УДК 519.722 + 519.723

ЭФФЕКТИВНОЕ СЖАТИЕ ИЗОБРАЖЕНИЙ НА ОСНОВЕ КОДИРОВАНИЯ НИЗКОЭНТРОПИЙНЫХ ИСТОЧНИКОВ

М. П. Бакулина

Учреждение Российской академии наук
Институт вычислительной математики и математической геофизики
Сибирского отделения РАН,
630090, г. Новосибирск, просп. Академика Лаврентьева, 6
E-mail: marina@rav.sscc.ru

Рассматривается одна из важных задач теории информации — задача сжатия данных, в частности изображений, без потерь качества. Предлагается эффективный алгоритм сжатия изображений, основанный на двухэтапном кодировании. Приводится сравнение экспериментальных результатов данного алгоритма и известного стандарта сжатия полутоновых изображений JPEG-LS, подтверждающее эффективность предложенного метода.

Ключевые слова: сжатие изображений, низкоэнтропийный источник, избыточность.

Введение. Сжатие данных — одна из важных задач теории информации. Её решению посвящён ряд исследований (см., например, [1]). Одним из наиболее распространённых подходов к решению задачи сжатия изображений без потерь качества является кодирование ошибки предсказания. Общая схема работы алгоритмов, основанных на этом подходе, может быть разбита на два этапа: моделирование и кодирование. На первом этапе с помощью модели некоторого класса изображений вычисляется оценка данных имеющегося изображения, позволяющая предсказывать значение яркости точки по предыдущим данным. На этапе кодирования производится расчёт ошибки предсказания, т. е. разности между реальным и предсказанным значениями яркости, в некоторое сжатое (обычно двоичное) представление. Такая схема сжатия изображений впервые была реализована в алгоритме Sunset [2]. Существуют различные вариации этого метода, например однопроходные [3], когда ошибка предсказания кодируется сразу, или двухпроходные, когда сначала вычисляются все ошибки, а на втором проходе осуществляется кодирование. Одним из лучших методов по соотношению алгоритмической сложности и эффективности сжатия является известный стандарт сжатия полутоновых изображений без потерь качества JPEG-LS [4].

Схема работы алгоритма, лежащего в основе стандарта JPEG-LS, построена на вычислении предсказанного значения яркости текущей точки. Для этого обычно используют k последовательных соседей текущей точки и представляют их в виде k-битного числа (это число называют контекстом). Более точно общий процесс предсказания и кодирования значения яркости x текущей точки можно разбить на три шага:

1) вычисление контекста текущей точки, который является функцией предыдущих значений яркости;

2) предсказание значения $\hat{x}$ яркости текущей точки по предыдущим значениям;

3) по полученным на шагах 1 и 2 данным вычисляются параметры вероятностной модели для ошибки предсказания $\varepsilon = \hat{x} - x$ и производится кодирование этой ошибки.

Для ε принимается общепризнанная гипотеза [5, 6] о том, что ошибка имеет двустороннее геометрическое распределение, т. е. симметричное относительно некоторого значения экспоненциально убывающее распределение. Поскольку вероятностная модель применяет-

ся для каждого контекста в отдельности, то удаётся сместить центр симметрии распределения так, что он находится достаточно близко к 0 [7]:

$$p(\varepsilon) = C(\theta, d)\theta^{|\varepsilon+d|}, \tag{1}$$

где $\varepsilon = 0, \pm1, \pm2, \ldots$ — ошибка предсказания; $0 < \theta < 1$; $0 \le d \le 1/2$ — некоторый параметр, характеризующий центр симметрии распределения; $C(\theta, d)$ — нормирующий множитель, задаваемый равенством

$$C(\theta, d) = \frac{1-\theta}{\theta^{1-d} + \theta^d}. \tag{2}$$

Алгоритм JPEG-LS превосходит по степени сжатия многие другие алгоритмы на фотореалистичных полутоновых изображениях. Однако на искусственных полутоновых изображениях он уже не является столь эффективным, что ограничивает его распространение на широкий класс изображений. Для случая искусственных изображений вероятность значения 0 ошибки предсказания ε значительно выше, чем для фотореалистичных изображений. Кодируется ε посимвольно на основе кода Райса — Голомба [3]. В случае искусственных изображений, где вероятность серий из нулей оказывается достаточно большой, возникает задача нахождения эффективных методов кодирования источников с малой энтропией и применения их к такого вида изображениям.

В [8] рассмотрен эффективный метод кодирования низкоэнтропийных источников. Доказано, что в отличие от известных методов он позволяет достигать любой заранее заданной избыточности при сохранении той же памяти кодера и декодера, что и у «общих» методов, тогда как его скорости кодирования и декодирования существенно выше.

Цель данной работы — создание алгоритма эффективного сжатия изображений и оценка его объёма памяти и скорости кодирования.

Алгоритм эффективного кодирования ошибок предсказания. Построим алгоритм эффективного кодирования последовательности ошибок предсказания, т. е. последовательности символов, порождаемых низкоэнтропийным бернуллиевским источником S_ε с алфавитом допустимых значений ошибки предсказания $A_\varepsilon = \{\varepsilon \,|\, \varepsilon \in [-n, n]\}$ и распределением вероятностей $p(\varepsilon)$, задаваемых формулами

$$p(\varepsilon) = \frac{(1-\theta)\theta^\varepsilon}{2\sqrt{\theta}}, \;\; \varepsilon \ne 0; \quad p(0) = 1 - \sqrt{\theta}. \tag{3}$$

Формулы (3) для распределения вероятностей следуют из (1) и (2), если принять $d = 1/2$ (при $d = 0$ и $d = 1/2$ распределение вероятностей (1) центрируется к 0 [7]) и учесть, что на практике значение $\varepsilon + d \gg d$, т. е. можно считать $|\varepsilon + d| \approx |\varepsilon|$.

Будем полагать

$$\hat{p} = \sum_{\varepsilon \in [-n, n]/\{0\}} p(\varepsilon); \quad \hat{q} = 1 - \hat{p}. \tag{4}$$

Используя (3) и (4), имеем

$$\hat{p} = \sum_{\varepsilon \in [-n, n]/\{0\}} p(\varepsilon) = 2\sum_{\varepsilon=1}^{n} p(\varepsilon) = 2\sum_{\varepsilon=1}^{n} \frac{(1-\theta)\theta^\varepsilon}{2\sqrt{\theta}} = \frac{1-\theta}{\sqrt{\theta}} \sum_{\varepsilon=1}^{n} \theta^i =$$

$$= \frac{1-\theta}{\sqrt{\theta}} \frac{\theta(1-\theta^n)}{1-\theta} = \sqrt{\theta}(1-\theta^n); \quad \hat{q} = 1 - \hat{p} = 1 - \sqrt{\theta}(1-\theta^n). \tag{5}$$

Кодирование последовательности значений ошибки предсказания ε будем осуществлять в два этапа. На первом этапе сообщение, порождаемое источником, сжимается простым кодом, а полученная на выходе последовательность кодируется на втором этапе более сложным кодом. Так как источник низкоэнтропийный, то после первого этапа длина входной последовательности существенно сокращается, что обеспечивает небольшое суммарное время кодирования в пересчёте на букву исходного сообщения. В данной работе на втором этапе будем использовать арифметический код, описанный в [9, 10], как наиболее эффективный среди известных на настоящее время алгоритмов.

Рассмотрим первый этап кодирования. Поступающую на вход последовательность символов разобьём на блоки (подслова) длины $l = \lceil 1/\sqrt{\hat{p}}\,\rceil$, где $\hat{p}$ определяется формулой из (4). Если блок состоит только из значений 0 ошибки, то кодом этого блока является 0. Если же блок содержит хотя бы один ненулевой символ ошибки (обозначим его через k, $1 \le |k| \le n$), то длина кодового слова равна $l+1$: начало этого кодового слова — любой редкий символ k, встречающийся в данном блоке, за которым следует тот же самый блок длины l. При этом наличие $(l+1)$-го символа k, расположенного в начале кодового слова, необходимо, так как он указывает на появление маловероятного символа в блоке длины l, находящемся после k.

Например, пусть $A = \{0, 1, 2\}$, $p(0) = 6/7$, $p(1) = 2/21$, $p(2) = 1/21$ и кодируется последовательность ошибок предсказания

$$00000000001000102000.$$

Из (4) следует, что $\hat{p} = 1/7$, $l = 3$ и закодированная последовательность имеет вид

$$0\ 0\ 0\ 1001\ 0\ 21020.$$

Пусть теперь $z_1 z_2 \dots z_s$ — последовательность, полученная после первого этапа кодирования, $z_i \in A$, $A = \{k \mid -n \le k \le n\}$.

На втором этапе её кодирование осуществляется арифметическим кодом из [9, 10]. Выделим в последовательности $z_1 z_2 \dots z_s$ блоки длины l, которые следуют после появления какого-либо маловероятного символа k, и «особые» символы $\underline{0}$, $\underline{k}$, не входящие в блоки, т. е. представим последовательность $z_1 z_2 \dots z_s$ в виде

$$\underline{0} \dots \underline{0}\ \ \underline{k}\ \ \underbrace{z_1 \dots z_l}_{l}\ \ \underline{0} \dots \underline{0}\ \ \underline{k}\ \ \underbrace{z'_1 \dots z'_l}_{l}.$$

Схема кодирования может быть описана следующим образом.

Особые символы $\underline{0}$ и $\underline{k}$ $(-n \le k \le n)$ кодируются с помощью кодера K_0 с вероятностями $\hat{q}^l$ и $1 - \hat{q}^l$ соответственно, где $\hat{q}$ определяется формулой из (4).

Рассмотрим кодирование символов, находящихся внутри блока $z_1 \dots z_s$ длины l. Пусть $z_1 \dots z_{i-1} = \underbrace{0 \dots 0}_{i-1}$ $(i = 1, 2, \dots, l)$. Определим вероятность символа z_i, находящегося в i-й позиции после $i-1$ появлений символа 0. Имеем

$$\tau_i^k = p\{z_i = k \mid z_1 \dots z_{i-1} = 0 \dots 0;\ z_1 \dots z_l \ne 0 \dots 0\} =$$

$$= \frac{p\{z_i = k; z_1 \dots z_{i-1} = 0 \dots 0 \mid z_1 \dots z_l \ne 0 \dots 0\}}{p\{z_1 \dots z_{i-1} = 0 \dots 0 \mid z_1 \dots z_l \ne 0 \dots 0\}}.$$

Так как

$$p\{z_i = k; z_1 \dots z_{i-1} = 0 \dots 0 \mid z_1 \dots z_l \ne 0 \dots 0\} =$$

$$= \frac{p\{z_i = k; z_1 \dots z_{i-1} = 0 \dots 0; z_1 \dots z_l \neq 0 \dots 0\}}{p\{z_i = k; z_1 \dots z_{i-1} = 0 \dots 0 \,|\, z_1 \dots z_l \neq 0 \dots 0\}} = \frac{p(|k|)\hat{q}^{i-1}}{1-\hat{q}^l}$$

и

$$p\{z_1 \dots z_{i-1} = 0 \dots 0 \,|\, z_1 \dots z_l \neq 0 \dots 0\} =$$

$$= \frac{p\{z_1 \dots z_{i-1} = 0 \dots 0; z_1 \dots z_l \neq 0 \dots 0\}}{p\{z_1 \dots z_l \neq 0 \dots 0\}} = \frac{\hat{q}^{i-1}(1-\hat{q}^{l-i+1})}{1-\hat{q}^l},$$

то, учитывая (3), окончательно получаем

$$\tau_i^k = \frac{p(|k|)}{1-\hat{q}^{l-i+1}} = \frac{(1-\theta)\theta^{|k|}}{2\sqrt{\theta}(1-\hat{q}^{l-i+1})}.$$

Кроме того,

$$p\{z_i = 0 \,|\, z_1 \dots z_{i-1} = 0 \dots 0; z_1 \dots z_l \neq 0 \dots 0\} = 1 - 2\sum_{k=1}^{n} \tau_i^k.$$

Таким образом, символ z_i, находящийся в i-й позиции после появления $i-1$ нулей в блоке $z_1 \dots z_l$, кодируется с помощью кодера K_i с вероятностями τ_i^k для символов k и $1 - 2\sum\limits_{k=1}^{n} \tau_i^k$ для 0.

Наконец, символы из блока $z_1 \dots z_l$, расположенные после появления в нём какого-либо символа k, кодируются с помощью кодера $\hat{K}$ с исходными вероятностями $\hat{q}$ и $p(|k|)$ для символов 0 и k соответственно, где $p(|k|)$ определяется формулой из (3).

Вероятности τ_i^k не хранятся в памяти кодера и декодера, а вычисляются рекуррентно. Сначала кодируется z_1 с вероятностями τ_1^k и $1 - 2\sum\limits_{k=1}^{n} \tau_1^k$ для символов k и 0 соответственно. Если $z_1 = k$, то все символы, следующие после символа k, кодируются с помощью кодера $\hat{K}$ с исходными вероятностями $p(|k|)$ и $\hat{q}$ для k и 0 соответственно. В противном случае вычисляется τ_2^k и буква z_2 кодируется с вероятностями τ_2^k и $1 - 2\sum\limits_{k=1}^{n} \tau_2^k$ для k и 0 соответственно. Таким образом, кодирование букв в блоке $z_1 \dots z_l$ осуществляется по следующей схеме.

Шаг 1. Вычисляется τ_i^k и символ z_i кодируется с вероятностями τ_i^k и $1 - 2\sum\limits_{k=1}^{n} \tau_i^k$.

Шаг 2. Если $z_i = k$, то все символы, следующие после z_i, кодируются с вероятностями $\hat{q}$ и $p(|k|)$. Иначе осуществляется переход к следующей букве и возврат к шагу 1.

Для вычисления τ_i^k используется рекуррентная формула

$$\frac{1}{\tau_{i+1}^k} = \frac{1}{\tau_i^k} - \frac{\hat{p}\hat{q}^{l-i}}{p(|k|)},$$

где $\hat{p}$, $\hat{q}$ определяются формулами (5), а $p(|k|)$ — формулой из (3). Следовательно, вычисление вероятностей τ_i^k можно организовать по схеме

$$\sigma^k := \sigma^k/\hat{q}; \quad (\hat{\tau}^k)^{-1} := (\hat{\tau}^k)^{-1} - \sigma^k \tag{6}$$

с начальными данными

$$\sigma^k := \frac{\hat{q}^l\hat{p}}{p(k)}; \quad (\hat{\tau}^k)^{-1} := \frac{1-\hat{q}^l}{p(k)}.$$

Таким же образом кодируются буквы следующего блока, причём перед каждым новым блоком начальные данные обновляются.

Рассмотрим эту конструкцию кода на примере предыдущей последовательности. Пусть $p(0) = \hat{q} = 6/7$, $p(1) = 2/21$, $p(2) = 1/21$, $\hat{p} = 1/7$, $l = 3$ и кодируется последовательность

$$0\ 0\ 0\ 1001\ 0\ 21020.$$

Представим такую последовательность в виде

$$\begin{array}{cccccccccccccc}
\underline{0} & \underline{0} & \underline{0} & \underline{1} & \underbrace{0} & 0 & \underbrace{1} & \underline{0} & \underline{2} & \underbrace{1} & 0 & \underbrace{2} & \underline{0} \\
z_1 & z_2 & z_3 & z_4 & z_5 & z_6 & z_7 & z_8 & z_9 & z_{10} & z_{11} & z_{12} & z_{13} \\
K_0 & K_0 & K_0 & K_0 & K_1 & K_2 & K_3 & K_0 & K_0 & K_1 & \hat{K} & \hat{K} & K_0.
\end{array}$$

Особые символы кодируются с помощью кодера K_0 с вероятностями

$$p(z_1) = p(z_2) = p(z_3) = p(z_8) = p(z_{13}) = (6/7)^3 = 216/343,$$

$$p(z_4) = p(z_9) = 1 - (6/7)^3 = 127/343.$$

Буквы первого блока $z_5z_6z_7$ кодируются следующим образом. Из (6) находим, что

$$(\tau_1^1)^{-1} = \frac{1-(6/7)^3}{2/21} = \frac{381}{98}, \quad (\tau_1^2)^{-1} = \frac{1-(6/7)^3}{1/21} = \frac{381}{49}.$$

Значит, символ $z_5 = 0$ кодируется с помощью кодера K_1 с вероятностью $(1 - \tau_1^1 - \tau_1^2) = = 78/127$. Переходим к символу z_6. Из (6) следует, что

$$(\tau_2^1)^{-1} = \frac{381}{98} - \frac{(6/7)^2 \cdot 1/7}{2/21} = \frac{39}{14}, \quad (\tau_2^2)^{-1} = \frac{381}{49} - \frac{(6/7)^2 \cdot 1/7}{1/21} = \frac{39}{7}.$$

Значит, $z_6 = 0$ кодируется с помощью кодера K_2 с вероятностью $(1 - \tau_2^1 - \tau_2^2) = 6/13$. Переходим к символу z_7. Из (6) также следует, что

$$(\tau_3^1)^{-1} = \frac{39}{14} - \frac{6/7 \cdot 1/7}{2/21} = \frac{3}{2},$$

поэтому $z_7 = 1$ кодируется с помощью кодера K_3 с вероятностью $\tau_3^1 = 2/3$. Аналогично кодируются буквы второго блока $z_{10}z_{11}z_{12}$. Так как $(\tau_1^1)^{-1} = 381/98$, то $z_{10} = 1$ кодируется с помощью кодера K_1 с вероятностью $\tau_1^1 = 98/381$. Оставшиеся символы z_{11} и z_{12} кодируются с помощью кодера $\hat{K}$ с вероятностями $p(z_{11}) = p(0) = 6/7$; $p(z_{12}) = p(2) = 1/21$.

Объём памяти кодера и декодера и скорость кодирования и декодирования предложенного метода характеризует

Теорема. Пусть дан низкоэнтропийный бернуллиевский источник S_ε, порождающий последовательность ошибок предсказания из алфавита $A_\varepsilon = \{\varepsilon \,|\, \varepsilon \in [-n, n]\}$ с распределением вероятностей $p(\varepsilon)$, задаваемых формулами (3), и некоторое r, $0 < r < 1$. Пусть для кодирования данного источника используется описанный выше код с $l = \lceil 1/\sqrt{\hat{p}}\,\rceil$ $(\hat{p} < 1/2)$ на первом этапе, где $\hat{p}$ определяется формулой из (4), и избыточностью $\bar{r} = r/2$ на втором. Тогда общая избыточность кода не превосходит r, а общий объём памяти кодера и декодера $V(S_\varepsilon)$ и среднее время кодирования и декодирования одного символа $T(S_\varepsilon)$ удовлетворяют неравенствам

$$V < C_1 \log \frac{1}{\hat{p}}, \quad T < C_2 \sqrt{\hat{p}} \cdot \log\left(\frac{1}{r\hat{p}}\right) \cdot \log\log\left(\frac{1}{r\hat{p}}\right) \cdot \log\log\log\left(\frac{1}{r\hat{p}}\right) + C_3,$$

где C_1, C_2 и C_3 — константы.

Доказательство. Так же, как и в [8], доказывается, что общая избыточность кода

$$R = l'\bar{r},$$

где $l' = l_1/l$ — значение средней длины кода (на символ), полученного после первого этапа кодирования. Так как вероятность появления блока, состоящего только из символов 0, выражается как $\hat{q}^l = (1-\hat{p})^l$, то

$$l' = \frac{1}{l}(\hat{q}^l + (l+1)(1-\hat{q}^l)) = 1 - \hat{q}^l + \frac{1}{l} = 1 - (1-\hat{p})^l + \frac{1}{l} \leq$$

$$\leq l\hat{p} + \frac{1}{l} < \hat{p}\Big(\frac{1}{\sqrt{\hat{p}}} + 1\Big) + \sqrt{\hat{p}} = 2\sqrt{\hat{p}} + \hat{p}.$$

При $\hat{p} < 1/2$ имеем

$$R = l'\bar{r} < \frac{r}{2}(2\sqrt{\hat{p}} + \hat{p}) < r,$$

т. е. общая избыточность не превосходит r.

Оценим среднее время кодирования и декодирования данного метода. Время кодирования, затрачиваемое на символ «сжатой» на первом этапе последовательности, равно

$$O(\log(1/R) \cdot \log\log(1/R) \cdot \log\log\log(1/R)).$$

Кроме того, время вычисления величин τ_i^k, используемых на втором этапе, не превышает

$$C\sqrt{\hat{p}} \cdot \log(1/r\hat{p}) \cdot \log\log(1/r\hat{p}) \cdot \log\log\log(1/r\hat{p}).$$

Умножая общее время второго этапа кодирования на среднюю длину кода l' и учитывая время первого этапа, равное $O(1)$, получим, что для предложенного метода среднее время кодирования и декодирования одного символа удовлетворяет неравенству

$$T < C_2\sqrt{\hat{p}} \cdot \log\left(\frac{1}{r\hat{p}}\right) \cdot \log\log\left(\frac{1}{r\hat{p}}\right) \cdot \log\log\log\left(\frac{1}{r\hat{p}}\right).$$

Таблица 1

№ п/п	Файл	Размер, байт	$k_{\text{JPEG-LS}}$	k_{NEW}	$t_{\text{JPEG-LS}}$	t_{NEW}
1	horiz	65666	0,094	**0,068**	1,09	**0,65**
2	circles	65666	0,152	**0,109**	0,74	**0,61**
3	crosses	65666	0,385	**0,286**	0,93	0,57
4	slope	65666	1,568	**1,204**	1,21	**0,73**
5	squares	65666	0,077	**0,059**	0,86	**0,41**
6	text	65666	1,629	**1,142**	1,29	**0,82**
7	среднее	65666	0,651	**0,478**	1,02	**0,63**

Экспериментальные результаты. Теорема даёт общее представление об эффективности предложенного метода. Однако более важным с точки зрения практического применения является вопрос о том, как предлагаемый метод работает на реальных данных. Для экспериментальной проверки эффективности рассмотренного алгоритма (назовём его NEW) использовался стандартный тестовый набор изображений Waterloo Repertoire GreySet1 [11]. Размеры всех изображений 256×256 пикселов, конфигурация тестового компьютера IBM PC следующая: процессор CPU Intel Pentium 4, тактовая частота 2,53 ГГц, объём оперативной памяти 512 Мбайт, операционная система Windows XP. Для сравнения был взят наиболее эффективный на настоящее время алгоритм сжатия полутоновых изображений JPEG-LS. Результаты работы обоих алгоритмов сравнивались (табл. 1, 2) по двум характеристикам: степени сжатия изображений k (в битах) и времени t (в секундах), требующимся кодеру для их упаковки. Под степенью сжатия в данном случае понимаем количество бит, которым представляется в сжатом файле один байт (8 бит) исходного (несжатого) изображения. Иначе говоря, если L — размер исходного файла, L_1 — размер сжатого файла, то степень сжатия — это $8(L_1/L)$. В табл. 1 занесены результаты по группе искусственных изображений. В табл. 2 собраны данные по группе фотореалистичных изображений. Табл. 1 показывает, что для предложенного алгоритма степень сжатия искусственных изображений в среднем на 27 % выше степени сжатия для алгоритма JPEG-LS, при этом скорость кодирования увеличилась примерно на 37 %. Для фотореалистичных изображений (см. табл. 2) новый алгоритм улучшает сжатие в среднем на 2 % по сравнению с алгоритмом JPEG-LS, скорость кодирования осталась по-прежнему высокой: улучшение составило около 21 %. Таким образом, результаты показывают хоро-

Таблица 2

№ п/п	Файл	Размер, байт	$k_{\text{JPEG-LS}}$	k_{NEW}	$t_{\text{JPEG-LS}}$	t_{NEW}
1	bird	65666	3,46	**3,31**	1,62	**1,23**
2	bridge	65666	5,78	**5,75**	1,45	**1,17**
3	camera	65666	4,31	**4,26**	1,51	1,24
4	lena	65666	4,57	**4,51**	1,32	**1,05**
5	goldhill	65666	5,27	**5,13**	1,49	**1,21**
6	montage	65666	2,72	**2,69**	1,57	**1,29**
7	среднее	65666	4,35	**4,28**	1,49	**1,20**

шую степень и скорость сжатия рассмотренного метода, что подтверждает эффективность построенного алгоритма.

Заключение. В данной работе предложен эффективный алгоритм сжатия изображений, основанный на двухэтапном кодировании. Экспериментально доказано, что для этого алгоритма степень сжатия и скорость кодирования искусственных изображений существенно выше (на 27 и 37 % соответственно), чем для алгоритма JPEG-LS. Это позволяет использовать его на практике для эффективного сжатия картографических изображений, спутниковых изображений земной поверхности, интернет-графики и т. д.

СПИСОК ЛИТЕРАТУРЫ

1. **Трофимов В. К.** Об эффективности равномерного по выходу кодирования бернуллиевских источников при неизвестной статистике сообщений // Автометрия. 2010. **46**, № 6. С. 32–39.
2. **Todd S., Langdon G. G., Rissanen J.** Parameter reduction and context selection for compression of the gray-scale images // IBM Journ. Res. Develop. 1985. **29**, N 2. P. 188–193.
3. **Howard P. G., Vitter J. S.** Fast and efficient lossless image compression // Proc. IEEE Data Compression Conference. Snowbird, Utah, 1993. P. 351–360.
4. **Weinberger M. J., Seroussi G., Sapiro G.** The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS // IEEE Trans. Image Process. 2000. **9**, N 8. P. 1310–1324.
5. **Netravali A., Limb J. O.** Picture coding: A review // Proc. IEEE. 1980. **68**, N 3. P. 366–406.
6. **Reininger R. C., Gibson J. D.** Distributions of the two-dimentional DCT coefficients for images // IEEE Trans. Commun. 1983. **31**, N 6. P. 835–839.
7. **Merhav N., Seroussi G., Weinberger M. J.** Optimal prefix codes for sources with two-sided geometric distributions // IEEE Trans. Inform. Theory. 2000. **46**, N 1. P. 121–135.
8. **Рябко Б. Я., Шарова М. П.** Быстрое кодирование низкоэнтропийных источников // Проблемы передачи информации. 1999. **35**, № 1. С. 49–61.
9. **Ryabko B. Ya., Fionov A. N.** Homophonic coding with logarithmic memory size // Algorithms and Computation. Berlin: Springer, 1997. P. 253–262.
10. **Witten I. H., Neal R. M., Cleary J. G.** Arithmetic coding for data compression // Commun. ACM. 1987. **30**, N 6. P. 520–540.
11. **Библиотека** изображений для тестирования и демонстрации алгоритмов сжатия данных. URL: http://cdb.paradice-insight.us/corpora/Corpus%20Waterloo/GreySet1/?C=N;O=A (дата обращения 1.12.2010).

Поступила в редакцию 11 ноября 2010 г.