УДК 681.3

В.П. КОЖЕМЯКО [a], Л.И. ТИМЧЕНКО [b], А.А. ЯРОВОЙ [a], И.Д. ИВАСЮК ,Н.И. КОКРЯЦКАЯ[a]

МЕТОДЫ ПИРАМИДАЛЬНОГО КОДИРОВАНИЯ ДЛЯ СЖАТИЯ ДАННЫХ

[a] Винницкий национальный технический университет
95, Хмельницкое шоссе, Винница, 21021, Украина

[b] Киевский университет экономики и технологий транспорта
19, ул. Лукашевича, Киев, 03049, Украина
E-mail: timchen@svitonline.com

Анотація. В статті розглядаються нові теоретичні підходи до техніки стиснення інформації на основі аналізу багаторівневого представлення даних на кожному рівні ієрархічної обробки. Представлені результати пропонуємого пірамідально-лінійного кодування свідчать про ефективність його практичної реалізації, оскільки такий тип кодування не передбачає побудову кодового дерева, відповідно немає необхідності додаткових витрат на перебалансування кодового дерева, що значно підвищує якість стиснення даних.

Аннотация. В статье рассматриваются новые теоретические подходы к технике сжатия информации, на основании анализа многоуровневого представления данных на каждом уровне иерархической обработки. Представленные результаты предлагаемого пирамидально-линейного кодирования свидетельствуют об эффективности его практической реализация, поскольку такой тип кодирования не предусматривает построения кодового дерева, соответственно не требуется дополнительных затрат на перебалансировку кодового дерева, что существенно улучшает качество сжатия данных.

Abstract. New theoretical approaches to engineering of the information compression, on the basis of analysis of the data multilevel representation at each level of hierarchical processing, are considered. The submitted results of offered pyramid-linear coding testify to efficiency its practical realization, as such type of coding does not provide construction of a code tree, accordingly it is not required of additional expenses on rebalancing of a code tree, that essentially improves quality of the data compression.

Ключевые слова: пирамидальное кодирование, параллельная обработка, кодовое дерево, сжатие данных.

ВВЕДЕНИЕ

За последние 30-40 лет техника сжатия данных применительно к информационным технологиям усовершенствовалась (повысился коэффициент сжатия) на 30-40 %. В современных информационных технологиях все больше актуализуется проблема создания новых концепций и подходов к технике сжатия информации.

В настоящее время в компьютерной технике сжатия используются статистические и словарные методы [1]. Из статистических методов наиболее распространенным является кодирование Шеннона-Фано и кодирование Хаффмена (статическое, динамическое и с блокированием), а из словарных - целое семейство алгоритмов, основанных на широко известном методе кодирования Лемпеля-Зива-Велча [2].

Можно отметить некоторые другие популярные методы сжатия, которые используются в компьютерных технологиях. Это арифметическое кодирование и кодирование на основе стопки книг. Проанализируем некоторые из них. Кодирование Хаффмена [3] имеет минимальную избыточность при условии, что каждый символ кодируется отдельной цепочкой в алфавите {0,1}. Его недостатком является зависимость степени сжатия от близости вероятности символов к отрицательной степени 2, что связано с кодированием каждого символа целым числом бит.

Арифметическое кодирование является методом, позволяющим упаковывать символы входного алфавита без потерь при условии, что известно распределение частот этих символов [4].

Концепция метода была изложена в 60-х годах в работах Элиаса. При реализации этого метода возникают две проблемы: во-первых необходима арифметика действительных чисел вообще говоря неограниченной точности, а во-вторых, результат кодирования становится известен лишь при окончании входного потока.

В основу словарных методов положено кодирование цепочек символов (LZ77 - compression) [5]. Суть его в следующем: упаковщик постоянно хранит некоторое количество последних обработанных символов в некотором буфере - скользящем словаре. По мере обработки входного потока вновь поступившие символы попадают в конец буфера, сдвигая предшествующие символы и вытесняя самые старые. Время сжатия при такой реализации пропорционально произведению длины входного потока на размер буфера, что непригодно для практического использования. Для улучшения процедуры быстрого поиска в словаре используется двоичное дерево, что позволило несколько повысить скорость работы [6].

В статье рассматриваются новые методы в технике сжатия, которые сводятся к идее анализа многоуровневого представления данных [7, 8, 9] на каждом уровне иерархии.

ПРИМЕНЕНИЕ ПИРАМИДАЛЬНОГО КОДИРОВАНИЯ ДЛЯ СЖАТИЯ ДАННЫХ

Рассмотрим более подробно на функциональном уровне организацию многоуровневого представления данных для реализации пирамидально-линейного кодирования. Пусть в информационном потоке данных n элементов есть k различных, где $p_1, p_2, \ldots, p_k$ - вероятность появления каждого элемента. Причем через p_1 обозначим вероятность наиболее часто встречающегося элемента, тогда

$$p_1 \geq p_2 \geq \ldots \geq p_i \geq \ldots \geq p_k . \qquad (1)$$

То есть расположим вероятности появления элементов в порядке не возрастания. Пирамидально-линейный метод кодирования состоит в следующем. Выбранный на каждом шаге преобразования элемент, кодируется единицей, а остальные - нулем. Поэтому, если в исходном потоке имеется n элементов, то и в первом столбце масок тоже будет n кодов-масок, из которых np_1 - единицы. Тогда во втором столбце масок уже будет $n - np_1$ кодов-масок, из которых np_1 - единицы. Таким образом, получается $k-1$ столбец масок, где количество элементов в каждом столбце будет:

Причем, в последнем столбце np_k - нулей, остальные единицы (т.е. np_{k-1}), и, следовательно, элемент с наименьшей частотой появления кодируется $k-1$ количеством нулей.

Найдем количество элементов в масках:

$$S = n + n - np_1 + n - np_1 - np_2 + n - np_1 - np_2 - np_3 + \cdots + n\left(1 - \sum_{i=1}^{k-2} p_i\right) = n + n(1 - p_1) + n(1 - (p_1 + p_2)) +$$

$$+ n(1 - (p_1 + p_2 + p_3)) + \ldots + n\left(1 - \sum_{i=1}^{k-2} p_i\right) = n(1 + 1 - p_1 + 1 - p_1 - p_2 + 1 - p_1 - p_2 - p_3 + \ldots +$$

$$+ 1 - p_1 - p_2 - p_3 - p_4 - \ldots - p_{k-2}) = n\left((k-1) - (k-2)p_1 - (k-3)p_2 - (k-4)p_3 - \ldots - 2p_{k-3} - p_{k-2}\right) =$$

$$= n\left((k-1-k+2)p_1 + (k-1-k+3)p_2 + (k-1-k+4)p_3 + \ldots + (k-1-2)p_{k-3} + (k-1-1)p_{k-2} +\right.$$

$$\left. + (k-1)p_{k-1} + (k-1)p_k\right) = n\left(p_1 + 2p_2 + 3p_3 + \ldots + (k-3)p_{k-3} + (k-2)p_{k-2} + (k-1)p_{k-1} + (k-1)p_k\right) =$$

$$= n\left(\sum_{i=1}^{k-1} ip_i + (k-1)p_k\right).$$

Таблица 1

Оценка числа элементов столбцов-масок для ПЛК

1 столбец	2 столбец	3 столбец	⋯	$k-1$ столбец
n	$n - np_1$	$n - np_1 - np_2$	⋯	$n\left(1 - \sum_{i=1}^{k-2} p_i\right)$

Если через A_n^k обозначить входной поток из n символов по k различных, *M* - столбец-маска, а

A_{p_i} - элемент с вероятностью $p_i, i=\overline{1,k}$, то в общем виде схему пирамидально-линейного метода кодирования можно записать в таком виде:

$$A_n^k = A_{p_1}\left(M_n^1\right) + A_{p_2}\left(M_{n(1-p_1)}^2\right) + \ldots + A_{p_{k-1}}\left(M_{n\left(\sum_{i=1}^{k-2} p_i\right)}^{k-1}\right) + A_{p_k}\left(M_{n\left(1-\sum_{i=1}^{k-2} p_i\right)}^{k-1}\right). \quad (2)$$

Процесс пирамидально-линейного кодирования (ПЛК) можно ускорить в 2 раза, если на каждом уровне анализировать пары соседних символов с вероятностями p_i и p_{i+1}. Тогда процесс ПЛК становится параллельным и его можно представить в виде кодового дерева (рис. 1) и записать схему кодирования в таком виде:

$$A_n^k = \left(A_{p_1} M_n^1 + A_{p_2} M_{n(1-p_1)}^1\right) + \left(A_{p_3} M_{n(1-p_1-p_2)}^2 + A_{p_4} M_{n\left(1-\sum_{i=1}^{3} p_i\right)}^2\right) + \ldots + \left(A_{p_{k-1}} M_{n\left(1-\sum_{i=1}^{k-2} p_i\right)}^{k/2} + A_{p_k} M_{n\left(1-\sum_{i=1}^{k-1} p_i\right)}^{k/2}\right).$$

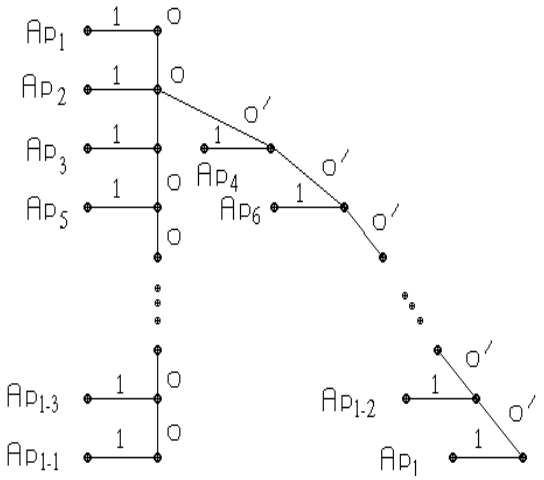

Рис. 1. Граф-схема кодового дерева ПЛК

На рис. 1 $l = 2k, \quad k \in N, \quad 0' = 00$.

Анализ выражения (2) и кодового дерева (рис. 1) показывает, что по сравнению с известными способами статистического кодирования [1, 3] за счет параллельной организации схемы ПЛК можно добиться существенного уменьшения времени сжатия. Так, для первой схемы ПЛК это время определяется путем анализа всех символов входного алфавита, а для второй схемы ПЛК - их половиной.

В то же время известные статистические способы ориентированы на последовательную выборку элементов в соответствии с вероятностями их появления в потоке данных. В этом случае время сжатия определяется длиной входного потока данных, а не длиной их алфавита.

То есть учтено замечание, касающееся последнего элемента с наименьшей вероятностью p_k.

Справедливо следующее выражение:

$$S = n\left(\sum_{i=1}^{k-1} i p_i + (k-1)p_k\right).$$

Проанализируем эффективность сжатия при реализации ПЛК. Сравним количество кодов-масок в масках-столбцах с исходным потоком информации, т.е. S и $n\log_2 k$. Тогда коэффициент сжатия R:

$$R = \frac{\log_2 k}{\left(\sum_{i=1}^{k-1} i p_i + (k-1)p_k\right)}. \quad (3)$$

Оценим, при каком условии $R > 1$, т.е. условие, при котором данный метод сжатия эффективен:

$$\frac{\log_2 k}{\left(\sum_{i=1}^{k-1} i p_i + (k-1)p_k\right)} > 1, \text{ тогда}$$

$$\log_2 k > \sum_{i=1}^{k-1} ip_i + (k-1)p_k . \log_2 k > p_1 + 2p_2 + 3p_3 + \ldots + (k-1)p_{k-1} + (k-1)p_k . \quad (4)$$

Используя метод математической индукции, выведем условие сжатия.

1. При $k = 3$, тогда $\log_2 3 > p_1 + 2p_2 + 3p_3$, $\log_2 3 > 1 + p_2 + p_3$, $\log_2 3 > 2 - p_1$, $p_1 > 2 - \log_2 3$.
2. Пусть формула (4) верна при k различных элементов, т.е.
$(k-2)p_1 + (k-3)p_2 + (k-4)p_3 + \ldots + 2p_{k-3} + p_{k-2} > (k-1) - \log_2 k..$

Докажем ее верность при $k+1$ элементе:

$$\log_2(k+1) > p_1 + 2p_2 + 3p_3 + \ldots + kp_k + kp_{k+1},$$
$$\log_2(k+1) > 1 + p_2 + 2p_3 + \ldots + (k-1)p_k + (k-1)p_{k+1},$$
$$\log_2(k+1) > 1 + 1 - p_1 + p_3 + \ldots + (k-2)p_k + (k-2)p_{k+1},$$
$$\log_2(k+1) > 2 - p_1 + (1 - p_1 - p_2) + \ldots + (k-2)p_k + (k-2)p_{k+1},$$
$$\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots$$
$$\log_2(k+1) > k - (k-1)p_1 - (k-2)p_3 - \ldots - 3p_{k-3} - 2p_{k-2} - p_{k-1},$$
$$(k-1)p_1 + (k-2)p_2 + \ldots + 3p_{k-2} + p_{k-1} > k - \log_2(k+1),$$
$$(k-2)p_1 + (k-3)p_2 + \ldots + p_{k-2} + 1 - p_k > k - \log_2(k+1),$$
$$(k-2)p_1 + (k-3)p_2 + \ldots + 2p_{k-3} + p_{k-2} > (k-1) - \log_2(k+1) + p_k,$$

где величина $p_k - \log_2(k+1)$ сравнима с $-\log_2 k$ при $k \to \infty$, а $p_k \to 0$.

Значит $\sum_{i=1}^{k-2} (k - (i+1))p_i > (k-1) - \log_2 k$ - является условием сжатия для ПЛК.

Качественно условие сжатия можно сформулировать так: двоичный логарифм от количества различных элементов и исходного потока данных должен быть больше суммы произведений вероятности элемента на его порядковый номер выборки (выборка осуществляется по не возрастанию вероятностей).

Анализ выражения (4) показывает, что эффективность метода ПЛК по сжатию данных сравнима с семейством методов сжатия на основе кодирования Хаффмена.

В отличие от методов кодирования Хаффмена, где необходима организация довольно сложной процедуры построения кодового дерева, для ПЛК его строить нет необходимости, хотя формальное построение его возможно.

Указанное преимущество особенно ценно при использовании адаптивного кодирования, для которого необходимо постоянная корректировка дерева в соответствии с изменяющейся статистикой входного потока. При реализации это требует значительных расходов на перебалансировку кодового дерева в соответствии с новыми частотами символов на каждом шаге.

Так как реализация ПЛК не предусматривает построения кодового дерева, то при использовании адаптивного кодирования не требуется дополнительных затрат на перебалансировку кодового дерева, что существенно улучшает качество сжатия.

Рассмотрим процесс пирамидально-нелинейного кодирования. Также как в предыдущем случае его можно определить как многоуровневый, состоящего из нескольких иерархических уровней. Их количество зависит от числа k (числа различных элементов) и в зависимости от вероятности появления элементов $p_1, p_2, \ldots, p_k$ изменяется в промежутке $1, 2, \ldots, k$.

На первом уровне элементов, сумма вероятностей которых ≈ 0.5, сортируются в порядке убывания вероятности. Эти элементы кодируются единицей, все остальные - нулем. Таким образом, множество всех элементов разбивается на два подмножества. На следующем уровне раздельно для каждого подмножества процедура полностью повторяется, т.е. процесс происходит рекурсивно, пока не появится подмножество, состоящее только из одного элемента.

Пусть на первом шаге сумма вероятностей первых l (считая от элемента с наибольшей вероятностью) будет около 0.5. Очевидно, что $l = 1, 2, \ldots, \frac{k}{2}$; причем $l = 1$ при $p_1 \approx 0.5$ и $l = \frac{k}{2}$, при

$p_1 = p_2 = \ldots = p_k$.

Поэтому, если в исходном потоке имеется n элементов, то в первом столбце-масок тоже будет n кодов-масок, из которых $n(p_1, p_2, \ldots, p_l)$ - единиц и $n\left(1 - \sum_{i=1}^{l} p_i\right)$ - нулей.

На втором шаге формируется два подмножества элементов:

I-ое из $n\sum_{i=1}^{l} p_i$ - элементов, II-ое из $n\sum_{i=l+1}^{k} p_i$ - элементов.

Для каждого из этих подмножеств, если $l \neq 1$, процедура описания для первого шага кодирования полностью повторяется и т.д., до тех пор пока не будут получены подмножества, состоящие из одного элемента.

Обозначим через l_2^2 количество элементов на втором уровне, вероятности которых (считая от элемента с наибольшей вероятностью) в подмножестве элементов, закодированных на предыдущем уровне нулем, будут около 0.5. На этом уровне эти элементы кодируются единицей. Обозначим через l_2^1 количество элементов на втором уровне, вероятности которых (считая от элемента с наибольшей вероятностью) в подмножестве элементов, закодированных на предыдущем уровне единицей будут около 0.5.

Тогда для $l_2^1 = 0,1,2,\ldots\frac{l}{2}$, причем $l_2^1 = 0$ при $l = 1$, $l_2^1 = 1$ при $p_i \approx 0.25$ и $l_2^1 = \frac{l}{2}$ при $p_1 = p_2 = \ldots = p_l$. Аналогично, для $l_2^2 = l+1, l+2, \ldots, {(k+1)}/{2}$, причем $l_2^2 = 0$ при $k = 1$, $l_2^2 = 1$ при $p_{l+1} \approx 0.25$ и $l_2^2 = {(k+1)}/{2}$ при $p_{l+1} = p_{l+2} = \ldots = p_k$. Поэтому во втором столбце количество единиц:

$$n\left(1 - n\sum_{i=l+1}^{l} p_i\right)\sum_{i=1}^{l_2^2} p_i = n\sum_{i=l+1}^{k} p_i \sum_{i=l+1}^{l_2^2} p_i \ . \qquad (5)$$

Соотношение (5) справедливо для подмножества элементов, закодированных на предыдущем уровне нулем.

$$n\sum_{i=1}^{l} p_i \left(p_1 + p_2 + \ldots + p_{l_2^1}\right) = n\sum_{i=1}^{l} p_i \sum_{i=1}^{l_2^1} p_i \ . \qquad (6)$$

Тогда соотношение (6) справедливо для подмножества элементов, закодированных на предыдущем уровне единицей.

Таким образом, на втором уровне получено четыре подмножества элементов:

I-ое: $n\sum_{i=1}^{l} p_i \sum_{i=1}^{l_2^1} p_i$; II-ое: $n\sum_{i=1}^{l} p_i \sum_{i=l_2^1+1}^{l} p_i$; III-ье: $n\sum_{i=l+1}^{k} p_i \sum_{i=l+1}^{l_2^2} p_i$; -ое: $n\sum_{i=l+1}^{k} p_i \sum_{i=l_2^2+1}^{k} p_i$.

В общем случае число l_x^y - число элементов с вероятностью p_i для каждого уровня, где x - номер уровня, $y = \overline{1,\ldots,2^{x-1}}$.

Причем, если y - нечетное, то это означает, что на предыдущем уровне данные элементы были закодированы единицей, если четное - то на предыдущем уровне данные элементы были закодированы нулем. Тогда для третьего столбца образуются 8 подмножеств элементов.

I-ое: $n\sum_{i=1}^{l} p_i \sum_{i=1}^{l_2^1} p_i \sum_{i=1}^{l_3^1} p_i$, II-ое: $n\sum_{i=1}^{l} p_i \sum_{i=1}^{l_2^1} p_i \sum_{i=l_3^1+1}^{l_2^1} p_i$, III-ье: $n\sum_{i=1}^{l} p_i \sum_{i=l_2^1+1}^{l} p_i \sum_{i=l_2^1+1}^{l_3^2} p_i$,

IV-ое: $n\sum_{i=1}^{l} p_i \sum_{i=l_2^1+1}^{l} p_i \sum_{i=l_3^2+1}^{l} p_i$, V-ое: $n\sum_{i=l+1}^{k} p_i \sum_{i=l+1}^{l_2^2} p_i \sum_{i=l+1}^{l_3^3} p_i$, VI-ое: $n\sum_{i=l+1}^{k} p_i \sum_{i=l+1}^{l_2^2} p_i \sum_{i=l_3^3+1}^{l_2^2} p_i$,

VII-ое: $n\sum_{i=l+1}^{k} p_i \sum_{i=l_2^2+1}^{k} p_i \sum_{i=l_2^2+1}^{l_3^4} p_i$, VIII-ое: $n\sum_{i=l+1}^{k} p_i \sum_{i=l_2^2+1}^{k} p_i \sum_{i=l_3^4+1}^{k} p_i$,

где $l_3^1 = 0,1,\ldots,\frac{l_2^1}{2};\ l_3^2 = l_2^1+1, l_2^1+2,\ldots,\frac{\left(l+l_2^1\right)}{2};\ l_3^3 = l+1, l+2,\ldots,\frac{l_2^2}{2};$

$l_3^4 = l_2^2+1, l_2^2+2,\ldots,\frac{\left(k+l_2^2\right)}{2}.$

Тогда для $p_1 \geq p_2 \geq \ldots \geq p_k$ можно записать количественную оценку числа элементов столбцов-масок, которая отражена в табл. 2.

Таблица 2

Оценка числа элементов столбцов-масок для пирамидально-нелинейного кодирования

№ столбца	1	2	3	⋯	j
Число элементов	M_1	M_2	M_3	⋯	M_j

$$M_1 = \begin{pmatrix} n\sum_{i=1}^{l} p_i \\ n\left(1-\sum_{i=1}^{l} p_i\right) \end{pmatrix}. \qquad M_2 = n\begin{pmatrix} \sum_{i=1}^{l} p_i \\ \sum_{i=1}^{l} p_i \\ \sum_{i=l+1}^{k} p_i \\ \sum_{i=l+1}^{k} p_i \end{pmatrix}^T \begin{pmatrix} \sum_{i=1}^{l_2^1} p_i \\ \sum_{i=l_2^1+1}^{l} p_i \\ \sum_{i=l+1}^{l_2^2} p_i \\ \sum_{i=l_2^2+1}^{k} p_i \end{pmatrix}.$$

Аналогично в матричной форме выглядит M_3 и несложно записать выражение для M_j.

$$M_j = n\begin{pmatrix} 2^{j-1}\left\{\sum_{i=1}^{l} p_i \\ \cdots \right. \\ 2^{j-1}\left\{\sum_{i=l+1}^{k} p_i \\ \cdots \right. \end{pmatrix}^T \begin{pmatrix} 2^{j-2}\left\{\sum_{i=1}^{l_2^1} p_i \\ \cdots \right. \\ 2^{j-2}\left\{\sum_{i=l_2^1+1}^{l} p_i \\ \cdots \right. \\ 2^{j-2}\left\{\sum_{i=l+1}^{l_2^2} p_i \\ \cdots \right. \\ 2^{j-2}\left\{\sum_{i=l_2^2+1}^{k} p_i \\ \cdots \right. \end{pmatrix} \begin{pmatrix} \sum_{i=1}^{l_j^1} p_i \\ \cdots \\ \sum_{i=l_j^1+1}^{l_{j-1}^2} p_i \\ \cdots \\ \sum_{i=l_{j-1}^1+1}^{l_j^{j-1}} p_i \\ \cdots \\ \sum_{i=l_j^{j-1}+1}^{l} p_i \\ \cdots \\ \sum_{i=l+1}^{l_j^j} p_i \\ \cdots \\ \sum_{i=l_{j-1}^{j-1}+1}^{l_j^{j-1}} p_i \\ \cdots \\ \sum_{i=l_j^j+1}^{l_j^{j+1}} p_i \\ \cdots \\ \sum_{i=l_{j-1}^{j-1}+1}^{l_j^{j+1}} p_i \\ \cdots \\ \sum_{i=l_j^{j+1}+1}^{k} p_i \end{pmatrix}.$$

Таким образом, количество элементов S в столбцах-масках находится в промежутке:

$$n\left(\sum_{i=1}^{k-1} ip_i + (k-1)p_k\right) \le S \le n\log_2 k,$$

а коэффициент сжатия R:

$$\frac{\log_2 k}{\sum_{i=1}^{k-1} ip_i + (k-1)p_k} \le R \le 1.$$

Причем, S и R принимают предельные значения:

1) если $p_1, p_2, \ldots, p_k$ - образуют ряд убывающей геометрической прогрессии;

2) если $p_1 = p_2 = \ldots = p_k$.

А модель кодирования в матричной форме тогда имеет вид:

$$A_n^k = \sum_{j=1}^{k}\sum_{i=1}^{k} A_{p_i} M_{p_i}^{n_j},$$

где n_j - число элементов на j-ом уровне, $n_1 = n$; M_{p_i} - матрица-столбец масок на j-ом уровне элемента с вероятностью p_i.

Метод рекомендуется применять для сжатия таких сигналов, как видеоизображения, которые можно представить как марковские источники низкого порядка, а также текстовой информации со сравнительно небольшим словарем.

Приведем выражение, позволяющее рассчитывать число тактов для полной записи (считывания) в памяти размерностью $H \cdot m$ слов при реализации пирамидального кодирования. Число тактов записи (чтения) для этого случая определяется по формуле:

$$T^{вет} = m - r + L, \tag{7}$$

где r - количество одинаковых слов в массиве; L - количество групп с одинаковыми словами. Для обычной последовательной памяти запись (считывание) двумерного массива из $H \cdot m$ слов потребует $H \cdot m$ тактов:

$$T^{CT} = H \cdot m. \tag{8}$$

Сравнивая формулы (7) и (8) видно, что память с реализацией пирамидального кодирования массива информации является более быстродействующей по сравнению с обычной памятью для кодирования данных на величину $r - L$.

Для записи в память информации об одном массиве необходим объем памяти:

$$O^{вет} = T^{вет} \cdot n + T^{вет} \cdot m = T^{вет} \cdot (n+m) = (m - r + L)(m+n). \tag{9}$$

Для обычной памяти для кодирования данных необходим объем:

$$O^{ст} = m \cdot n. \tag{10}$$

Вычитая (9) из (10), получаем :

$$P = O^{ст} - O^{вет} = r \cdot (n+m) - L \cdot (n+m) - m^2. \tag{11}$$

Из (11) следует, что если

$$\begin{aligned} &r \cdot (n+m) - L \cdot (n+m) - m^2 = 0, \quad \text{то } O^{ст} = O^{вет}, \\ &r \cdot (n+m) - L \cdot (n+m) - m^2 > 0, \quad \text{то } O^{ст} > O^{вет}, \\ &r \cdot (n+m) - L \cdot (n+m) - m^2 < 0, \quad \text{то } O^{ст} < O^{вет}. \end{aligned} \tag{12}$$

Из (12) следует, что объем памяти при реализации пирамидального кодирования будет тем меньше, чем выше значения r и n, и меньше значения L и m.

Эффективность сокращения объема памяти можно оценить следующим соотношением

$$R = \frac{O^{cm}}{O^{вет}} = \frac{mn}{(m - r + L)(n + m)}.$$

Рассмотрим конкретный пример: пусть имеется некоторый массив данных с такими параметрами $m = 15, n = 8, r = 10$ и $L = 4$:

$$T^{вет} = 15 - 10 + 4 = 9\tau, \quad T^{cm} = 15\tau,$$
$$O^{вет} = 9(15 + 8) = 207 \text{бит}, \quad O^{cm} = 15 \times 8 = 120 \text{ бит}.$$

Для массива той же размерности, но с другими параметрами $m = 15, n = 8, r = 13$ и $L = 3$:

$$T^{вет} = 15 - 13 + 3 = 5\tau, \quad T^{cm} = 15\tau,$$
$$O^{вет} = 5(15 + 8) = 115 \text{бит}, \quad O^{cm} = 15 \times 8 = 120 \text{ бит}.$$

В первом случае быстродействие памяти при реализации пирамидального кодирования выше стандартного метода записи данных, но объем больше. Во втором случае и быстродействие выше и объем памяти меньше стандартной. В общем случае число тактов $T^{вет}$ записи (считывания) информации при реализации пирамидального кодирования находится в пределах

$$\tau \le T^{вет} \le T^{cm}.$$

Исследованная организация памяти при реализации пирамидального кодирования относится к вычислительным структурам не Неймановского типа, ориентированных на параллельную и компактную обработку информационных полей данных, что позволяет производить преобразование в реальном масштабе времени.

ВЫВОДЫ

1. В отличие от различных модификаций метода кодирования Хаффмена, для которых необходима организация довольно сложной процедуры построения кодового дерева, для пирамидально-линейного метода кодирования его строить нет необходимости, хотя формальное построение его возможно. Указанное преимущество особенно ценно при использовании адаптивного кодирования, для которого необходимо постоянная корректировка дерева в соответствии с изменяющейся статистикой входного потока. При программной реализации это требует значительных расходов на перебалансировку кодового дерева в соответствии с новыми частотами символов на каждом шаге.
2. Так как реализация пирамидально-линейного метода кодирования не предусматривает построения кодового дерева, то при использовании адаптивного кодирования не требуется дополнительных затрат на перебалансировку кодового дерева, что существенно улучшает качество сжатия. Условием сжатия для пирамидально-линейного метода кодирования является - $\sum_{i=1}^{k-2}(k - (i+1))p_i > (k-1) - \log_2 k$, а для пирамидально-нелинейного кодирования - $\frac{\log_2 k}{\sum_{i=1}^{k-1} i p_i + (k-1)p_k} \le R \le 1$.
3. Возможность дополнительного сжатия масок позволяет улучшить упаковку данных, что возможно при выполнении определенных граничных условий.
4. Анализ методов сжатия на основе многоуровневого подхода показывает, что по сравнению с известными способами статистического кодирования за счет параллельной организации схемы пирамидально-линейного метода кодирования можно добиться существенного уменьшения времени сжатия. Так, для первой схемы пирамидально-линейного способа кодирования это время определяется путем анализа всех символов входного алфавита, а для второй - их половиной.

СПИСОК ЛИТЕРАТУРЫ

1. Кричевский Р.Е. Сжатие и поиск информации. - М. - Радио и связь. - 1989.
2. Ziv I., Lempel A. Compression of individual sequences via variablerate coding // IEEE Trans. Inf. Theory IT - 24. - 1978. - №5, PP. 530 - 536.
3. Huffman D.A. A method for the construction of minimum - redundancy codes // Proc. Inst. Electr. Radio Eng. - 40. - 1952. - №9, PP. 1984-1101.
4. Witten I.H., Neal R.M., Cleary I.G. Arithmetic coding for data compression //Commun. ACM. - 30. - 1987. - №6, PP. 520 - 540.
5. Welch T.E. Atechnique for highperfomance data compression //IEEE Comput. - 17. - 1984. - № 6, pp. 8-19.
6. Bell T.C. IEEE Trans. COM - 34. - 1986, PP. 1176-1182.
7. L.I. Timchenko, Y.F. Kutaev, S.V. Chepornyuk, M.A. Grudin, D.M. Harvey, A.A. Gertsiy. A Brain - Like Approach to Multistage Hierarchial Image. Springer-Verlag Processing. - *in Proc. Image Analysis and Processing*, Florence, Italy, Septemger 17 - 19, 1997, PP. 246 - 253.
8. Кожем'яко В.П., Тимченко Л.І., Кутаев Ю.Ф., Івасюк І.Д. Вступ в алгоритмічну теорію ієрархії і паралелізму нейроподібних обчислювальних середовищ та її застосування до перетворення зображень. Основи теорії пірамідально-сітьового перетворення зображень. К: УМК ВО, 1994, 272 с.
9. V.P. Kozhemyako, L.I. Tymchenko, Yu.F. Kutaev, A.A. Yaroviy Approach for real–time image recognition. // Оптико-електронні інформаційно-енергетичні технології. – 2001 р. – №1. – С. 110-124.

Надійшла до редакції 08.12.2004 р.

КОЖЕМЯКО В.П. – академик АИНУ, д.т.н., профессор, заведующий кафедрой лазерной и оптоэлектронной техники, Винницкий национальный технический университет, Винница, Украина.

ТИМЧЕНКО Л.И. – д.т.н., профессор, заведующий кафедрой автометрии и цифровых систем передачи, Киевский университет экономики и технологий транспорта, Киев, Украина.

ЯРОВОЙ А.А. – к.т.н., ст. преподаватель кафедры интеллектуальных систем, Винницкий национальный технический университет, Винница, Украина.

ИВАСЮК И.Д. – преподаватель математики и информатики, директор Винницкой общеобразовательной школы I-III ст. №30, Винница, Украина.

КОКРЯЦКАЯ Н.И. – ассистент кафедры прикладной математики, Винницкий национальный технический университет, Винница, Украина.