УДК 621.391.3:681.3.06

[1]**Кубицкий В.И.**, к.т.н.,
[2]**Жуков И.А.**, д.т.н.

АЛГОРИТМЫ РЕАЛИЗАЦИИ ПОСЛЕДОВАТЕЛЬНОГО МЕТОДА КОДИРОВАНИЯ ПОЛИНОМИАЛЬНЫХ КОДОВ

[1]**Всероссийский научно-исследовательский институт радиоаппаратуры**
vkubitski@mail.ru

[2]**Национальный авиационный университет**
zhuia@ukr.net

Разработаны алгоритмы программ для реализации последовательного метода кодирования кодов Лагранжа и определена их операционная сложность

Ключевые слова: коды, кодирование, алгоритмы, сложность алгоритмов

Введение

Во многих компьютерных системах для защиты информации от воздействия помех применяются методы помехоустойчивого кодирования. Требования к верности передачи, ввода, обработки и хранения информации в этих системах значительно выше, чем в обычных производственных или коммерческих системах.

В [1] разработаны требования к кодам, учитывающие особенности компьютерных систем. Исследования показали, что наиболее полно удовлетворяют этим требованиям и являются перспективными для применения в системах обработки, передачи и хранения информации полиномиальные коды Лагранжа [2].

В [3] разработан последовательный метод кодирования кода Лагранжа. Предложим алгоритмы, реализующие данный метод и определим их сложность.

Сложность алгоритмов будем характеризовать операционной сложностью – количеством операций в конечном поле $GF(2^m)$ (модульных операций): сложение ($N_{\oplus}$), умножение ($N_{\otimes}$), инвертирование ($N_{\ominus}$).

Последовательный метода кодирования

Последовательный метод кодирования состоит в следующем [3]:

При кодировании кода Лагранжа вычисление значений кодового полинома $f(x)$ в каждом последующем j-ом контрольном узле производится с учётом значений f_i этого полинома в информационных узлах и вычисленных ранее значений f_{s+j-1} этого полинома в предыдущих $(j-1)$-ых контрольных узлах:

$$f(\beta_j) = -\sum_{i=0}^{s+j-1} f_i \prod_{l=j+1}^{r} (x_i - \beta_l)/(\beta_j - \beta_l), \qquad j = \overline{1,\ r}, \tag{1}$$

где $\prod_{l=j+1}^{r} (x_i - \beta_l)/(\beta_j - \beta_l) = -L^{(i)}_{S_{j-1}}(\beta_j) = L^{(j)}_{T_j}(x_i), \qquad S_{j-1} = S \cup \{\beta_1, \ldots, \beta_{j-1}\},$

$x_i \in S_{j-1},\ T_j = T \setminus \{\beta_1, \ldots, \beta_j\}, \qquad T = \{\beta_1, \ldots, \beta_r\}, \quad T_r = \varnothing, \qquad \beta_j \in T.$

Здесь используются обозначения: S, T, V – подмножества поля F_q; $S = \{x_0, x_1, \cdots, x_s\}$ – множество информационных узлов мощности k; $T = \{\beta_1, \cdots, \beta_r\}$ – множество контрольных узлов мощности r; $f(\beta_j)$ – контрольные символы закодированного слова.

Метод вычисления контрольных символов в соответствии с формулой (1)

назван последовательным методом, а процедура вычисления – последовательным алгоритмом кодирования кода Лагранжа.

Рассмотрим два случая при кодировании.

1) Необходимо вычислять коэффициенты Лагранжа, т.е. при $L^{(i)}(x) \neq const$. Такая необходимость возникает, например, при исправлении стираний, когда заранее неизвестно в каких интерполяционных узлах потребуется вычислять значения кодового полинома. Поэтому коэффициенты $L^{(i)}(x)$ вычисляются только после определения местоположения стираний.

2) Узлы интерполирования фиксированы, значения $L^{(i)}(x)$ могут быть вычислены заранее и храниться в памяти как постоянные величины, которые можно использовать в процессе вычислений. В этом случае, как показано в [4], выражение для вычисления значений полинома в контрольных узлах принимает вид

$$f(\beta_j) = \sum_{i=0}^{s+j-1} f_i A_{ji}, \qquad j = \overline{1,\ r}, \tag{2}$$

где $A_{ji} = L^{(i)}_{S_{j-1}}(\beta_j) = const$.

Алгоритмы кодирования

Запишем формулу (1) в виде выражений:

$$f(\beta_1) = f_{s+1} = -\frac{1}{b_{12}b_{13}\ldots b_{1r}} \sum_{i=0}^{s} ((\ldots(f_i a_{ir})\ldots)\, a_{i3})\, a_{i2},$$

$$f(\beta_2) = f_{s+2} = -\frac{1}{b_{23}b_{24}\ldots b_{2r}} \left[\sum_{i=0}^{s} ((\ldots(f_i a_{ir})\ldots)\, a_{i4})\, a_{i3} + ((\ldots(f_{s+1} b_{1r})\ldots)\, b_{14})\, b_{13} \right],$$

$$f(\beta_3) = f_{s+3} = -\frac{1}{b_{34}b_{35}\ldots b_{3r}} \left[\sum_{i=0}^{s} ((\ldots(f_i a_{ir})\ldots)\, a_{i5})\, a_{i4} + ((\ldots(f_{s+1} b_{1r})\ldots) b_{15}) b_{14} + \right.$$

$$\left. + ((\ldots(f_{s+2} b_{2r})\ldots) b_{25})\, b_{24} \right],$$

$$\vdots \tag{3}$$

$$f(\beta_{r-2}) = f_{s+r-2} = -\frac{1}{b_{r-2,r-1} b_{r-2,r}} \left[\sum_{i=0}^{s} (f_i a_{ir}) a_{i,r-1} + (f_{s+1} b_{1r}) b_{1,r-1} + (f_{s+2} b_{2r}) b_{2,r-1} + \right.$$

$$\left. + \ldots + (f_{s+r-3} b_{r-3,r-1}) b_{r-3,r} \right],$$

$$f(\beta_{r-1}) = f_{s+r-1} = -\frac{1}{b_{r-1,r}} \left[\sum_{i=0}^{s} f_i a_{ir} + f_{s+1} b_{1r} + f_{s+2} b_{2r} + \ldots + f_{s+r-2} b_{r-2,r} \right],$$

$$f(\beta_r) = f_{s+r} = -\left[\sum_{i=0}^{s} f_i + f_{s+1} + f_{s+2} + \ldots + f_{s+r-1} \right],$$

где $b_{12} = \beta_1 - \beta_2$, $b_{13} = \beta_1 - \beta_3$, $b_{23} = \beta_2 - \beta_3$,

$\vdots$

$b_{1r} = \beta_1 - \beta_r$, $b_{2r} = \beta_2 - \beta_r, \ldots,$ $b_{r-1,r} = \beta_{r-1} - \beta_r$,

$a_{i2} = x_i - \beta_2$, $a_{i3} = x_i - \beta_3, \ldots,$ $a_{ir} = x_i - \beta_r$.

Обозначим

$$\gamma_1 = -1/b_{12}b_{13}\ldots b_{1r},$$

$$\gamma_2 = -1/b_{23}b_{24}\ldots b_{2r},\ldots,\quad \gamma_{r-1} = -1/b_{r-1,r}.$$

Алгоритм, позволяющий определять значения кодового полинома в контрольных узлах в соответствии с формулами (3), следующий [5]:

1. Для каждого $i = \overline{0, s}$, задавая значения $l = \overline{r, 2}$, вычисляем величины $a_{il} = x_i - \beta_l$ и определяем суммы: $\sum_{i=0}^{s} f_i a_{ir}$, $\sum_{i=0}^{s}(f_i a_{ir})\, a_{i,r-1},\ldots,\ \sum_{i=0}^{s}((\ldots(f_i a_{ir})\ldots)\, a_{i3})\, a_{i2}$. Отдельно определяется $\sum_{i=0}^{s} f_i$.

2. Вычисляем $\gamma_j = -1/\prod_{l=j+1}^{r}(\beta_j - \beta_l)$. На первом шаге $j = 1$, в дальнейшем каждый раз при обращении к этому пункту увеличиваем j на 1 до $(r-1)$.

3. Определяем контрольные символы f_{s+j} ($j = \overline{1, r-1}$). При $j = r-1$ переходим к пункту 5.

4. Для каждого $j = \overline{1, r-1}$, задавая значения $l = \overline{r, j+2}$, вычисляем произведения $\alpha_j^{(l)} b_{jl}$, где

$$\alpha_j^{(l)}:\quad \alpha_j^{(r)} = f_{s+j},\quad \alpha_j^{(r-1)} = f_{s+j}b_{jr},\ \ldots,$$
$$\alpha_j^{(j+2)} = (\ldots(f_{s+j}b_{jr})\ldots)\, b_{j,j+3};$$

$b_{jl} = \beta_j - \beta_l$. При этом прибавляем величины f_{s+j} к сумме $\sum_{i=0}^{s} f_i$. После каждого изменения j выполняем пункт 2.

5. Суммируя f_{s+r-1} и $\sum_{i=0}^{s+r-2} f_i$, вычисляем f_{s+r}.

Блок-схема последовательного алгоритма кодирования (при $L^{(i)}(x) \neq const$) приведена на рис. 1.

Количество операций в поле $GF(2^m)$ для последовательного алгоритма составляет (при $L^{(i)}(x) \neq const$):

$$N_{\oplus} = n(2r-1) - r(r+1),$$

$$N_{\otimes} = (n-1)(r-1),$$

$$N_{\circlearrowleft} = r-1.$$

Здесь $n = k + r$ – длина кодового сообщения.

Расчёт количества операций производится с учётом наличия $(r-1)$ регистров памяти для хранения величин b_{jl}. С учётом наличия памяти для хранения n узлов интерполирования необходимо иметь $(n+r-1)$ регистров.

Блок-схема последовательного алгоритма кодирования при $L^{(i)}(x) = const$, реализующая выражение (2), приведена на рис. 2, блок-схема алгоритма вычисления величин $L^{(i)}(x)$ – на рис. 3.

Количество операций в поле $GF(2^m)$ для последовательного алгоритма кодирования при $L^{(i)}(x) = const$ равно:

$$N_{\oplus} = (2n-r-3)\, r/2\ ,$$

$$N_{\otimes} = (2n-r-2)(r-1)/2\ .$$

Для хранения коэффициентов $L^{(i)}(x)$ нужно $(r-1)(2n-r-2)/2$ регистров.

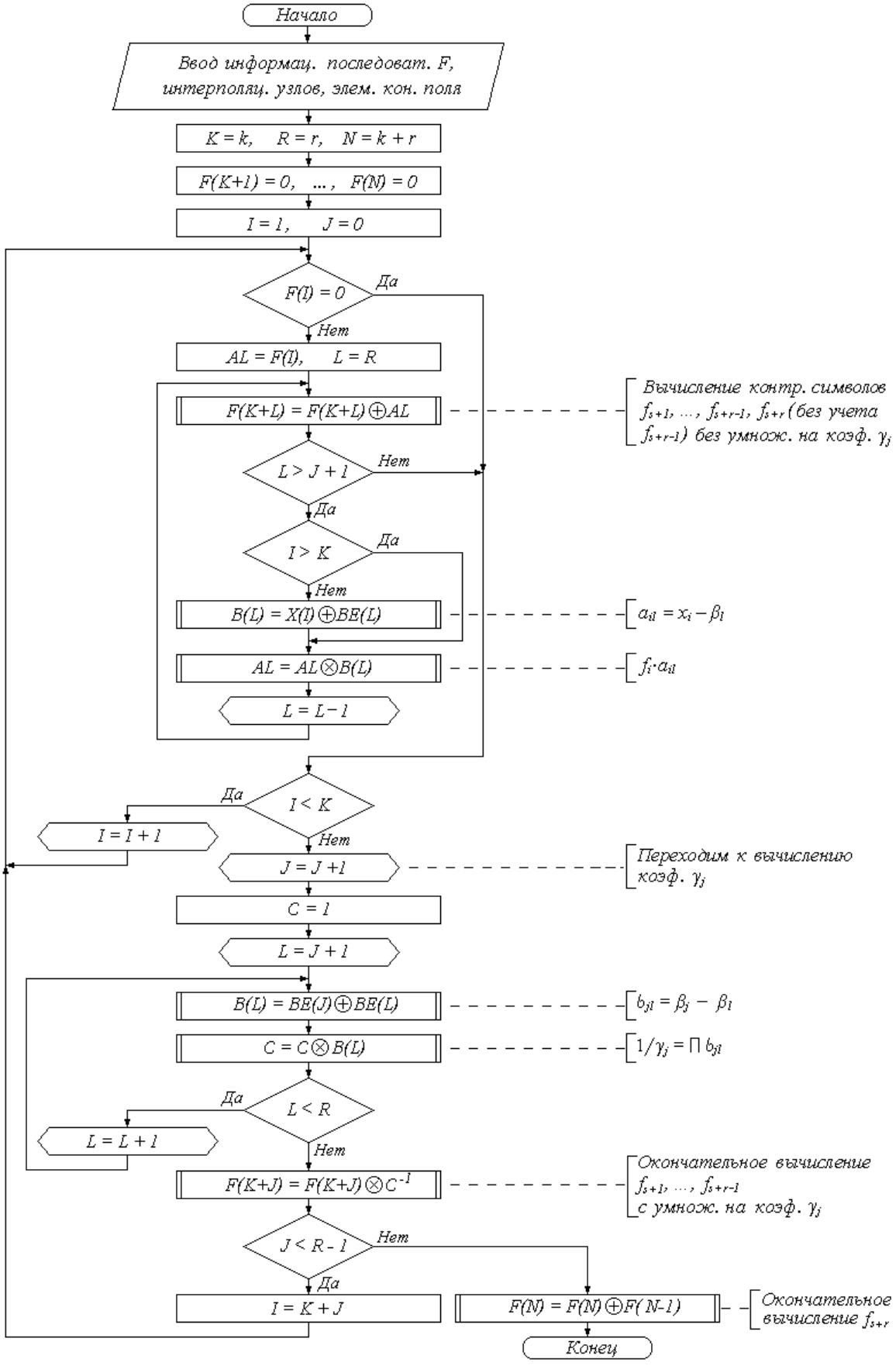

Рис. 1. Блок-схема последовательного алгоритма кодирования (при $L^{(i)}(x) \neq const$)

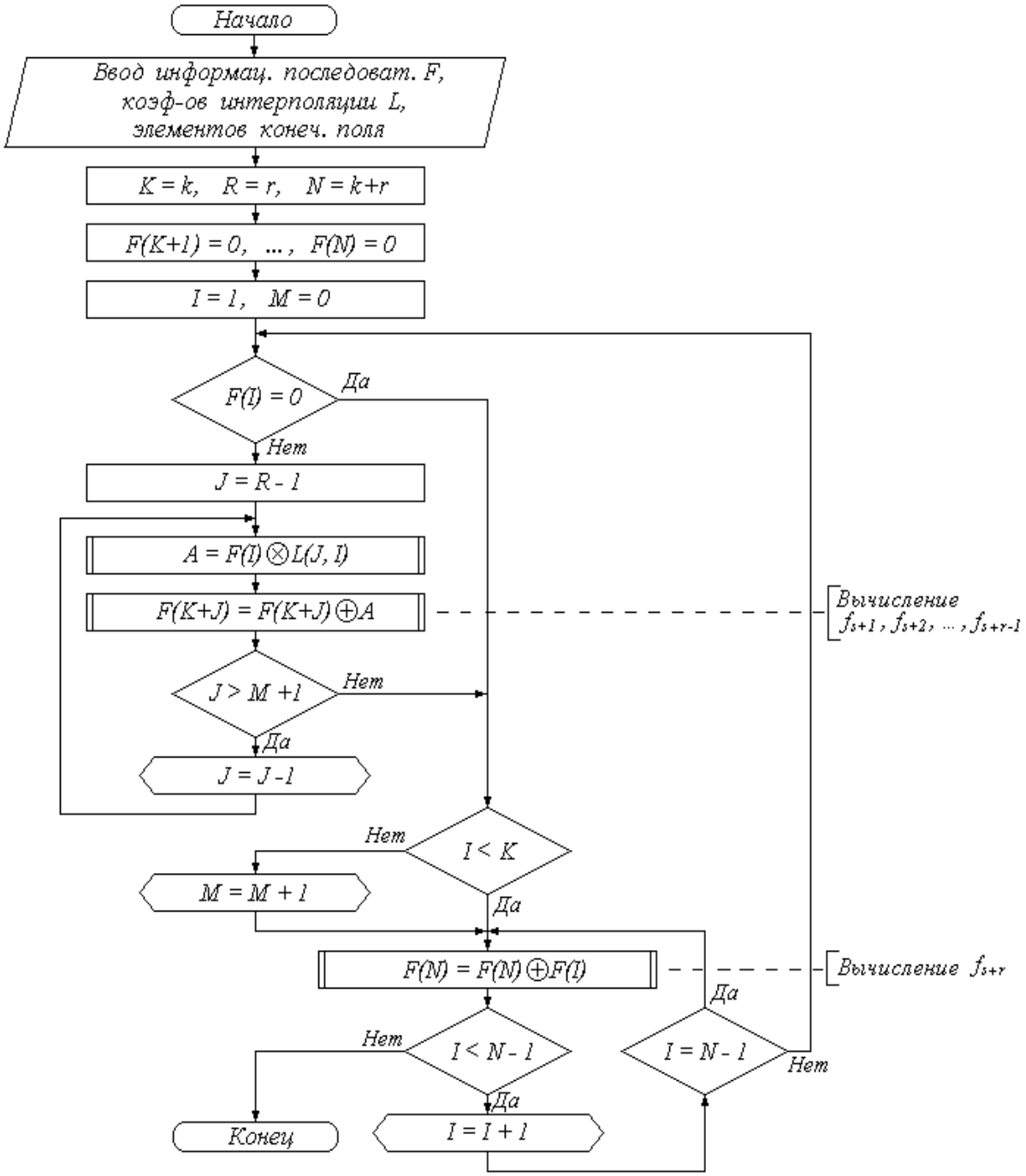

Рис. 2. Блок-схема последовательного алгоритма кодирования (при $L^{(i)}(x) = const$)

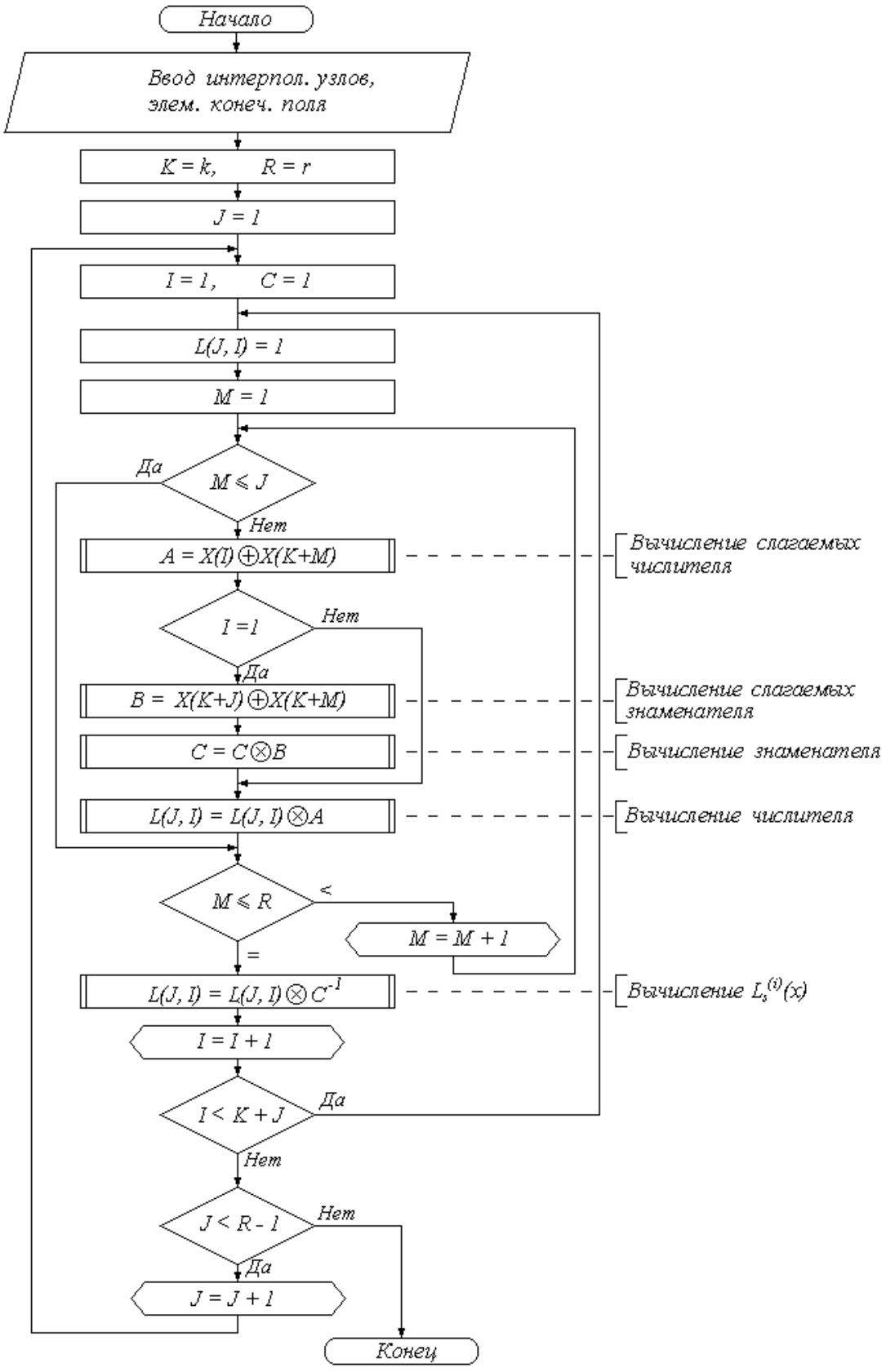

Рис. 3. Блок-схема алгоритма вычисления $L^{(i)}(x)$ для последовательного метода кодирования

Выводы

Разработанные алгоритмы позволяют выполнять программную реализацию последовательного метода кодирования кодов Лагранжа и могут использоваться не только в процедурах кодирования информации в компьютерных системах, но и для вычисления контрольных символов при декодировании [6].

Полученные аналитические выражения сложности разработанных алгоритмов позволят проводить сравнение и выбирать лучшие из методов и алгоритмов кодирования по необходимым параметрам (количеству модульных операций).

Список литературы

1. Кубицкий В.И. Восстановление стёртых пакетов в компьютерных сетях – / В.И.Кубицкий // Науч. вестн. МГТУ ГА. – 2011. – № 169 (7). – С. 65-72.

2. Амербаев В.М. Теоретические основы машинной арифметики / В.М.Амербаев. – Алма-Ата: Наука, 1976. – 324 с.

3. Кубицький В.І. Методи та засоби завадостійкого кодування інформації в комп'ютерних системах на основі кодів Лагранжа: автореферат дисертації на здобуття наукового ступеня канд. тех. наук / В.І.Кубицький. – Київ, 2012. – 21 с.

4. Кубицкий В.И. Процедуры кодирования и декодирования для полиномиальных кодов / В.И.Кубицкий // Эксплуатация программного обеспечения систем реального времени, построенных на базе микро- и мини-ЭВМ: сб. науч. тр. – 1989. – С. 67-71.

5. Zhukov I.A. Method of renewal of the lost packets at transmission in computer networks / I.A.Zhukov, V.I.Kubitsky // AVIATION IN THE XXI-st CENTURY – Safety in Aviation and Space Technologies: IV World Congress, 21–23 September 2010: proceedings cong. – K., 2010. – Vol. 1. – P. 18.5-18.8.

6. Жуков И.А. Вычисление синдромов при декодировании кодов Лагранжа / И.А. Жуков, В.И.Кубицкий // Методи та засоби кодування, захисту й ущілнення інформації: Третя міжнар. наук.-практ. конф., 20–22 квітня 2011 р.: тези допов. – Вінниця, 2011. – С. 24-25.

Статью представлено в редакцию 3.03.2014