Министерство образования и науки РФ

Севастопольский государственный университет

Институт информационных технологий и управления в технических системах

Лабораторная работа №2

«Исследование динамических методов сжатия данных: Хаффмана, Лемпеля-Зива (LZ), Лемпеля-Зива-Велча ( LZW)»

Выполнил:

ст.гр. ИСб-22д

Воронин И.Ю.

Проверил:

Маслова М.А

Севастополь

2015

1.Цель работы

Исследование динамических методов сжатия данных: Хаффмана, Лемпеля-Зива (LZ), Лемпеля-Зива-Велча ( LZW), RLE

2.Вариант задания

Составить текстовое сообщение на естественном языке, содержащее 15-25 символов, среди которых должны быть повторяющиеся.

2. Составить таблицу первичного алфавита источника на основе сообщения п.1 (модель Бернулли ) с указанием частоты встречаемости символов.

3. Рассчитать информационные параметры источника и сообщения: количество информации, энтропию, избыточность.

4. Закодировать сообщение п.1 двоичным безизбыточным кодом.

5. Закодировать сообщение п.1 динамическим кодом Хаффмана.

6. Закодировать сообщение п.1 кодом LZ.

7. Закодировать сообщение п.1 кодом LZW.

8. Рассчитать среднюю длину символа сообщения и определить коэффициенты сжатия для методов п.п.4-7.

9. Декодировать сообщения п.п. 4-7.

10. Составить пример черно-белого бинарного изображения размером 8х8 пикселей и закодировать его по методу RLE.

11. Сделать выводы по работе

Вариант 99

Вышел месяц из тумана.

3.Краткая теория

Кол-во информации: I = n\*∑pi\*log21/pi

Энтропия: H = ∑pi\*log21/pi

Избыточность: D= 1- H/log2m

4.Ход работы

|  |  |  |
| --- | --- | --- |
| Пр | 3 | 00000 |
| Е | 2 | 00001 |
| А | 2 | 00010 |
| М | 2 | 00011 |
| Л | 1 | 00100 |
| В | 1 | 00101 |
| Ы | 1 | 00110 |
| С | 1 | 00111 |
| Я | 1 | 01000 |
| Ц | 1 | 01001 |
| И | 1 | 01010 |
| З | 1 | 01011 |
| Т | 1 | 01100 |
| У | 1 | 01101 |
| М | 1 | 01110 |
| Ш | 1 | 01111 |
| Н | 1 | 10000 |

Всего 22 символ. Вес сообщения при двоичном безизбыточным коде 110 бит.

* Динамический метод Хаффмана









Рисунок 4.1-Дерево Хаффмана.

Результат:

До: Вышел месяц из (при кодировании в ASCII=120 бит)

После: Вышел м1001сяц100001из11(Хаффман =108 бит)

* **Метод сжатия LZ77**

Рассмотрим на примере фразы «Кот ломом колол слона»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Скользящее окно | | | Выходной код | | |
| Словарь | Буфер(5) | Совп. | i | j | s |
|  | кот\_л |  | 1 | 0 | к |
| К | от\_ло |  | 1 | 0 | о |
| Ко | т\_лом |  | 1 | 0 | т |
| Кот | \_ломо |  | 1 | 0 | \_ |
| кот\_ | ломом |  | 1 | 0 | л |
| кот\_л | омом\_ | о | 4 | 1 | м |
| кот\_лом | ом\_ко | ом | 2 | 2 | \_ |
| кот\_ломом\_ | колол | ко | 10 | 2 | л |
| кот\_ломом\_кол | ол\_сл | ол | 2 | 2 | \_ |
| кот\_ломом\_колол\_ | слона |  | 1 | 0 | с |
| кот\_ломом\_колол\_с | лона | ло | 5 | 2 | н |
| кот\_ломом\_колол\_слон | а |  | 1 | 0 | а |
| кот\_ломом\_колол\_слона |  |  | 0 |  |  |

Таблица 4.1-LZ77.

В данном случае для кодирования i достаточно 5 бит, а для j 3 бит, когда в свою очередь для представления им понадобится байт(8 бит). У нас 12 строк, значит вес будет равен 12\*(5+3+8)=192 бита.

В обычном случае это занимало бы память при 21 символе и 8 байтов веса для него 168 бит.

Это доказывает то, что коэффициент сжатия у LZ77 больше единицы(1,26)

**Метод LZW**

Изначально необходимо заполнить таблицу словаря единичными символами без повторов. Символы буду принимать четырёхразрядные двоичные коды.(табл.4.2)

|  |  |
| --- | --- |
| Словарь | Символ |
| 0 | к |
| 1 | о |
| 2 | т |
| 3 | \_ |
| 4 | л |
| 5 | м |
| 6 | с |
| 7 | н |
| 8 | а |

Таблица 4.2 – LZW словарь.

Добавляем к пустой строке символ ‘к’ и проверим, есть ли данная строка “к” в таблице. Так как этот символ был занесён в таблицу при инициализации, то получаем код 0. Далее считываем следующий символ ‘о’ из входного потока и проверяем есть ли строка “ко” в таблице. Данной строки нет, следовательно, запоминаем её под кодом 9. Следуя этому алгоритмы получим таблицу (4.3).

На выходном потоке получаем сообщение:

0 1 2 3 4 1 5 15 3 9 13 4 3 6 13 7 8

В данном словаре есть числа, для кодирования которых необходимо 5 разрядов. Однако на выходном потоке все числа можно закодировать и при помощи 4-рязрядного двоичного числа. Получаем:

* при 4 разрядах вес = 16 \*4 = 64 бит;

процент сжатия = 58 %

* при 5 разрядах вес = 16 \*5 = 80 бит;

процент сжатия = 72 %

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Текущая строка | Тек.символ | След.символ | Вывод | | Словарь |
| Код | Биты |
| ко | К | о | 0 | 0000 | 9 : ко |
| от | О | т | 1 | 0001 | 10 : от |
| т\_ | Т | \_ | 2 | 0010 | 11 : т\_ |
| \_л | \_ | л | 3 | 0011 | 12 : \_л |
| ло | Л | о | 4 | 0100 | 13 : ло |
| ом | О | м | 1 | 0001 | 14 : ом |
| мо | М | о | 5 | 0101 | 15 : мо |
| ом | О | м | ----- | ------ | -------------- |
| ом\_ | М | \_ | 15 | 1111 | 16 : ом\_ |
| \_к | \_ | к | 3 | 0011 | 17 : \_к |
| ко | к | о | ----- | ------ | -------------- |
| кол | о | л | 9 | 1001 | 18 : кол |
| ло | л | о | ----- | ------ | -------------- |
| лол | о | л | 13 | 1101 | 19 : лол |
| л\_ | л | \_ | 4 | 0100 | 20 : л\_ |
| \_с | \_ | с | 3 | 0011 | 21 : \_с |
| сл | с | л | 6 | 0110 | 22 : сл |
| ло | л | о | ----- | ------ | -------------- |
| лон | о | н | 13 | 1101 | 23 : лон |
| на | н | а | 7 | 0111 | 24 : на |
| а | а | ------------- | 8 | 1000 | -------------- |

Таблица 4.3 – Результат LZW кодирования.

* **RLE сжатие изображений**

Пусть имеется чёрно-белое изображение 8х8 пикселей. (рис.4.2)

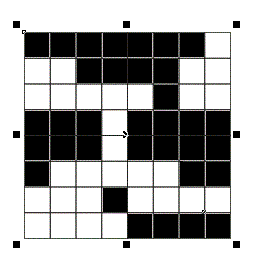


Рисунок 4.2 – Тестовое изображение.

Если у нас будет встречаться чёрный квадрат, то это будет обозначаться цифрой 1, то есть TRUE, в противном случае, когда квадрат белый – 0 (FALSE).

Всего у нас 64 ячейки, следовательно, нам необходимо 6 разрядов, так как 26 = 64. Старший бит будет использовать под обозначение света по выше указанному обозначению. Таким образом, нам хватит для кодирования 7-разрядного двоичного числа.

Передаваться будет следующая последовательность бит (табл.4.4).

Таким образом при декодировании, зная разрешение изображения и условие кодировки, мы будет заполнять пикселе как при обычном последовательном обходе матрицы.

|  |  |  |
| --- | --- | --- |
| Цвет | Кол-во повторений | Код |
|  | 7 | 1000111 |
|  | 3 | 0000011 |
|  | 4 | 1000100 |
|  | 7 | 0000111 |
|  | 1 | 1000001 |
|  | 2 | 0000010 |
|  | 3 | 1000011 |
|  | 1 | 0000001 |
|  | 7 | 1000111 |
|  | 1 | 0000001 |
|  | 5 | 1000101 |
|  | 5 | 0000101 |
|  | 2 | 1000010 |
|  | 3 | 0000011 |
|  | 1 | 1000001 |
|  | 8 | 0001000 |
|  | 4 | 1000100 |

Таблица 4.4 - RLE сжатие изображения.

ВЫВОДЫ

В данной лабораторной работе были исследованы динамические методы кодирования. Кодировались как изображения, так и сообщения. Был произведён анализ каждого из метода на удобных для этого тестовых примерах.