Vol. 3, No. 1, 2021

DOI: https://doi.org/10.23939/csn2021.01.001

УДК 519.6:004.77

ПРОГРАМНА МОДЕЛЬ КОДІВ РІДА – СОЛОМОНА

Є. Я. Ваврук[1], Б. Р. Попович[3], Р. Б. Попович[2]

Національний університет "Львівська політехніка",
[1] кафедра електронних обчислювальних машин,
[2] кафедра спеціалізованих комп'ютерних систем,
[3] Національний медичний університет, кафедра медичної інформатики
e-mail: yevhenii.y.vavruk@lpnu.ua

Розроблено програму для моделювання завадостійких кодів Ріда – Соломона на основі об'єктно-орієнтованої технології. Вхідними даними для системи є блоки байтів для передавання через канал зв'язку. В цих блоках можуть виникати помилки. Створена програма реалізує коди типу (255, 239) та (255, 223) для скінченного поля із 256 елементів GF(2^8) зі стандартними породжуючими багаточленами $x^8+x^4+x^3+x^2+1$ та $x^8+x^7+x^2+x+1$. Крім того, передбачено можливість за необхідності додавати інші типи кодів та багаточлени, які породжують скінченне поле.

Ключові слова: завадостійкий код; скінченне поле; процедура кодування; процедура декодування.

Вступ

Потреба використання лише достовірної інформації зумовила застосування різних засобів (криптологія, завадостійке кодування, стеганографія) [1] для захисту даних від завад. Серед великої кількості кодів, які забезпечують завадостійкість повідомлення, виділяється код Ріда – Соломона [6]. Код має багато застосувань, серед яких найпопулярнішими є споживчі технології (MiniDiscs, компакт-диски, DVD-диски, диски Blu-ray, QR-коди), технології передавання даних (DSL, WiMAX), системи мовлення (супутниковий зв'язок, DVB, ATSC), системи зберігання даних (RAID 6).

З математичного погляду – це потужний код із високою коригувальною здатністю, водночас алгебраїчні процедури кодування та декодування для нього прості, ефективні та прозорі [3, 5, 7].

Окреслення проблеми

Розширення сфер використання кодів Ріда – Соломона спонукає до розроблення та впровадження нових методів та пристроїв їх кодування та декодування.

Хоча існують доступні програми, які моделюють роботу кодів Ріда – Соломона, проте:

– їх важко використати через відсутність відповідних їх описів та проблему отримання версій, які належно працюватимуть;

– вони не реалізують усіх потрібних варіантів опрацювання даних.

Цим зумовлена потреба в розробленні програмної моделі кодів Ріда – Соломона.

Завдання роботи

Розробити із використанням об'єктно-орієнтованої технології програму, яка моделює роботу кодів Ріда – Соломона, передбачивши можливість легкої її модифікації на різні типи кодів та стандартні багаточлени.

Отримані результати

Для розроблення використано середовище візуального програмування Delphi версії 7.0.

Скінченне поле із q елементів (q – натуральний степінь простого числа) позначаємо через GF(q). У розробленій програмі процедури кодування та декодування, а також операції додавання та множення для цих процедур реалізовано в скінченному полі GF(2^8) із 256 елементів. Це поле й виконання операцій у ньому залежать від багаточлена, що породжує поле. Реалізовано стандарт DVB (багаточлен $x^8+x^4+x^3+x^2+1$) та стандарт Intelsat (багаточлен $x^8+x^7+x^2+x+1$).

Якщо ж треба було б моделювати завадостійкі коди Ріда – Соломона над скінченним полем із кількістю елементів понад 256, тобто працювати з полями GF(2^9), GF(2^{10}), GF(2^{11}), GF(2^{12}), GF(2^{13}), GF(2^{14}), GF(2^{15}), GF(2^{16}), то елементи цих полів необхідно було б зображати не як байти, а як слова. Для цього потрібно внести відповідні зміни в функцію множення елементів скінченного поля. Зауважимо, що коди Ріда – Соломона над зазначеними скінченними полями також починають використовувати.

Блок-схему розробленої програми зображено на рис. 1. Для опису кодування, виникнення помилок у каналі зв'язку та декодування програма працює із трьома файлами: data.txt, code.txt та decode.txt. Згадані файли подано в програмі як файли типу txt.

Файл data.txt є блоком даних, які надсилаються у канал зв'язку, де вони, можливо, будуть спотворені через виникнення фізичних завад. Він складається із рядків, кожен з яких містить напис: i-й байт (i = 0, 1, …, 238 для кодів типу (255, 239) та i = 0, 1, …, 222 для кодів типу (255, 223)) й значення відповідного байта в шістнадцятковій системі числення.

З метою захисту корисних (інформаційних) байтів від спотворень у каналі зв'язку виконується кодування цих даних і створюється файл code.txt. У цьому файлі більше байтів, ніж у початковому файлі data.txt, у нашому конкретному випадку 255 байтів.

Файл decode.txt є протоколом декодування даних, які отримані після проходження закодованих даних через канал зв'язку. Він містить повідомлення про номери байтів файла code.txt, в яких сталися помилки в каналі зв'язку, та правильні (відтворені) значення цих спотворених байтів. Крім того, у цьому файлі наведено дані, отримані на етапах декодування – конкретні обчислені значення синдрому помилок, багаточлена місць (локаторів) помилок та багаточлена значень помилок.

Далі описано процедури кодування та декодування для кодів Ріда – Соломона [2, 5, 7], які використано під час розроблення програмної моделі.

Кодування для кодів Ріда – Соломона

Нехай $(d_{k-1}, d_{k-2}, ..., d_1, d_0)$ позначає k m-бітових символів даних (у нашому випадку байтів), які треба передати через канал зв'язку (або зберігати в пам'яті). Ці символи розглядаємо як елементи скінченного поля GF(2^m) та кодуємо кодовим словом $(c_{n-1}, c_{n-2}, ..., c_1, c_0)$ з $n > k$ символів. Ці кодові символи передаємо через канал зв'язку.

Для коду Ріда – Соломона над GF(2^m), $n = 2^m - 1$, k – непарне, код може виправляти до $t = (n-k)/2$ спотворених символів включно. Процес кодування описуємо як перетворення багаточлена даних $D(z) = d_{k-1}z^{k-1} + d_{k-2}z^{k-2} + ... + d_1 z + d_0$ на кодовий багаточлен $C(z) = c_{n-1}z^{n-1} + c_{n-2}z^{n-2} + ... + c_1 z + c_0$. $C(z)$ ділиться на твірний багаточлен коду $G(z) = \prod_{i=0}^{2t-1} (z - \alpha^{i+1})$, де α – примітивний елемент скінченного поля.

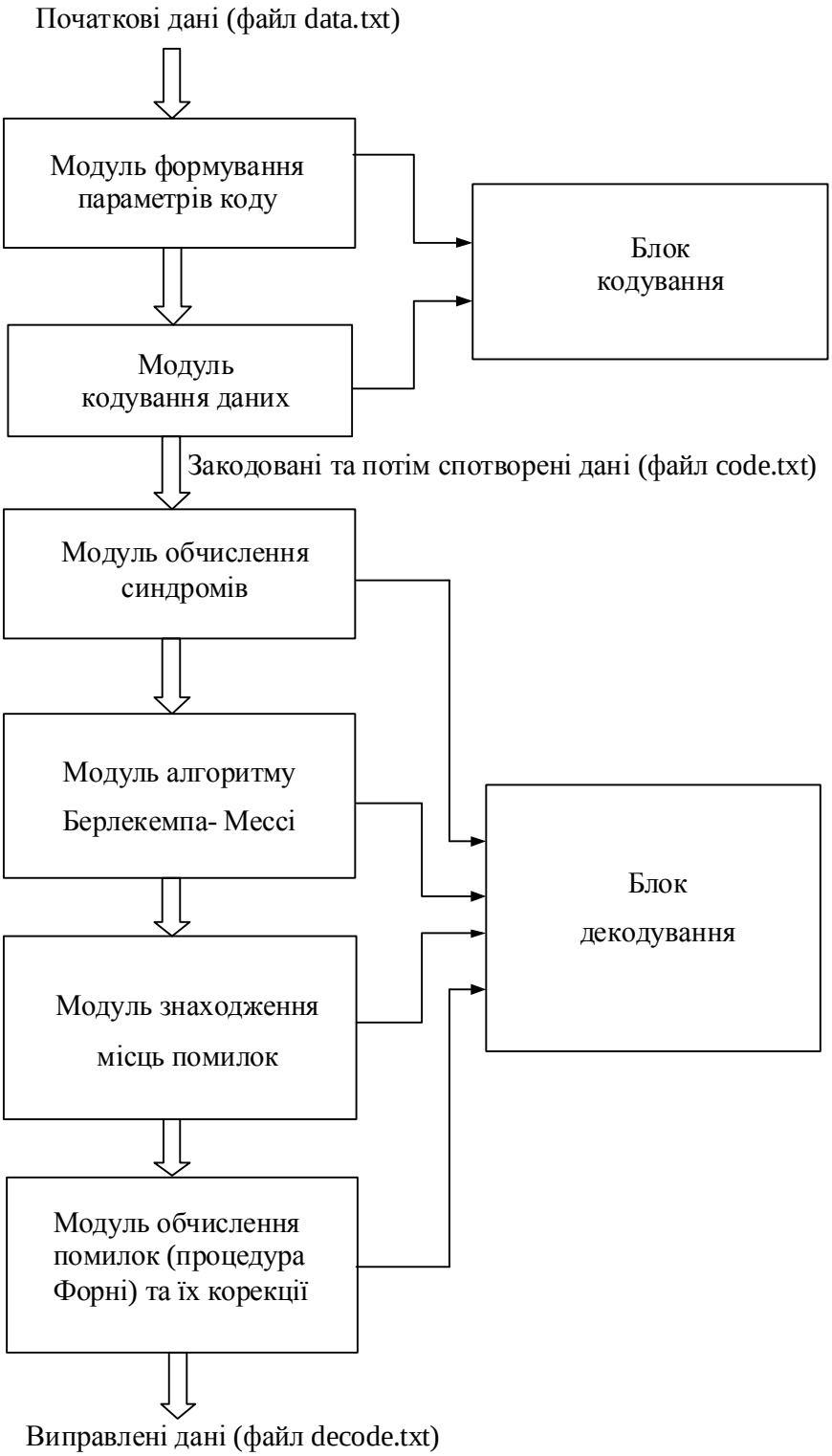

Рис. 1. Блок-схема програми

Систематичний процес кодування утворює кодові слова, що складаються із символів даних, за якими йдуть контрольні (перевіркові) символи. Кодові слова утворюємо так. Багаточлен $z^{n-k}D(z)$ степеня $n-1$ ділимо на багаточлен $G(z)$ степеня $2t=n-k$. Залишок $P(z)$ від цього ділення є багаточленом степеня щонайбільше $n-k-1$ і його коефіцієнти дають контрольні символи.

Декодування для кодів Ріда –Соломона

$C(z)$ позначає багаточлен кодового слова, яке було надіслане, а $R(z)$ – багаточлен прийнятого кодового слова. Входом декодера є $R(z)$, він задовольняє співвідношення

$$R(z) = C(z) + E(z).$$

Якщо степінь e багаточлена значень помилок $E(z)$ не дорівнює нулю, то це означає, що під час передавання трапились помилки. Багаточлен $E(z)$ можна записати у вигляді

$$E(z) = Y_1 z^{i_1} + Y_2 z^{i_2} + \ldots + Y_e z^{i_t},$$

припустивши, що помилки значень $Y_1, Y_2, \ldots, Y_e$ сталися на місцях помилок $X_1 = \alpha^{i_1}, X_2 = \alpha^{i_2}, \ldots, X_e = \alpha^{i_e}$.

Завдання декодера – визначити $E(z)$, маючи на вході $R(z)$. Декодер виправляє помилки, віднімаючи $E(z)$ від $R(z)$. Якщо $e \le t$, то це можливо, тобто можна виправити до t помилок включно.

Перший етап декодування полягає в обчисленні значень синдромів

$$s_i = R(\alpha^{i+1}) = E(\alpha^{i+1}), 0 \le i \le 2t-1.$$

Якщо всі $2t$ значення синдромів дорівнюють нулю, помилок не сталося. В іншому випадку $e > 0$ і використовуємо багаточлен синдромів $S(z) = s_0 + s_1 z + \ldots + s_{2t-1} z^{2t-1}$, щоб знайти місця помилок і значення помилок.

Означимо багаточлен місць (локаторів) помилок $\Lambda(z)$ степеня e та багаточлен значень помилок $\Omega(z)$ степеня щонайбільше $e-1$:

$$\Lambda(z) = \prod_{j=1}^{e} (1 - X_j z),$$

$$\Omega(z) = \sum_{i=1}^{e} Y_i X_i \prod_{j=1, j\ne i}^{e} (1 - X_j z).$$

Ці багаточлени зв'язані із $S(z)$ так званим ключовим рівнянням:

$$\Lambda(z) S(z) \equiv \Omega(z) \bmod z^{2t}.$$

Другий етап декодування – розв'язання ключового рівняння для визначення $\Lambda(z)$ та $\Omega(z)$ за $S(z)$ – є найскладнішою частиною декодування. Для його розв'язання використано алгоритм Берлекемпа – Мессі [2, 4] – ітеративну процедуру для розв'язання ключового рівняння. Ми реалізували модифікацію цього алгоритму, в якій відсутні операції знаходження оберненого в скінченному полі. Вона приваблива для подальшої апаратної реалізації.

На третьому етапі знаходимо місця помилок, перевіряючи, чи $\Lambda(\alpha^{-j}) = 0$ для кожного $j, 0 \le j \le n-1$. Цей процес називають пошуком Чена [5]. Якщо $\Lambda(\alpha^{-j}) = 0$, то α^j є одним із місць помилок (скажімо, X_i). Інакше кажучи, r_j є помилкою, і її треба скоригувати.

На четвертому етапі обчислюємо значення помилки Y_i, яке потім додаємо до r_j, за допомогою такого виразу (процедура Форні):

$$Y_i = -\left. \frac{z\Omega(z)}{z\Lambda'(z)} \right|_{z = \alpha^{-j}},$$

де $\Lambda'(z)$ позначає формальну похідну для $\Lambda(z)$.

Реалізація процесів кодування й декодування залежить від того, як реалізовано операції скінченного поля. Як правило, в алгоритмах використовують такі операції: додавання, піднесення до квадрата, множення довільного елемента на константу, множення (довільних елементів), знаходження оберненого відносно множення елемента. Операції додавання, піднесення до квадрата, множення на константу порівняно прості, а операції множення та знаходження оберненого складніші.

У результаті роботи запропонованої програми користувач побачить в одному із вікон програми блок початкових даних. Ці дані можна задавати й записувати у файл data.txt (рис. 2).

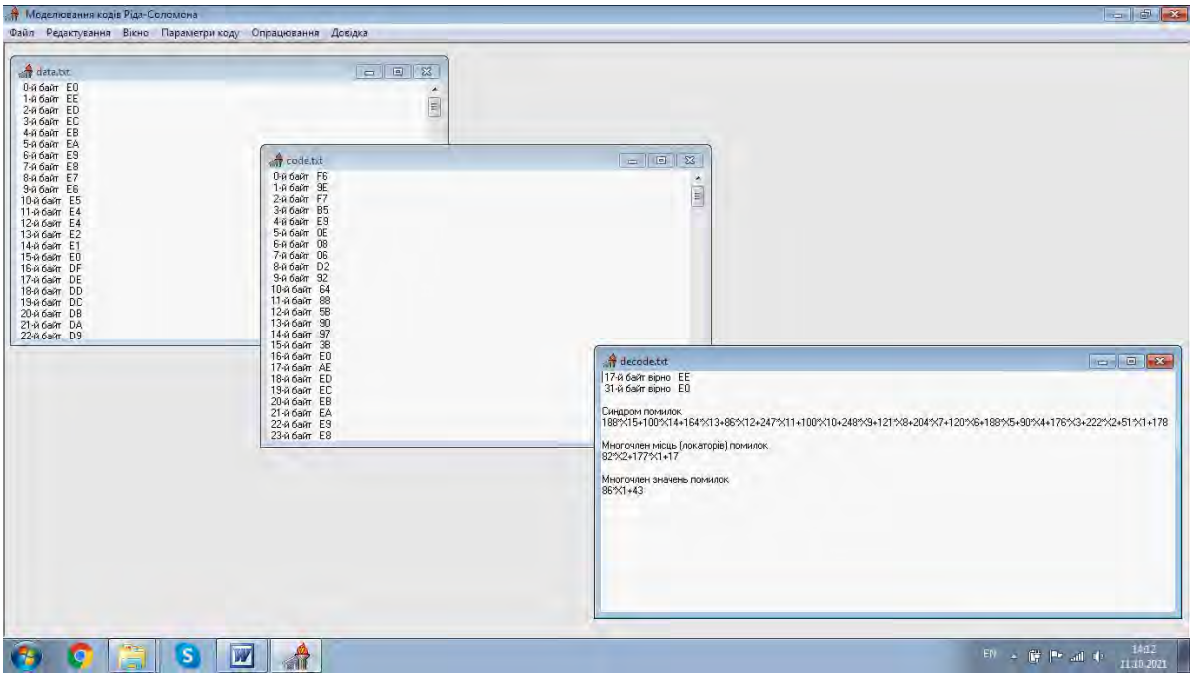

Рис. 2. Вікно програми із початковими й закодованими даними та протоколом декодування

В іншому вікні можна переглядати закодовані початкові дані, спотворювати їх (імітуючи помилки в каналі зв'язку) і записувати в файл code.txt.

У третьому вікні можна переглядати протокол декодування закодованих (і потім спотворених) початкових даних. У цьому протоколі вказано номери спотворених байтів та правильні значення цих байтів (тобто ті, які були до спотворення). Крім цього, наведено обчислені під час декодування проміжні результати.

Пункт меню "Параметри коду" дає змогу вибирати тип коду Ріда – Соломона ((255, 239) або (255, 223)) та стандартний багаточлен скінченного поля (стандарт DVB або стандарт Intelsat), згідно з якими виконують обчислення під час кодування та декодування. Крім того, передбачено можливість за необхідності легко додати інші типи кодів та багаточлени, які породжують скінченне поле.

Пункт меню "Опрацювання" дає змогу запускати процес кодування чи декодування.

Тестування роботи програмного продукту, виконане на різних блоках даних, показало ефективність виправлення помилок. Під час перевірки програмного засобу на надійність програма запускалася із неправильними або суперечливими даними. Кожного разу це було виявлено і програма видавала відповідне повідомлення.

Висновки

Розроблено програму для моделювання завадостійких кодів Ріда – Соломона на основі об'єктно-орієнтованої технології. Вхідними даними для неї є блоки байтів для передавання через

канал зв'язку. В цих блоках можливі помилки. Створена програма реалізує коди типу (255, 239) та (255, 223) для породжуючих багаточленів скінченного поля із 256 елементів GF(2^8) $x^8+x^4+x^3+x^2+1$ (стандарт DVB) та $x^8+x^7+x^2+x+1$ (стандарт Intelsat). Крім того, передбачено можливість за необхідності легко додавати інші типи кодів та багаточлени, які породжують скінченне поле. Програмний продукт може бути рекомендований для використання під час проєктування апаратних та програмних засобів для різноманітних систем передавання інформації.

1. Emets V., Melnyk A., Popovych R. (2003). Suchasna kryptografiia: osnovni poniattia. Lviv: BaK, 144 p. (In Ukrainian).

2. Berlekamp E. R. (2015). Algebraic Coding Theory. Singapore: World Scientific Publishing Co, 501 p.

3. Lin S., Costello D. J. (2004). Error Control Coding. Pirson: Prentice Hall, 1272 p.

4. Massey J. L. (1969). Shift-register synthesis and BCH decoding, IEEE Transactions on Information Theory, Vol. 15, No. 1, pp. 122–127.

5. Reed I. S., Chen X. (1999). Error-Control Coding for Data Networks. Boston: Kluwer Academic Publishers, 549 p.

6. Reed I. S., Solomon G. (1960). Polynomial Codes over Certain Finite Fields, Journal of the Society for Industrial and Applied Mathematics, Vol. 8, No. 2, pp. 300–304.

7. Tomlinson M., Tjhai C. J., Ambroze M. A., Ahmed M., Jibril M. (2017). Error-Correction Coding and Decoding: Bounds, Codes, Decoders, Analysis and Applications. Springer, 522 p.

PROGRAM MODEL OF REED – SOLOMON CODES

E. Vavruk[1], B. Popovych[3], R. Popovych[2]

Lviv Polytechnic National University,
[1] Department of Electronic Computing Machines,
[2] Department of Specialized Computer Systems,
[3] National Medical University, Department of Medical Informatics

Software is designed for modeling of Reed – Solomon codes on a base of object-oriented technology. Input data for system are blocks of bytes for transmitting through communication channel, where errors can occur in the blocks. Designed program realizes codes of (255, 239) and (255, 223) type for finite field GF(2^8) with standard generating polynomials $x^8+x^4+x^3+x^2+1$ and $x^8+x^7+x^2+x+1$. Moreover, a possibility is provided to add other types of codes and generating polynomials.

Key words: error correcting code; finite field; encoding; decoding.