УДК 519.72

І.О. Завадський, к. ф.-м. н., доцент

Комбінаторний алгоритм підвищення завадостійкості кодів, що генеруються скінченними автоматами

Київський національний університет імені Тараса Шевченка, 03680, м. Київ, пр-т. Глушкова 4д, e-mail: zava@ukr.net

Igor O. Zavadskyi, Associate Professor, Ph. D.

Search algorithm for increase the performance of codes generated by finite automatons

Taras Shevchenko National University of Kyiv, 03680, Kyiv, Glushkova av., 4d

Описано комбінаторний метод пошуку помилок у кодових словах, що дає змогу суттєво підвищити завадостійкість кодування на основі скінченних автоматів, зокрема згорткових кодів. При цьому часова складність покращуваного базового методу декодування збільшується несуттєво.

Ключові слова: завадостійке кодування, скінченний автомат, завадостійкість, часова складність.

The combinatorial algorithm searching error patterns that improves the performance of forward error correcting codes exploiting the finite automatons as core encoding and decoding unit is introduced. As convolutional codes can be represented in the form of finite automaton diagram, the algorithm discussed can improve convolutional code performance too. Algorithm analyzes the soft output of decoder based on modification of Viterbi algorithm with linear metric. It allows to achieve 0,7 dB gain at BER $1{,}7\cdot10^{-5}$ comparing to conventional convolutional NASA(171,133) code, while the increase of time complexity of decoding is negligibly small. Code performance is simulated on BPSK channel. The impact of different custom parameters on algorithm performance is investigated, i.e. block length, checksum length or number of possible error positions. Only the codes of rate ½ are discussed, although the results could be easily generalized to other error correcting code types.

Key words: error correcting codes, finite automaton, noise immunity, time complexity.

Статтю представив д.ф.-м.н., проф. Анісімов А.В.

У [1] розглянуто метод вдосконалення одного з найпоширеніших різновидів завадостійких кодів — згорткових кодів. Вони розглядаються як частинний випадок більш широкого класу кодів, що генеруються скінченними автоматами. Кожної ітерації такий автомат зчитує t бітів із вхідної послідовності, залежно від зчитаного значення здійснює перехід в один із 2^t станів, записуючи при цьому p бітів у кодове слово. Показано, що хоча структура діаграм стану автоматів, які відповідають згортковим кодам, у певному розумінні є оптимальною, оптимальним не є маркування, тобто зіставлення дугам на діаграмі станів символів, що записуються автоматом у код на виході. Оптимізація способу маркування дала можливість підвищити завадостійкість коду приблизно вдвічі для деяких співвідношень рівня корисного сигналу та шуму в каналі зв'язку.

Для подальшого підвищення завадостійкості коду слід застосовувати інші підходи, наприклад комбінаторні методи пошуку помилок. Ми опишемо один такий метод, що покращує ефективність коду значно суттєвіше за наведений в [1] метод маркування дуг діаграми станів автомату, лише незначно підвищуючи при цьому часову складність декодування. Розглядатимемо м'яке рішення в каналі зв'язку з двійковою фазовою модуляцією [2]: біти вхідного повідомлення кодуються амплітудами 1 (одиничний біт) і –1 (нульовий біт), на які накладається адитивний білий гаусів шум, і на вході декодера отримуємо дійсні числа, що визначають вірогідність хибної інтерпретації того чи іншого сигналу як нульового чи одиничного біта.

Метод дає змогу покращити результати застосування розглянутої в [1] модифікації добре відомого алгоритму декодування Вітербі.

Нагадаємо, що алгоритм Вітербі виконується у два проходи: прямий та зворотний. Якщо розглядати коди швидкості ½ з такою структурою діаграми станів автомату, як описано в [1], то в кожен стан існуватимуть переходи з двох інших станів і під час прямого проходу алгоритму Вітербі кожному зі станів j, у яких кодувальний автомат може перебувати після i-го переходу, приписується метрика

$$m_i^j = \max\left(m_{i-1}^q + \delta(s_{qj}, r_i), m_{i-1}^t + \delta(s_{tj}, r_i)\right) \quad (1)$$

Тут q і t — стани, з яких є переходи в стан j, s_{qj} і s_{tj} — двохсимвольні рядки, які записує кодувальний автомат у вихідний код на переході з q-го та t-го станів в j-й, r_i — набір із двох дійсних чисел — сигналів на вході декодера, що відповідають i-му переходу, $\delta(s_{qj}, r_i) = -2 + \sum_{k=0}^{1}(-1)^{s_{qj}^k+1} r_i^k$, а $\delta(s_{tj}, r_i)$ обчислюється аналогічно. Зауважимо, що у наведеній сумі величина r_i^k береться зі знаком «–», якщо k-й символ у рядку s_{qj} дорівнює «0» та зі знаком «+», якщо він дорівнює «1». Таким чином, величини $\delta(s_{qj}, r_i)$ та $\delta(s_{tj}, r_i)$ будуть тим більшими, чим менше дійсні числа r_i^k відхиляються від значень амплітуд, що відповідають символам, генерованим на переході з q-го чи t-го стану в j-й. Отже, на прямому проході алгоритму Вітербі припускаємо, що на i-й ітерації у кожен стан j міг відбутися перехід зі стану q, якщо $\delta(s_{qj}, r_i) > \delta(s_{tj}, r_i)$ та зі стану t, якщо $\delta(s_{qj}, r_i) < \delta(s_{tj}, r_i)$. На зворотному проході вибираємо стан, якому на останній ітерації відповідає максимальна метрика, та відновлюємо переходи автомата у зворотному порядку згідно із зазначеними вище припущеннями. Під час декодування, як і під час кодування, автомат починає роботу в нульовому стані, тому покладемо $m_0^0 = 0$.

Щоб виправити помилки, які можуть виникнути після декодування за описаним вище методом, перед кодуванням послідовності довжини L дописуватимемо до неї справа контрольну суму довжини t, i-й біт якої дорівнює сумі за модулем 2 усіх бітів, номери яких дорівнюють $i(\mathrm{mod}\ t)$. Після декодування знову визначатимемо за першими L бітами контрольну суму і, якщо вона збігається з останніми t бітами, вважатимемо, що декодування виконано успішно. Інакше шукатимемо переходи, які на зворотному проході методу Вітербі могли бути визначені помилково. А саме, припускатимемо, помилки могли виникнути насамперед у тих переходах, де величина (2) є найменшою. Ця величина являє собою різницю між елементами, серед яких у формулі (1) визначається максимальний.

$$\left|m_{i-1}^q + \delta(s_{qj}, r_i) - m_{i-1}^t - \delta(s_{tj}, r_i)\right| \quad (2)$$

Тоді ми змінюємо цей перехід, як показано на рис. 1: якщо за попереднім припущенням у вершину j перехід здійснювався з вершини q, то тепер вважатимемо, що він здійснюється з вершини t, і навпаки. Після цього ми застосовуємо зворотний прохід алгоритму Вітербі повторно, знову обчислюємо контрольну суму за першими L бітами і, якщо вона не дорівнює останнім t бітам, припустимо, що помилковим був перехід із другою найменшою різницею (2) або обидва переходи із найменшою та другою найменшою різницею (2) і т.д.

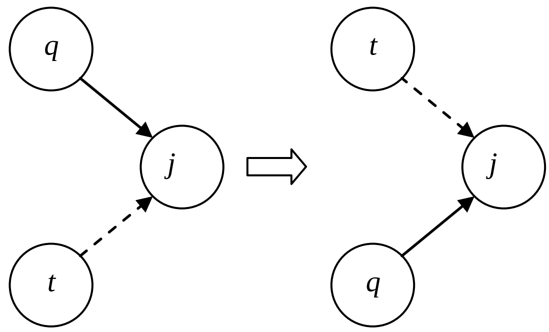

Рис. 1. Змінення переходів, у яких припускається наявність помилки

Унаслідок додавання t бітів довжина кодового слова збільшується. Тому, щоб швидкість коду залишалася незмінною, застосуємо так звану «редукцію переходів». Для кодів швидкості ½ ця техніка буде такою. Якщо до вхідної послідовності довжиною L бітів додано t бітів, під час кодування отриманих $L+t$ бітів $2t$ переходів редукуються: замість двох бітів на кожному з них у вихідний код записується один біт і, таким чином, довжина кодового слова залишається тією самою, тобто $2L$.

Алгоритм декодування детально опишемо нижче. На його вхід подається $2L$-бітове кодове слово.

1. Виконуємо прямий та зворотний проходи алгоритму Вітербі. Результатом

буде послідовність бітів довжини $L+t$, а також отримана на зворотному проході послідовність станів такої само довжини.

2. Обчислюємо t-бітову контрольну суму за першими L бітами. Якщо вона дорівнює останнім t бітам, припускаємо, що вхідне повідомлення успішно декодовано. Інакше переходимо до кроку 3.
3. Нехай $i_1,\ldots,i_n$ — номери переходів із n найменшими значеннями різниць метрик (2), де n — це параметр, значення якого можна добирати.
4. Для будь-якого j, $1 \le j \le n$, припускаємо, що перехід i_j помилковий і змінюємо його, як показано на рис. 1. Це означає, що якщо на зворотному проході на цій ітерації було вибрано стан s, в нього є переходи зі станів q і t і метрика переходу зі стану q вища, цій метриці присвоюємо число, яке значно менше 0, наприклад –1000, і навпаки. Після цього знову виконуємо зворотний прохід алгоритму Вітербі та обчислюємо t-бітову контрольну суму за значеннями перших L бітів. Якщо вона збігається з останніми t бітами, припускаємо, що вхідне повідомлення декодовано успішно. Інакше зміненій метриці надаємо початкового значення і переходимо до наступного значення j.
5. Повторюємо попередній крок за припущення, що є 2 помилкових переходи із номерами з набору $i_1,\ldots,i_n$ і при цьому значення різниці метрик (2) для одного з цих переходів належить до k найменших значень різниць метрик, де $k < n$ — параметр, значення якого можна добирати.
6. Якщо на кроках 4 і 5 успішний результат не отримано, припускаємо, що результатом декодування є $L+t$-бітова послідовність, обчислена на першому кроці.

Зауважимо, що основна обчислювальна складність декодера Вітербі припадає на прямий прохід. Його часова складність становить $O(Lm)$, де L — довжина бітової послідовності, а m — кількість станів кодувального/декодувального автомату. Водночас часова складність зворотного проходу становить лише $O(L)$. Оскільки ми повторюємо лише зворотний прохід, часова складність підвищується незначно, причому чим більшим буде відношення E_b/N_0 рівня корисного сигналу до рівня шуму в каналі зв'язку, тим меншу кількість разів у середньому потрібно повторювати зворотний прохід.

Порівняння удосконаленого згорткового коду, автомат якого має 128 станів, із найкращим з відомих звичайних згорткових кодів із таким само розміром автомату наведено в табл. 1 і на рис. 2. Моделювався канал зв'язку із BPSK-модуляцією та неквантованим м'яким рішенням для декодера. Як базовий кодувальний/декодувальний пристрій було обрано автомат, описаний в [1], результати роботи якого покращувалися завдяки застосуванню описаного в цій статті алгоритму. Його настроюваним параметрам було надано такі значення:

- Довжина вхідної бітової послідовності L=600.
- Довжина контрольної суми t=8.
- Кількість найменших різниць метрик для змінення переходів на кроці 4 алгоритму n=30.
- Кількість найменших різниць метрик для змінення двох переходів на кроці 5 алгоритму k=12.

Як видно, зі збільшенням відношення E_b/N_0 виграш від покращень згорткового коду також зростає, а різниця в середній часовій складності зменшується. Так, рівень бітових помилок у декодованих словах $1{,}7\cdot10^{-5}$ досягається у покращеному методі за рівня шуму в каналі на 0,7Дб вищого, ніж у звичайному згортковому коді NASA, в той час як часова складність декодування є приблизно тією самою, що видно з рис. 3.

Табл. 1. Порівняння завадостійкості згорткового коду NASA та покращеного коду

E_b/N_0, Дб	BER, NASA (171,133)	BER, покращений код
–2	$4{,}8\cdot10^{-2}$	$4{,}5\cdot10^{-2}$
–1	$6{,}7\cdot10^{-3}$	$2{,}4\cdot10^{-3}$
0	$4{,}7\cdot10^{-4}$	$6{,}8\cdot10^{-5}$
1	$1{,}7\cdot10^{-5}$	$1\cdot10^{-6}$

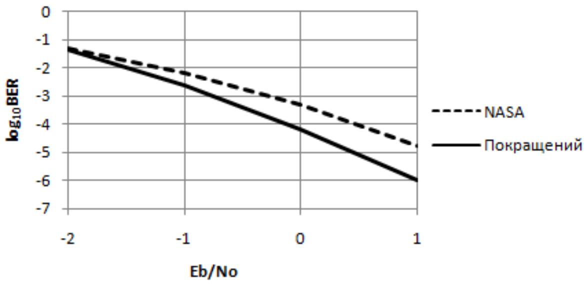

Рис. 2. Порівняння завадостійкості згорткового коду NASA та покращеного коду

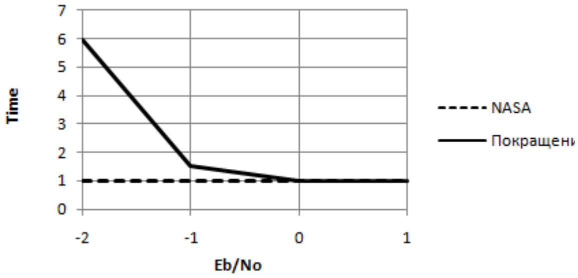

Рис. 3. Порівняння часової складності декодування згорткового коду NASA та покращеного коду

Щоб дослідити вплив довжини *t* контрольної суми на рівень завадостійкості за різних значень довжини вхідної бітової послідовності *L*, за співвідношення E_b/N_0=–1 Дб у каналі зв'язку із BPSK-модуляцією та неквантованим м'яким рішенням проведено чисельний експеримент, результати якого подано в табл. 2. Як бачимо, довжина контрольної суми, за якої забезпечується найвища завадостійкість коду (у таблиці відповідні значення бітової помилки BER виділені жирним), перебуває в межах від 2 до 8, залежно від довжини кодованого блоку. Подальше збільшення довжини контрольної суми до підвищення завадостійкості не приводитиме, оскільки воно не компенсуватиме зростання кількості редукованих переходів, які погіршують якість коду. Варто також зауважити, що зі зростанням довжини кодованого блоку оптимальна довжина контрольної суми теж збільшується.

Табл. 2. Вплив довжини контрольної суми на завадостійкість коду, BER

L \ *t*	1	2	3	4	5	6	7	8	9	10
100	0,0029	**0,0026**	0,0026	0,0029	0,0035	0,0041	0,0048	0,0063	0,0061	0,0090
200	0,0029	0,0026	**0,0024**	0,0027	0,0028	0,0028	0,0032	0,0031	0,0038	0,0041
300	0,0030	0,0028	0,0026	**0,0024**	0,0025	0,0028	0,0025	0,0027	0,0030	0,0031
400	0,0032	0,0027	0,0027	**0,0025**	0,0029	0,0026	0,0026	0,0028	0,0027	0,0025
500	0,0032	0,0030	0,0027	0,0026	0,0025	0,0028	**0,0024**	0,0027	0,0026	0,0027
600	0,0032	0,0028	0,0027	0,0027	0,0027	**0,0022**	0,0024	0,0024	0,0029	0,0025
700	0,0032	0,0032	0,0030	0,0027	0,0028	0,0028	**0,0023**	0,0026	0,0025	0,0025
800	0,0037	0,0031	0,0033	0,0032	0,0030	0,0029	0,0026	**0,0024**	0,0028	0,0029
900	0,0034	0,0032	0,0032	0,0031	0,0031	0,0029	0,0026	**0,0025**	0,0027	0,0031

Список літератури

1. *Завадський І.О.* Завадостійкі коди на основі скінченних автоматів як узагальнення згорткових кодів / І.О. Завадський // Вісник Київського національного університету ім. Т. Шевченка. Серія: фізико-математичні науки. – 2014. – №3. – с. 105–110.

2. *Морелос-Сарагоса Р.* Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение. – М.: Техносфера, 2005. – 320 с.

References

1. ZAVADSKYI I.O. (2014) The error-correcting codes by means of finite automatons as generalization of convolutional codes. *Bulletin of Taras Shevchenko National University of Kyiv. Series Physics & Mathematics*. No 3. pp. 105–110.

2. MORELOS-ZARAGOZA R. (2002) *The Art of Error Correcting Coding*. John Wiley & Sons. – 320 p.

Надійшла до редколегії 06.10.2014