск номера разряда, в котором произошла ошибка. Для этого будет использовано не более $N_/ = 31$ операции сравнения. По найденному номеру разряда инвертируется жесткий символ путем $N_+ = 1$ сложения по модулю 2. Далее на основе декодированных 31 символов путем $N_\oplus = 30$ сложений по модулю 2 вычисляется последний символ гипотезы. Процедура повторяется для всех 16 гипотез.

Всего на данном шаге обработки строки используется $N_x = 2560$ операций умножения, $N_\oplus = 2896$ операций сложения по модулю 2, $N_+ = 64$ сложения, $N_/ = 496$ сравнений.

Шаг 4. Перед расчетом одной метрики значения элементов гипотезы необходимо заменить по правилу «0» на «1», а «1» на «-1». Для этого используется $N_/ = 32$ сравнения. Для расчета метрики используется $N_+ = 32$ вычитания и $N_+ = 31$ сложение, а также $N_x = 32$ умножения полученного результата на самого себя (возведение в квадрат).

Всего на данном шаге обработки строки используется $N_x = 512$ операций умножения, $N_+ = 1008$ сложений, $N_/ = 512$ сравнений.

Шаг 5. Гипотеза с наименьшим значением метрики находится путем $N_/ = 15$ сравнений метрик гипотез друг с другом.

Всего на данном шаге обработки строки используется $N_/ = 15$ сравнений.

Шаг 6. Гипотеза «конкурент» для i-ого символа находится путем $N_/ = 15$ сравнений метрик гипотез друг с другом $N_/ = 15$ проверок условия $d_i \neq k_i$.

Всего на данном шаге обработки строки используется $N_/ = 960$ операций сравнения для поиска 32 гипотез «конкурентов».

Шаг 7. Оценка сложности ведется для расчета поправок по формуле (1).

$$w_j = ((R' - K_i)^2 - (R' - D)^2) / 4 \times d_i \quad (1)$$

Значения в скобках уже рассчитаны на шаге 4. Для расчета одной поправки используется $N_+ = 1$ вычитание и $N_x = 2$ умножения (операция деления на 4 заменена на умножение на 0.25).

Всего на данном шаге обработки строки используется $N_+ = 32$ вычитания и $N_x = 512$ умножения.

Шаг 8. Для формирования одного элемента «мягкого» входа используется $N_x = 1$ умножение и $N_+ = 1$ сложение.

Всего на данном шаге обработки строки используется $N_+ = 32$ сложения и $N_x = 32$ умножения.

После обработки всех строк и столбцов необходимо получить 64 синдрома, каждый из которых рассчитывается путем перемножения 32 символов на проверочную матрицу кода Хэмминга из 6 строк и 32 столбцов. Всего для расчета синдромов необходимо $N_x = 12288$ умножений и $N_\oplus = 11904$ сложений по модулю 2. Перед расчетом синдрома необходимо получить «жесткие» апостериорные решения посредствам $N_/ = 1024$ сравнений. $N_/ = 1$ сравнение требуется для проверки выхода за максимальное число итераций.

Сравнение сложности итерации декодирования LDPC кода и турбо кода приводится в таблице 1. Моделирование на ПК показало, что проведение итерации LDPC декодирования требует в 3 раза меньше времени, чем проведение итерации блокового турбо декодирования.

Таблица 1 – Сравнение сложности декодирования.

Операция	Количество на итерацию декодирования	
	LDPC («Min-sum normalized»)	Блоковый турбо код (Алгоритм Чейза)
Сложение	37024	72704
Умножение	38706	215040
Сравнение	64159	149825
Взятие модуля	33888	2048
Сложение по модулю 2	4218	201344

Следует подчеркнуть, что процедура декодирования блокового турбо кода может быть оптимизирована с точки зрения вычислений, как это показано в [1]. Однако в случае оптимизированного алгоритма декодирования турбо кода его сложность следует сравнивать со сложностью оптимизированного алгоритма декодирования LDPC.

Анализ соотношений между сложностью сравниваемых кодов на итерацию декодирования и их средним используемым числом итераций показывает, что в конкретном случае выгоднее проделывать больше итераций декодирования LDPC с меньшей сложностью, чем осуществлять меньшее число итераций турбо декодирования, но с большей сложностью.

Список использованных источников:

1. Архипкин, А.В. Разработка алгоритмов кодирования и декодирования для телекоммуникационных систем радиосвязи с ортогональными поднесущими : диссертация на соискание ученой степени кандидата технических наук / А.В. Архипкин. – Москва, 2008.

2. Золотарев, В.В. Обзор исследований и разработок методов помехоустойчивого кодирования / В.В. Золотарев, Г.В. Овечкин. – Москва, 2005.

UDC 621.391

COMPARISON OF THE COMPLEXITY OF THE LDPC CODE AND TURBO CODE DECODING ITERATION

Abramov I.O., Barabanov M.Yu., master students gr. 167001

Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus

Astrovskiy I.I. – PhD in Engineering sciences

Annotation. This article estimates the computational complexity of one iteration of decoding the block turbo code (32,26) x (32,26) decoded by the Chase algorithm, and compares the resulting computational complexity with the computational complexity of the LDPC decoding iteration of the code decoded by the minimum sum algorithm "Min-sum normalized".

Keywords. Turbo code, LDPC code, convolutional code, comparison of the complexity of the decoding iteration