Сравнение алгоритмов декодирования двоичных МПП-кодов с жёстким входом

Игорь Жилин
ИППИ РАН
zhilin@iitp.ru

Павел Рыбин
ИППИ РАН
prybin@iitp.ru

Виктор Зяблов
ИППИ РАН
zyablov@iitp.ru

Аннотация

В работе рассматриваются известные алгоритмы декодирования с "мягким" и "жестким" входом для системы с "жестким" приемом. Предлагается оригинальный метод использования алгоритмов декодирования с "мягким" входом для систем с "жестким" приемом. Представлены результаты моделирование алгоритмов декодирования при передаче кодового слова кода с малой плотностью проверок на четность (МПП-кода Галлагера) по двоичному каналу с аддитивным белым гауссовским шумом (АБГШ). В заключении проводится сравнительный анализ полученных результатов для алгоритмов с "жестким" и "мягким" входом.

1. Введение

В работе [1] Р. Галлагер впервые описал конструкцию кодов с малой плотностью проверок (МПП-кодов Галлагера) и предложил разработанный им итеративный алгоритм декодирования "распространения доверия" (belief propogation). Затем в работе [2] аналогичный алгоритм декодирования был описан на основе фактор-графа, соответствующего графу Таннера [3] МПП-кода Галлагера. Данный алгоритм получил название Sum-Product. Также в работе [2] была предложена несколько упрощенная версия Sum-Product, которая получила название Min-Sum. Описанные алгоритмы декодирования относятся к так называемым алгоритмам декодирования с "мягким" входом.

Данные алгоритмы декодирования обладают хорошими корректирующими свойствами, но предъявляют жесткие требования к аппаратуре приемника: для реализации "мягкого" приема необходимо использовать линейные усилители и высокочастотные АЦП. Для некоторых систем реализация подобных декодеров является либо невозможной, либо нецелесообразной. Поэтому разработка эффективных алгоритмов с "жестким" входом является актуально задачей.

На данный момент существует большое количество разнообразных алгоритмов декодирования с "жестким" входом. В данной работе нас будут интересовать просто реализуемые итеративные алгоритмы декодирования: широко известный мажоритарный алгоритм декодирования и алгоритм декодирования с вводом стираний, предложенный в [4].

В данной работе рассматриваются упомянутые выше алгоритмы декодирования с "мягким" и "жестким" входом для системы с "жестким" приемом. Проводится моделирование следующих алгоритмов декодирования при передаче кодового слова МПП-кода по двоичному каналу с аддитивным белым гауссовским шумом (АБГШ):

- мажоритарного декодирования,
- декодирования с введением стираний,
- Sum-Product,
- Min-Sum.

Производится сравнительный анализ полученных результатов для алгоритмов с "жестким" и "мягким" входом.

Работа организована следующим образом: в § 2 описана конструкция МПП-кода Галлагера, затем в § 3 описаны рассматриваемые алгоритмы декодирования, в § 4 описан разработанный способ применения алгоритмов с "мягким" входом для каналов с "жестким" приемом, в § 5 и § 6 приведены результаты моделирования рассматриваемых алгоритмов декодирования МПП-кода при передаче кодового слова по двоичному каналу с АБГШ, проводится сравнительный анализ полученных результатов и делается вывод по работе.

2. Структура и свойства МПП-кода

Рассмотрим построение проверочной матрицы $\mathbf{H}$ кода с малой плотностью проверок на четность (МПП-кода Галлагера). Проверочную матрицу H_0

кода с проверкой на четность длины n_0 можно записать как $\mathbf{H}_0 = \underbrace{111...1}_{n_0}$. Запишем диагональную блочную матрицу $\mathbf{H}_b$ с b проверочными матрицами $\mathbf{H}_0$ на главной диагонали:

$$\mathbf{H}_b = \begin{pmatrix} \mathbf{H}_0 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathrm{H}_0 & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \underbrace{\mathbf{0} \quad \mathbf{0} \quad \dots \quad \mathbf{H}_0}_{b} \end{pmatrix},$$

где b очень велико. Поскольку размер матрицы $\mathbf{H}_0$ равен $1 \times n_0$, то размер матрицы $\mathbf{H}_b - b \times bn_0$. Обозначим $\pi(\mathbf{H}_b)$ случайную перестановку столбцов матрицы $\mathbf{H}_b$. Тогда матрица, составленная из $\ell > 2$ таких перестановок в качестве слоев,

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \\ \vdots \\ \mathbf{H}_\ell \end{pmatrix} = \begin{pmatrix} \pi_1(\mathbf{H}_b) \\ \pi_2(\mathbf{H}_b) \\ \vdots \\ \pi_\ell(\mathbf{H}_b) \end{pmatrix}$$

является разреженной проверочной матрицей $\mathbf{H}$ размера $\ell b \times bn_0$, которая определяет ансамбль МПП-кодов Галлагера длины $n = bn_0$, где $n \gg n_0$. Обозначим этот ансамбль $\mathscr{E}(n_0, \ell, b)$.

О п р е д е л е н и е 1. Для компонентного кода с проверкой на четность с проверочной матрицей $\mathbf{H}_0$ независимо и равновероятно выбирая случайные перестановки π_l , $l = 1, 2, ..., \ell$, определим ансамбль МПП-кодов Галлагера $\mathscr{E}(n_0, \ell, b)$.

Нижняя оценка на скорость R кода из $\mathscr{E}(n_0, \ell, b)$ определяется формулой:

$$R \geqslant 1 - \ell(1 - R_0),$$

где $R_0 = \frac{n_0 - 1}{n_0}$ – скорость кода с проверкой на четность. Равенство достигается только в случае, когда $\mathbf{H}$ имеет полный ранг.

Как следует из построения, МПП-код Галлагера из $\mathscr{E}(n_0, \ell, b)$ имеет $n = bn_0$ кодовых символов, которые распределены между ℓb компонентных кодов (b в каждом слое) с проверочной матрицей $\mathbf{H}_0$. Такие коды могут быть представлены с помощью двудольного графа Таннера [3] $G = (V_1 : V_2, E)$ с $n = bn_0$ вершинами-символами V_1 и ℓb вершинами-кодами V_2, как на рис. 1. Если символ входит в проверку кода-компонента, то в графе Таннера существует ребро из E, соединяющее соответствующую вершину-символ из V_1 с соответствующей вершиной-кодом из V_2. В соответствии с конструкцией проверочной матрицы МПП-кода Галлагера каждая вершина-код включает одно проверочное уравнение, представляющее собой сумму по модулю два n_0 входящих в данную

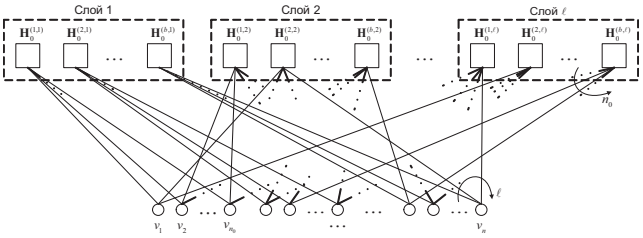

Рис. 1. Двудольный граф Таннера соответствующий проверочной матрице H МПП-кода Галлагера

проверку символов. Каждый кодовый символ входит в проверочное уравнение точно одного кода-компонента в каждом слое. Таким образом, соответствующий граф Таннера регулярен со степенью вершины-символа равной ℓ и степенью вершины-кода равной n_0.

3. Алгоритмы декодирования

В данной работе исследуются алгоритмы декодирования только двоичных МПП-кодов.

Алгоритмы декодирования можно разделить на алгоритмы с жёстким входом и алгоритмы с мягким входом.

Первый класс алгоритмов, известный также как вероятностное декодирование (probabilistic decoding) – это алгоритмы, которые в качестве входа получают оценку вероятностного распределения символов, полученную из канала, и работают с численными значениями вероятностей. Этот класс включает в себя такие алгоритмы, как sum-product (belief propagation) и его упрощённый вариант min-sum.

Второй класс включает в себя такие методы декодирования, работающие непосредственно со значениями символов. Это такие алгоритмы, как мажоритарное декодирование и алгоритм декодирования с введением стираний.

Хочется подчеркнуть отличия алгоритмов декодирования с мягким и с жёстким входом. Первые имеют теоретически лучшую корректирующую способность, которая в пределе хороших соотношения сигнал/шум составляет 3 дБ, а в пределе плохих – не отличается от алгоритмов с жёстким входом. В то же время в реальных условиях может отсутствовать информация о вероятностях, и алгоритм с мягким входом будет неприменим непосредственно.

В этом контексте отдельный интерес представляет возможность адаптации алгоритмов с мягким входом для каналов с жёстким выходом.

3.1. Общие черты декодеров

Все рассматриваемые декодеры МПП-кодов работают с представлением кода в виде фактор-графа, так же известного как граф Таннера.

Рассматриваемые алгоритмы являются итерационными. Каждая итерация состоит из последовательной обработки сначала данных вершин-проверок, а затем вершин-переменных.

Остановка итеративной части алгоритма всегда осуществляется, если все проверки оказались выполнены или по достижении максимального числа итераций. Возможны так же дополнительные критерии остановки.

3.2. Мажоритарное декодирование

Мажоритарное декодирование является простейшим методом декодирования. Алгоритм имеет жёсткий вход и выполняет следующие итерации:

1. Вычисление проверок. Результатом являются данные о том, выполняется ли данная проверка, хранящаяся в данной вершине.
2. Изменение символов, в которых не выполняется более половины проверок, на противоположные.

Кроме основных условий останова, возможны дополнительные. Алгоритм может завершать свою работу, если:

- не изменился вес синдрома,
- не изменился синдром,
- не изменился вектор символов.

Данный алгоритм лежит в основе алгоритмов, работающих с жёстким представлением данных.

Модификацией данного алгоритма является вариант, когда на втором шаге итерации производится инвертирование символов, имеющих не не более половины невыполненных проверок, а максимальное среди всех.

Алгоритм отличается вычислительной простотой: в нём выполняются практически только логические операции.

3.3. Декодер с введением стираний

Усовершенствованием алгоритма мажоритарного декодирования является алгоритм с введением стираний. В данном алгоритме вместо того, чтобы инвертировать оцененный как ошибочный символ, производится сначала его пометка как “стёртого”, а затем повторное вычисление проверок с учётом стёртых символов и восстановление символов по вычисленным проверкам. Таким образом алгоритм на каждой итерации выполняет следующие действия:

1. Вычисление проверок. Результатом являются данные о том, выполняется ли данная проверка, хранящаяся в данной вершине.
2. Пометка символов, оцененных как ошибочные, стёртыми. Как ошибочные можно помечать, например, символы, в которых не выполняется более половины проверок, или символы, в которых не выполняется максимальное число проверок.
3. Вычисление проверок с учётом стёртых символов. В случае, если в проверке стёрт единственный символ, то её значение может быть использовано только в этом символе.
4. Мажоритарное вычисление стёртых символов по проверкам: выбирается то значение, которое является предпочтительным исходя из значений символов. В случае, если по итогам вычисления символов нет предпочтительного значения, то восстанавливается предыдущее.

Данный алгоритм так же отличается простотой и является лишь несколько сложнее алгоритма мажоритарного декодирования.

3.4. Sum-Product

Sum-Product, известный так же под названиями Belief Propagation, алгоритм с распространением доверия – алгоритм декодирования, являющийся алгоритмом обмена сообщениями на фактор-графе. Алгоритм получает на вход вероятности приёма символов, которые в практической реализации для двоичного случая передаются как отношения правдоподобия. Далее алгоритм итеративно выполняет вычисления на вершинах-символах и вершинах-проверках, передавая между ними сообщения, представляющие из себя по сути отношения правдоподобия данного символа (проверки), вычисленные для соответствующей проверки (символа) на основе всех остальных, кроме того, для кого производится вычисление. Таким образом, алгоритм представляет из себя циклическое выполнение итераций, содержащих два шага:

1. Вычисление сообщений от проверок к символам. В логарифмах отношения правдоподобия формулы выглядят следующим образом:

$$\gamma_{mn} = \prod_{n' \in N(m) \backslash n} \alpha_{mn'} \cdot f\Big(\sum_{n' \in N(m) \backslash n} f(\beta_{mn'}) \Big)$$

   где γ_{mn} – сообщение от m-й проверки n-му символу, α_{mn} – знак сообщения от n-го символа для m-й проверки, β_{il} – модуль (абсолютное значение) сообщения от n-го символе для m-й проверки, $N(m)$ – множество символов в m-й проверке, $N(m) \backslash n$ – множество символов в m-й проверке

кроме n и функция $f(x)$:

$$f(x) = \ln \frac{e^x + 1}{e^x - 1}$$

Выражение $f(\sum f(\beta))$ по сути является вычислением мягкого минимума среди всех входящих в сумму значений β.

2. Вычисление сообщений от символов проверкам. В логарифмах отношения правдоподобия формулы выглядят следующим образом:

$$\alpha_{n'}\beta_{n'} = \alpha_n\beta_n + \sum_{m \in M(n)} \gamma_{mn}$$

$$\alpha_{mn}\beta_{mn} = \alpha_n\beta_n + \sum_{m' \in M(n)\backslash m} \gamma_{m'n}$$

где $\alpha_{n'}$ и $\beta_{n'}$ – оценка знака и модуля n-го символа, γ_{mn} – сообщение от m-й проверки n-му символу, α_{mn} – знак сообщения от n-го символа для m-й проверки, β_{mn} – модуль (абсолютное значение) сообщения от n-го символе для m-й проверки, $M(n)$ – множество проверок, в которые входит n-й символ, $M(n)\backslash m$ – множество проверок, в которые входит n-й символ кроме m.

Следует отметить, что в пределе, когда Belief Propagation выполняется на фактор-графе без циклов, он сходится к оценке максимального правдоподобия. В то же время при разумных длинах кода (от сотен до сотен тысяч символов) это условие может быть выполнено только для небольшого числа итераций порядка 2–4, а далее начинают сообщения становятся скореллированными и несут смысл не вероятностей, а оценок надёжности символов. В то же время результативным является выполнение существенно большего числа итераций, порядка десятков.

Belief Propagation является алгоритмом, который минимизирует вероятность ошибки на бит. При этом вероятность ошибки кодового слова получается выше, чем у алгоритмов, минимизирующих вероятность ошибки на кодовое слово.

Sum-Product имеет высокую вычислительную сложность: на каждой итерации происходит вычисление сообщений, происходящее с целыми числами, а так же вычисление нетривиального вида функции.

3.5. Min-Sum

Алгоритм Min-Sum (сумма наименьших) является упрощением алгоритма Sum-Product, обладающим существенно меньшей вычислительной сложностью. Каждая итерация содержит два шага:

1. Вычисление сообщений от проверок к символам. В логарифмах отношения правдоподобия формулы выглядят следующим образом:

$$\gamma_{mn} = \prod_{n' \in N(m)\backslash n} \alpha_{mn'} \cdot \min_{n' \in N(m)\backslash n} \beta_{mn'}$$

где γ_{mn} – сообщение от m-й проверки n-му символу, α_{mn} – знак сообщения от n-го символа для m-й проверки, β_{il} – модуль (абсолютное значение) сообщения от n-го символе для m-й проверки, $N(m)$ – множество символов в m-й проверке, $N(m)\backslash n$ – множество символов в m-й проверке кроме n.

2. Вычисление сообщений от символов проверкам. В логарифмах отношения правдоподобия формулы выглядят следующим образом:

$$\alpha_{n'}\beta_{n'} = \alpha_n\beta_n + \sum_{m \in M(n)} \gamma_{mn}$$

$$\alpha_{mn}\beta_{mn} = \alpha_n\beta_n + \sum_{m' \in M(n)\backslash m} \gamma_{m'n}$$

где $\alpha_{n'}$ и $\beta_{n'}$ – оценка знака и модуля n-го символа, γ_{mn} – сообщение от m-й проверки n-му символу, α_{mn} – знак сообщения от n-го символа для m-й проверки, β_{mn} – модуль (абсолютное значение) сообщения от n-го символе для m-й проверки, $M(n)$ – множество проверок, в которые входит n-й символ, $M(n)\backslash m$ – множество проверок, в которые входит n-й символ кроме m.

Min-Sum имеет меньшую вычислительную сложность по сравнению с Sum-Product благодаря отсутствию необходимости вычислять нетривиальные функции, однако остаётся существенно сложнее алгоритмов мажоритарного декодирования и алгоритма с введением стираний из-за необходимости производить арифметические операции.

4. Применение декодеров с мягким входом для каналов с жёстким выходом

Особенный интерес представляло использование алгоритмов с мягким входом совместно с каналом с жёстким выходом. В то время как обратное использование очевидно – декодеру с жёстким решением можно просто подать более вероятные значения входных символов канала с мягким выходом – обратное преобразование требует замены имеющихся на входе символов после жёсткого решения на некоторые оценки надёжности символов.

В целом, вопрос того, какой способ преобразования жёсткого выхода канала в мягких вход декодера является оптимальным и в чём будут проявляться отличия между ними является предметом отдельного исследования и оставлен за рамками данной статьи. Здесь же приведём основные использованные идеи.

Для начала сделаем некоторое отступление о разнице в работе итеративных алгоритмов с жёстким и с мягким представлением внутренней информации. Очевидно, что в итеративных алгоритмах на

каждой итерации может быть посчитано значение числа ошибок. Эксперименты показывают, что при использовании декодера Sum-Product с мягким выходом из канала (в штатном режиме) число ошибок обычно монотонно падает от итерации к итерации. Интересной особенностью алгоритмов Sum-Product и Min-Sum является то, что кроме дискретного числа ошибок можно посчитать аналогичную непрерывную величину – сумму вероятностей ошибок по всем символам. Для этого нужно для каждого символа вычислить оценки надёжности в виде вероятностей p_i того, что символ нулевой, после чего для символов, которые при передаче были единичными посчитать величину $1 - p_i$ и просуммировать. В случае, если выполняется декодирование нулевого кодового слова это будет просто сумма оценок надёжности p_i. В контексте алгоритма Sum-Product данная величина интересна тем, что позволяет получить более глубокое понимание происходящих процессов – она усредняет именно те оценки, с которыми работает алгоритм. Рассмотрение её поведения показывает, что в типичной ситуации (рис. 2) успешного декодирования она практически совпадает с числом ошибок и при работе алгоритма от итерации к итерации меняется похожим образом. В случае, если на вход декодера подаётся мягкий выход канала, вероятности которого изменены в сторону увеличения или уменьшения надёжности, оценки суммы вероятностей ошибок соответственно изменяются, увеличивается вероятность отказа от декодирования, а при отсутствии отказа при занижении надёжностей время декодирования увеличивается (растёт число итераций), при завышении уменьшается.

Отсюда можно сделать вывод, что оптимальным может являться такой вход, который обеспечивает равенство суммы вероятностей ошибок, используемой декодером, реальному числу ошибок в кодовом слове.

Вариантом такого входа для двоичного кода является замена входных символов в виде 0 и 1 на вероятности p и $1-p$, где p – оценка вероятности ошибки в данном канале. Логарифмы отношения правдоподобия при этом выглядят как π и $-\pi$, где $\pi = \ln \frac{1-p}{p}$.

При использовании этого метода сумма вероятностей ошибок ведёт себя аналогично случаю работы с мягким выходом канала (рис. 3): практически совпадает с числом ошибок, от итерации к итерации меняется подобно ей. При этом при занижении оценки надёжности алгоритм сходится более медленно и наоборот.

Хочется отметить, что в данном случает с непосредственно жёсткими решениями алгоритм работает только на первой итерации – далее символам присваиваются оценки надёжности и алгоритм продолжает работу уже с ними. В отличие от алгоритмов с жёстким внутренним представлением оценок символов это позволяет сохранить и использовать существенную часть данных и тем самым улучшить эффективность работы.

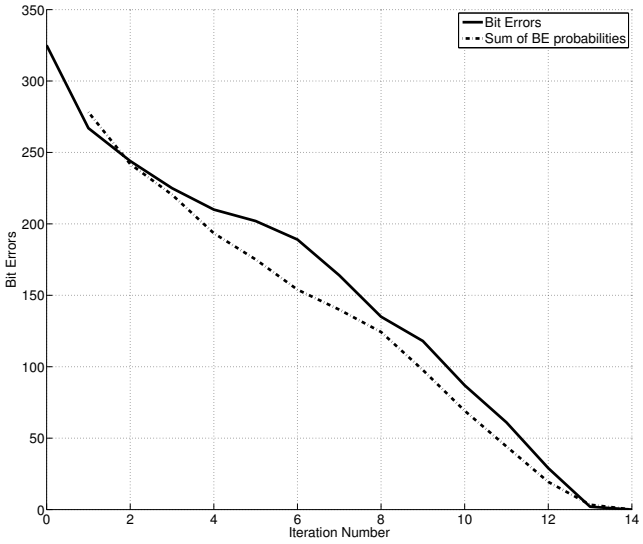

Рис. 2. Пример зависимости числа ошибок (синие звёздочки) и суммарной вероятности ошибки (зелёные круги) от номера итерации Sum-Product при соотношении сигнал/шум -0.7 дБ

Способы оценки вероятности ошибки p так же могут быть различны. В данной работе было проведено сравнение трёх способов. Первые два – это вычисление эмпирической вероятности как отношения числа ошибок в кодовом слове к его длине, а так же расчёт коэффициента ошибок канала через известное соотношение сигнал/шум. Третьим способом было использование фиксированной константы, соответствующую работе при наихудшем соотношении сигнал/шум, при котором декодирование обычно происходит успешно.

5. Результаты моделирования

Моделирование работы декодеров производилось методами иммитационного моделирования с использованием среды MatLab.

Для моделирования декодеров была написана их реализация на языке Си в виде функции для MatLab. Для сравнения декодеров использовался двоичный МПП-код длины 3280 и скорости 1/2. В качестве канала использовался канал с двоичной фазовой модуляцией с аддитивным белым гауссовским шумом.

Было произведено моделирование нескольких режимов работы декодеров.

Сравнение различных способов преобразования жёсткого выхода канала в мягкий вход декодера (непосредственное вычисление числа ошибок, рас-

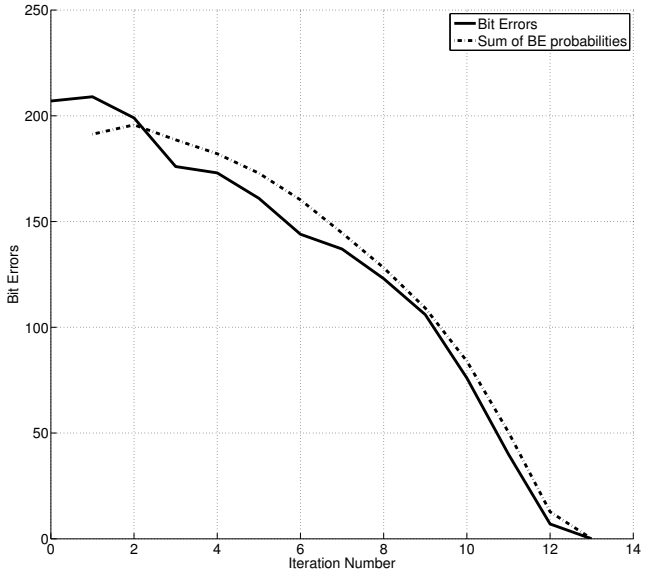

Рис. 3. Пример зависимости числа ошибок (синие звёздочки) и суммы вероятностей ошибок (зелёные круги) от номера итерации Sum-Product при подаче на вход оценок вида $\pm\pi = \ln\frac{1-p}{p}$ при соотношении сигнал/шум +0.6 дБ

чёт на основе соотношения сигнал/шум, а так же использование константы) не показало статистически значимого отличия для Sum-Product.

Результаты работы алгоритмов Sum-Product и Min-Sum приведены для случая подачи на него значений с фиксированной константой.

Декодеры с жёсткой логикой сравнивались в вариантах, когда символы считались ненадёжными в случае невыполнения наибольшего числа проверок среди всех символов, результаты приведены как для мажоритарного декодера, так и для декодера с введением стираний.

График зависимости числа ошибок от соотношения сигнал/шум показан на рисунке 4. Общее число экспериментов составило $2 \cdot 10^4$ кодовых слов или $6.56 \cdot 10^7$ бит. Результаты сравнивались по уровню вероятности ошибки, равному 10^{-5}.

Результаты показали выигрыш порядка 1.1 дБ при применении алгоритма Sum-Product по сравнению с алгоритмами, работающими с жёстким представлением символов (соотношение сигнал/шум 1.2 дБ против 2.3 дБ). Так же исходя из графика видно, что алгоритм Min-Sum оказался практически неприменим для канала с жёстким выходом, показывая результаты на 0.3 дБ хуже, чем у мажоритарного декодирования.

6. Заключение

Результаты моделирования показали высокую эффективность декодера Sum-Product при применении его в сочетании с каналом с жёстким выходом. Сравнение различных вариантов оценки надёжности символов показало работоспособность каждого метода. Отдельный интерес представляет из себя более детальное изучение эффектов от их выбора, а так же способы оптимизации константы в простейшем методе.

Список литературы

[1] R. G. Gallager, *Low-Density Parity-Check Codes*, MIT Press, 1963.

[2] Frank R. Kschischang, Brendan J. Frey, Hans-andrea Loeliger, *Factor Graphs and the Sum-Product Algorithm*, IEEE Transactions on Information Theory, 1998, vol. 47, pp. 498–519

[3] M. Tanner, *A Recursive Approach to Low Complexity Codes*, IEEE Trans. Inform. Theory, 1981 vol. 27, no. 5, pp. 533–547.

[4] Зяблов В. В., Рыбин П. С., Фролов А. А., *Алгоритм декодирования с вводом стираний для МПП-кодов, построенных над полем GF(q)*, Информационно-управляющие системы, 2011, № 1, С. 62–68.

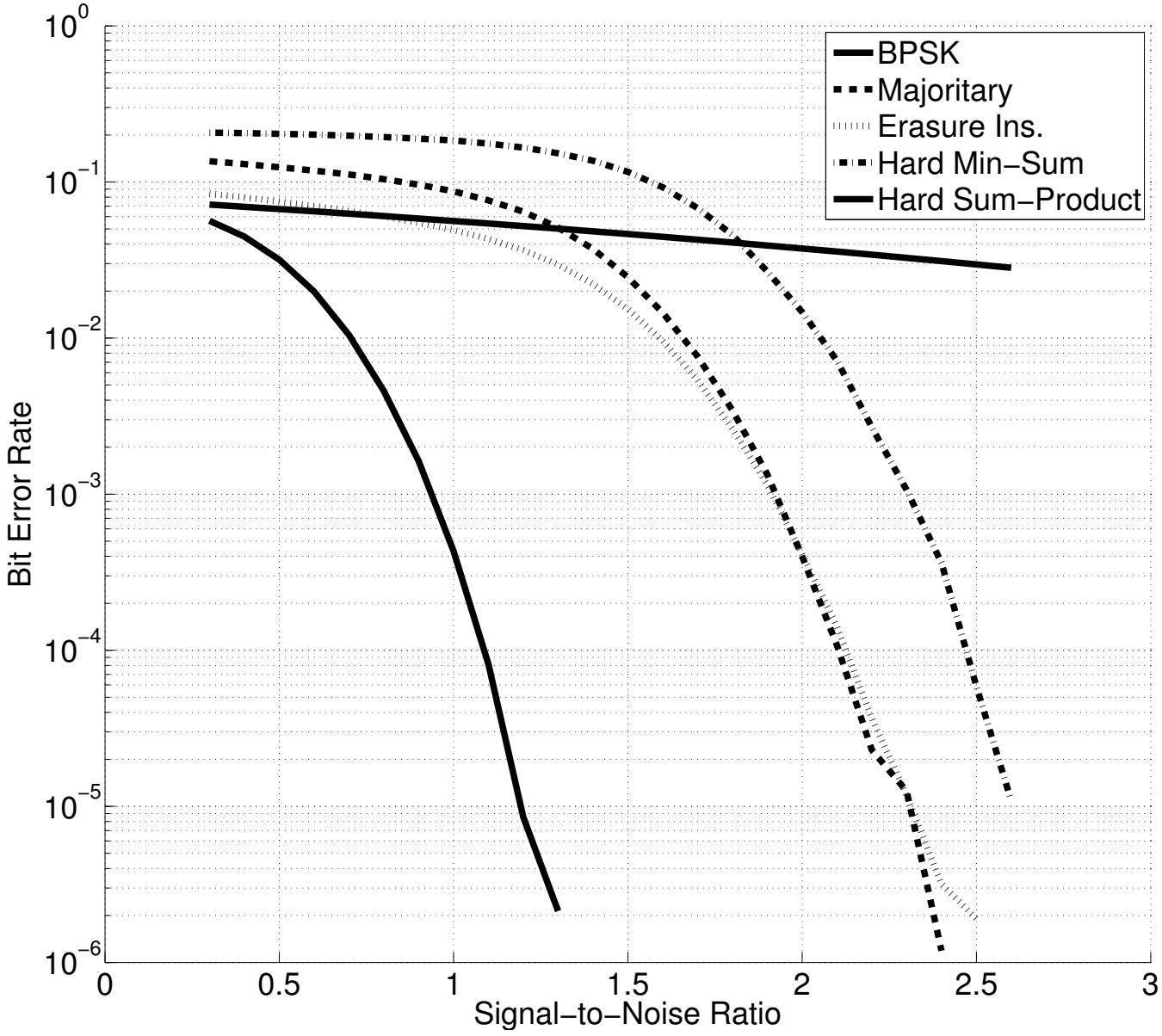

Рис. 4. График зависимости числа ошибок от соотношения сигнал/шум для различных алгоритмов