INCREASING EFFICIENCY OF IMAGE COMPRESSION BASED ON THE RLE METHOD

M. P. Bakulina

Institute of Computational Mathematics and Mathematical Geophysics SB RAS, 630090, Novosibirsk, Russia

DOI: 10.24412/2073-0667-2023-4-73-77
EDN: QBTMBL

In this article the problem of increasing efficiency of image compression is considered. As is known, digital image intended for transmission or storage is converted to a raster image. The RLE method based on run lengths coding is especially effective for raster images with large single-color areas. Based on RLE algorithm and arithmetic code a new algorithm for run lengths coding is proposed. This algorithm gives an increase of compression ratio of raster images compared to RLE.

Key words: lossless coding, digital image, RLE, compression ratio.

References

1. Tropchenko A. Ju, Tropchenko A. A. Metody szhatija izobrazhenii, audiosignalov i video. SPb: SPbGU ITMO, 2009. 108 s.
2. Bell T. C., Moffat A., Witten I. H. Compressing the Digital Library // Proc. Digital Libraries Texas: College Station, 1994. P. 41–46
3. Jiawan Zh., Jizhou S., Zhigang S. Accelerate volume splatting by using run length encoding // Lecture Notes in Computer Science. 2003. V. 2657. P. 907–914.
4. Witten I. H., Neal R., Cleary J. G. Arithmetic coding for data compression // Comm. ACM. 1987. V. 30, N 6. P. 520–540.
5. Selomon M. Szatie dannyh, izobrazhenii i zvuka M.: Tehnosthera, 2004. 368 s.

The research was carried out within the framework of a state assignment of the Institute of Computational Mathematics and Mathematical Geophysics SB RAS (ICM&MG SB RAS) 0251-2022-0005.

ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ СЖАТИЯ ИЗОБРАЖЕНИЙ НА ОСНОВЕ МЕТОДА RLE

М. П. Бакулина

Институт вычислительной математики и математической геофизики СО РАН,
630090, Новосибирск, Россия

УДК 519.722
DOI: 10.24412/2073-0667-2023-4-73-77
EDN: QBTMBL

В работе рассмотрена актуальная на сегодняшний день задача повышения эффективности сжатия цифровых изображений. Как известно, цифровое изображение, предназначенное для передачи или хранения, преобразуется в растровое. При этом для растровых изображений с большими одноцветными областями особенно эффективен метод RLE, основанный на кодировании длин серий. На основе алгоритма RLE и арифметического кода предлагается новый алгоритм кодирования длин серий, обеспечивающий более высокую по сравнению с RLE степень сжатия растровых изображений.

Ключевые слова: кодирование без потерь, растровое изображение, RLE, коэффициент сжатия.

Введение. Постоянное увеличение объема передаваемых данных требует усовершенствование прежних и разработку новых алгоритмов сжатия данных. Среди общего объема цифровой информации изображения занимают особое место. Изображение может храниться и обрабатываться в любой модели, но перед воспроизведением оно всегда преобразуется в растровое. Это связано с конструкцией и принципом действия большинства экранных и печатных устройств. Поэтому в автоматизированном информационном обмене растровые изображения играют особую роль. Хранение и передача таких изображений занимает довольно большой объем памяти, поэтому применяется сжатие с помощью кодирования без потерь. Такое кодирование выполняется следующим образом: сначала прямоугольное изображение представляется прямоугольной матрицей цветных точек, потом цвет каждой точки записывается числом или группой чисел, а к полученной числовой или символьной последовательности применяется какой-либо алгоритм сжатия.

1. Постановка задачи. Повышение эффективности сжатия цифровых изображений является на сегодняшний день актуальной задачей. Это связано с большой распространенностью таких изображений в различных предметных областях. Так как любое изображение для передачи или хранения преобразуется в растровое, то возникает задача кодирования без потерь таких изображений с целью сокращения объема (компрессии) данных. При этом для растровых изображений с большими одноцветными областями особенно эффективен метод кодирования длин серий RLE [1–3]. Этот алгоритм поддерживается большинством форматов файлов растровых изображений: TIFF, BMP и PCX.

Исследования выполнены в рамках государственного задания ИВМиМГ СО РАН 0251-2022-0005.

Суть метода RLE состоит в замене цепочек, серий повторяющихся байтов или их последовательностей на один кодирующийся байт или счетчик числа их повторений. Первый байт указывает, сколько раз нужно повторить следующий байт. Так как метод состоит в кодировании длин повторяющихся серий, то особенно эффективным алгоритм RLE становится для изображений с большими одноцветными областями. Однако теоретически метод RLE может быть значительно улучшен, если дополнительно сжимать полученные последовательности каким-нибудь эффективным кодом.

На основе RLE и арифметического кода [4, 5] предлагается новый алгоритм кодирования, обеспечивающий более высокую по сравнению со стандартным методом RLE степень сжатия растровых изображений.

2. Алгоритм решения задачи. Метод RLE основан на кодировании длин серий встречающихся символов. Рассмотрим теперь новый алгоритм, позволяющий эффективно кодировать серии повторяющихся символов. Поступающую группу символов можно рассматривать как последовательность $x_1 \ldots x_t$. Обозначим через n_0 символ с максимальной вероятностью, а через $n_1 \ldots n_k$ оставшиеся маловероятные символы. Положим

$$\hat{p} = \sum_{k=1}^{m} p(n_k), \quad \hat{q} = 1 - \hat{p} = p(n_0), \tag{1}$$

где $p(n_k)$ $(1 \leqslant k \leqslant m)$ — вероятность появления символа n_k.

Тогда кодирование проводится в два этапа и осуществляется следующим образом. Рассмотрим первый этап кодирования. Последовательность $x_1 \ldots x_t$ разбиваем на блоки (подслова) длины $l = \lceil 1/\sqrt{\hat{p}} \rceil$, где $\hat{p}$ определяется формулой (1). Если блок состоит только из символов n_0, то его кодом является n_0. Если же блок содержит хотя бы один из символов n_k $(1 \leqslant k \leqslant m)$, то длина кодового слова равна $l+1$: началом этого кодового слова является любой символ n_k, встречающийся в данном блоке, за которым следует тот же самый блок длины l. На втором этапе полученная последовательность сжимается арифметическим кодом.

Например, пусть $A = \{a_1,a_2,a_3\}$, $p(a_1) = 6/7$, $p(a_2) = 2/21$, $p(a_3) = 1/21$ и кодируется последовательность $a_1a_1a_1a_1a_1a_1a_1a_1a_1a_1a_1a_2a_1a_1a_1a_2a_1a_3a_1a_1a_1$. В этой последовательности a_1 является высоковероятным символом n_0, а a_2, a_3 — маловероятными символами n_1, n_2. Из (1) следует, что $\hat{p} = 1/7$, $l = 3$ и закодированная последовательность имеет вид: $a_1a_1a_1a_2a_1a_1a_2a_1a_3a_2a_1a_3a_1$.

Пусть теперь $y_1y_2 \ldots y_s$ — последовательность символов, полученная после первого этапа кодирования, $y_i \in \{a_1,a_2,\ldots,a_n\}$. Рассмотрим второй этап кодирования этой последовательности, осуществляемый арифметическим кодом из [4, 5].

Опишем сначала алгоритм подсчета вероятностей символов последовательности $y_1y_2 \ldots y_s$. Для этого выделим в ней блоки длины l, которые следуют после появления маловероятного символа n_k, и символы n_0, n_k, не входящие в блоки, то есть представим последовательность $y_1y_2 \ldots y_s$ в виде: $n_0 \ldots n_0 n_k y_1 \ldots y_l n_0 \ldots n_0 n_k y_1 \ldots y_l \ldots$

Символы n_0 и n_k, не входящие в блоки, будем кодировать с вероятностями $\hat{q}^l$ и $1 - \hat{q}^l$ соответственно. Кодирование же символов внутри блока $y_1y_2 \ldots y_l$ длины l будем производить следующим образом. Пусть $y_1y_2 \ldots y_{i-1} = n_0 \ldots n_0$. Тогда символ y_i, находящийся в i-ой позиции после $i-1$ символов n_0, кодируется с вероятностями π_i^k для символа n_k и $\left(1 - \sum_{k=1}^{m} \pi_i^k\right)$ для символа n_0. Вычисление вероятностей π_i^k можно организовать по схеме:

$$\omega^k := \omega^k/\hat{q}, \quad \frac{1}{\hat{\pi}^k} := \frac{1}{\hat{\pi}^k} - \omega^k$$

с начальными данными
$$\omega^k := \frac{\hat{q}^l \cdot \hat{p}}{p(n_k)}, \quad \frac{1}{\hat{\pi}^k} := \frac{1-\hat{q}^l}{p(n_k)}.$$

Наконец, символы из блока $y_1 y_2 \dots y_l$, идущие после появления в этом блоке любого символа n_k, кодируются с исходными вероятностями $\hat{q}$ для символа n_0 и $p(n_k)$ для символа n_k.

Аналогично кодируются символы следующего блока $y_1' \dots y_l'$, причем перед каждым новым блоком начальные данные обновляются. Отметим, что время вычисления величин π_i^k не превышает $O\left(\log(\frac{1}{r\hat{p}}) \log\log(\frac{1}{r\hat{p}}) \log\log\log(\frac{1}{r\hat{p}})\right)$, где r — избыточность кодирования.

Рассмотрим теперь, как осуществляется арифметическое кодирование. Для этого сначала вычисляются суммарные вероятности

$$Q_0 = 0, \quad Q_k = \sum_{i=1}^{k} p(a_i), \tag{2}$$

где $p(a_i)$ — вероятности символов a_i, полученных после первого этапа кодирования. Затем полуинтервал $[Q_1, Q_k)$ разбивается так, чтобы каждой букве исходного алфавита соответствовал свой полуинтервал, пропорциональный ее вероятности. В процессе кодирования выбирается полуинтервал, соответствующий текущей букве кодируемого слова, который снова разбивается пропорционально вероятностям исходных букв. Таким образом происходит сужение исходного полуинтервала до тех пор, пока не будет получен последний символ кодируемого сообщения. Двоичное представление с определенной точностью любой точки, расположенной внутри полуинтервала, соответствующего последней букве кодируемого сообщения, будет кодом исходной последовательности.

Обозначим через l_i нижнюю границу полуинтервала, соответствующего i-ой букве исходной последовательности, через h_i — верхнюю границу этого полуинтервала, а через r_i — длину полуинтервала $[l_i, h_i)$. Определим начальные значения $l_0 = Q_0$, $h_0 = Q_k$, $r_0 = h_0 - l_0$.

Границы полуинтервала, соответствующего кодируемому символу, будем вычислять по следующим формулам:

$$l_i = l_{i-1} + r_{i-1} \cdot Q_{x_i - 1}, \quad h_i = l_{i-1} + r_{i-1} \cdot Q_{x_i - 1}, \tag{3}$$

где x_i — i-ый символ кодируемой последовательности $x_1 x_2 x_3 \dots$

Например, пусть $A = \{a_1, a_2, a_3, a_4\}$, $p(a_1) = 0.1$, $p(a_2) = 0.4$, $p(a_3) = 0.2$, $p(a_4) = 0.3$ и кодируется последовательность $a_3 a_2 a_3 a_1 a_4$. Из (2) находим, что $Q_0 = 0$, $Q_1 = 0.1$, $Q_2 = 0.5$, $Q_3 = 0.7$, $Q_4 = 1$. Начальные значения $l_0 = 0$, $h_0 = 1$, $r_0 = 1$. Из (3) определяем границы полуинтервала, соответствующие первому символу кодируемого сообщения: $l_1 = 0.5$, $h_1 = 0.7$, $r_1 = 0.2$. Дальнейший процесс сужения полуинтервала схематически можно изобразить следующим образом:

$$[0,1] \stackrel{a_3}{\to} [0.5, 0.7] \stackrel{a_2}{\to} [0.52, 0.6] \stackrel{a_3}{\to} [0.56, 0.576] \stackrel{a_1}{\to} [0.56, 0.5616] \stackrel{a_4}{\to} [0.56112, 0.5616].$$

Кодом последовательности $a_3 a_2 a_3 a_1 a_4$ будет двоичная запись любой точки из полуинтервала $[0.56112, 0.5616)$.

Отметим, что для однозначного декодирования исходной последовательности достаточно взять $\lceil \log r_n \rceil$ разрядов двоичной записи любой точки из полуинтервала $[l_i, h_i)$, где r_n — длина полуинтервала после кодирования n символов источника.

Таблица

Результаты сравнения работы алгоритмов RLE и RLE_new

	$k(RLE)$	$\boldsymbol{k(RLE\_new)}$	$t(RLE)$	$\boldsymbol{t(RLE\_new)}$
Image_1	2.56	**4.09**	3.1	**6.9**
Image_2	2.17	**3.26**	2.3	**5.1**
Image_3	1.21	**1.84**	2.6	**5.4**
Image_4	3.09	**4.85**	4.4	**9.3**
Image_5	2.73	**4.17**	3.9	**8.2**

3. Численные расчеты. В качестве эксперимента были взяты 5 различных растровых изображений с большим количеством одноцветных областей и размером 1024 на 1024 пикселов с глубиной цвета 24 бит, каждое из которых занимает около 3 Мб памяти. Данные изображения сжимались сначала стандартным алгоритмом RLE, а затем новым RLE_new. Сравнение проводилось по двум параметрам: коэффициенту сжатия k, которое рассчитывалось как отношение объема несжатых данных к сжатому объему, и времени кодирования и декодирования t (в секундах). Результаты сравнения приведены в таблице.

Из таблицы видно, что новый алгоритм RLE_new повышает коэффициент сжатия примерно в 1,5 раза при небольшом увеличении времени кодирования и декодирования.

Заключение. Предложенный метод кодирования позволяет более эффективно сжимать изображения для их передачи по каналам связи либо хранения в памяти компьютера. Алгоритм наиболее эффективен для изображений с большими одноцветными областями, например, деловой и научной графики.

Список литературы

1. Тропченко А. Ю, Тропченко А. А. Методы сжатия изображений, аудиосигналов и видео. СПб: СПбГУ ИТМО, 2009. 108 с.

2. Bell T. C., Moffat A., Witten I. H. Compressing the Digital Library // Proc. Digital Libraries — Texas: College Station, 1994. P. 41–46.

3. Jiawan Zh., Jizhou S., Zhigang S. Accelerate volume splatting by using run length encoding // Lecture Notes in Computer Science. 2003. V. 2657. P. 907–914.

4. Witten I. H., Neal R., Cleary J. G. Arithmetic coding for data compression // Comm. ACM. 1987. V. 30, N 6. P. 520–540.

5. Сэломон М. Сжатие данных, изображений и звука. М.: Техносфера, 2004. 368 с.

Бакулина Марина Павловна — канд. физ.-мат. наук, научный сотрудник Института вычислительной математики и математической геофизики (ИВМиМГ СО РАН). Научные интересы: теория кодирования, сжатие данных, моделирование информационных систем, e-mail: `marina@rav.sscc.ru`.

Bakulina Marina Pavlovna is a Researcher of Institute of Computational Mathematics and Mathematical Geophysics (ICMMG SB RAS). PhD in Physics and Mathematics. Research interests: coding theory, data compression, information systems modeling.

Дата поступления — 09.11.2023