УДК 51.37

Масин Н.В
специальность 09.03.03 Прикладная информатика в экономике Московский Технологический Институт
Россия, г. Москва
Сохранский В.В.
специальность 09.03.03 Прикладная информатика в экономике Московский Технологический Институт Россия, г. Москва

ИЗУЧЕНИЕ АЛГОРИТМА СЖАТИЯ ХАФФМАНА

Аннотация: *Изучить алгоритм оптимального префиксного кодирования Хаффмана и его использование для сжатия сообщений. Обработка кодов с рассчитанными сложением подач байтов; но поскольку все в моментальном упрощённом виде. Рассчитаем коэффициент сжатия относительно использования кодировки ASCII (8 бит/символ). С использованием веточных узлов в разновидных массивов. А также с построением с низа до вершены, добавляем существенные узлы с низкими весами.*

Ключевые слова: *алгоритм, Хаффман, дерево кодирования, веточные узлы, бинарный код.*

Masin N.V, 09.03.03 Applied Informatics,
Applied Computer Science in Economics Moscow Technological Institute
Russia, Moscow
Sokhransky V.V. 09.03.03 Applied Informatics,
Applied Computer Science in Economics Moscow Technological Institute
Russia, Moscow

LEARNING PROCESS USING THE HUFFMAN ALGORITHM

Annotation: Study the Huffman Optimal Prefix Coding algorithm and its use for message compression. Processing codes with calculated addition of byte feeds; but because everything is in an instantly simplified form. Let's calculate the compression ratio relative to the use of ASCII encoding (8 bits/character). Also build the binary coding tree bottom-up, by combining existing nodes that have the lowest weights.

Key words: algorithm, Huffman, coding tree, branch nodes, binary numbers.

Алгоритм Хаффмана — адаптивный жадный алгоритм оптимального префиксного кодирования алфавита с минимальной избыточностью. Был разработан в 1952 году аспирантом Массачусетского технологического института Дэвидом Хаффманом A Method for the Construction of Minimum Redundancy Codes [Proceedings of the IRE 40 (9): 1098–1101 (1952)]. С использованием постоянной частности для сжатие данных по Хаффману применяется при сжатии фото и видеоизображений (JPEG, стандарты сжатия MPEG), в архиваторах (PKZIP,LZH и др.), в протоколах передачи данных MNP5 и MNP7. Жадный алгоритм (Greedy algorithm) — алгоритм, заключающийся в принятии локально оптимальных решений на каждом этапе, допуская, что конечное решение также окажется оптимальным. Идея алгоритма Хаффмана состоит в следующем: зная вероятности символов в

сообщении, можно описать процедуру построения кодов переменной длины, состоящих из целого количества битов. Символам с большей вероятностью ставятся в соответствие более короткие коды. Коды Хаффмана обладают свойством префикса (то есть ни одно кодовое слово не является префиксом другого), что позволяет однозначно их декодировать. Нетрудно доказать с помощью метода индукции по m, что этот способ действительно позволяет минимизировать взвешенную длину пути [См. E. S. Schwartz, Information and Control 7 (1964), 37-44; G. Markowsky, Acta Informatica 16 (1981), 363-370.] Это значит, что существует прекрасный способ поиска дерева Хаффмана при условии, что веса расположены в порядке не убывания. Тогда достаточно создать две очереди, одна из которых будет содержать исходные веса, а другая — объединенные веса. На каждом этапе этой процедуры наименьший неиспользованный вес будет находиться в начале одной из очередей, поэтому его не придется искать. Размер веточного узла равен соответственному потоку алфавита; вес продлившего узла равен равный суммой сортировок. С другой страны, для продливших узлов которые выделяет сам индекс для дерево кодирования, и последовательно начинаются с 1.

Чтобы разделить веточные узлы и продливший узлы, с использованием знаком индекса. То бишь, позитивный индекс обозначается как продливший узел; там где негативный индекс равен веточному (алфавитному) узлу. Современное кодирование Хаффмана требует передачу резонансов алфавита. Это указано для кодированного алфавита в Таблице №1.

Таблица 1. **Коммуникационная таблица резонансов**

Поле	**Вес**	**Действие**
original_size	sizeof(int)	Количество байтов в исходыым
num_active	1 байт	Количество активных алфавитов
Алфавитный поток	1 байт + sizeof(int)	Первый код дает бинарный код алфавита

Современное кодирование Хаффмана требует передачу резонансов алфавита. Это указано для кодированного алфавита. Обработка кодов с рассчитанными сложением подач байтов; но поскольку все в моментальном упрощённом виде. Эти файлы с алфавитами могут быть модулированы предусмотрены в верхней таблице. Сам процесс имплементации может достигнуть самый минимальный уровень. Максимальное число узлов бинарного дерево кодов состоявшийся на число активных алфавитов. Для самой функций алфавитов, идет локализация с ее количество битов и парных совпадающих индексных узлов. Дале, для каждого узла в дереве кодирования, мы также парные сортировки. Этот процесс очень нуждается в том что когда идет установка дерево веточные узлы могут продвигаться в правую часть ветки чтобы не сниженный вес мог быть продвинутым в потоке данных, а не последовательным. Дале индексного веточного узла имеет свойства,

$$(v\ +\ 1)$$

где v является ёмкость и вес алфавита. Дополняем единицу для того, чтобы индекс не был равен нолю. Первые изображения были отправлены по проводам еще в 1843 г., но современные факсы в офисах так и не появились. до 1960-х годов. В то время для отправки одностраничного письма по телефонным линиям общего пользования требовалось около шестиминутный

стандарт группы 1 для передачи, который был введен международной телеграфной и телефонной связью. Стандарт группы 2, введенный в 1976 г. сократил время до отправить страницу за три минуты, но все же не смог обеспечить передачу в достаточно плотном разрешении для четкое воспроизведение мелкого шрифта. В 1980 году был введен стандарт группы 3. Стандарт группы 3 улучшен разрешение сканирования факса и внедрили методы цифровой передачи, обеспечивающие скорость передачи 14400 бит в секунду. Содержание узлов, которые является часть бинарных узлов, которые состоят из комбинации новых узлов, как сам алгоритм Хаффмана представляет.

Возьмем слово "abracadbra" где φ символ конец сообщение. Получаем данные в Таблице №2;

Таблица 2 **Визуальная таблица алгоритма**

Байт	**Символ**	**Резонанс**
10	φ	1
97	a	5
98	b	2
99	c	1
100	d	1
114	r	2

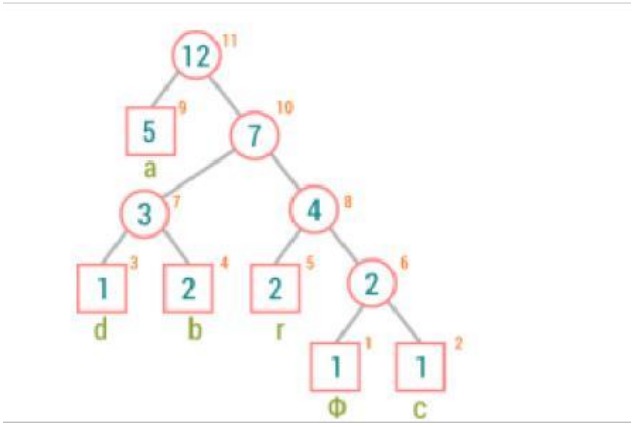

Рисунок 1. Бинарное дерево

В Рисунке.1 показывает установку бинарного кодированного дерева. Пусть v — внутренний узел, который находится на максимальном расстоянии от корня. Если веса φ и с еще не приписаны детям узла V, то ими можно заменить величины, которые уже там находятся, не увеличивая взвешенную длину пути. Содержание узлов, которые является часть бинарных узлов, которые состоят из комбинации новых узлов, как сам алгоритм Хаффмана представляет.

Рассчитаем коэффициент сжатия относительно использования кодировки ASCII (8 бит/символ).

$ASCII$ = 8 · 12 = 96 бит $^{L}Huff$ = 6 · 2+3 · 3+2 · 2 · 3+2 · 2 · 4+8 · 5= 89 бит.

$$K_{\text{сж}} = {}^{L}\!/_{L}\,\cancel{ASCII} \approx 1.078\ Huff$$

Коэффициент сжатия относительно равномерного кода (5 бит/символ, т. к. у нас всего 12 символов) будет равен;

$$\text{К}_{\text{сж}} = \frac{5 \cdot 12}{L} = \frac{60}{89} \approx 0.674\ Huff$$

Рассчитаем среднюю длину полученного кода по формуле;

$$l_{cp} = \sum p \cdot l_s,\ s$$

где S — множество символов алфавита; p — вероятность появления символа; l_S — количество бит в коде символа. Для полученного кода средняя длина будет равна

$$l_{cp} = 0.24 \cdot 2 + 0.12 \cdot 3 + 2 \cdot 0.08 \cdot 3 + 2 \cdot 0.08 \cdot 4 + 8 \cdot 0.01 \cdot 5 =$$

2.36 бит/символ.

Поскольку при построении дерева кода Хаффмана может возникнуть некоторый п ного варианта кода используют дисперсию. Дисперсия показывает насколько сильно отклоняются длины индивидуальных кодов от их средней величины. Лучшим будет код с наименьшей дисперсией. Дисперсия рассчитывается по формуле:

$$\delta = \sum p \cdot (l_s - l_{cp})^2$$

Для полученного кода дисперсия будет равна;

$$\delta = 0.24 \cdot (2 - 2.36)^2 + 0.12 \cdot (3 - 2.36)^2 + 2 \cdot 0.08 \cdot (3 - 2.36)^2 + 2 \cdot 0.08 \cdot (4 - 2.36)^2 + 8 \cdot 0.01 \cdot (5 - 2.36)^2 = 1.1337$$

Для уменьшения дисперсии кода существует правило: когда на дереве имеется более двух узлов с наименьшей вероятностью, следует объединять символы с наибольшей и наименьшей вероятностью; это сокращает общую дисперсию кода.

Использованные источники:

1. D.A. Huffman, “A method for the construction of minimumredundancy codes,” Proceedings of the IRE, vol. 40, no. 9, pp. 1098–1101, Sep. 1952.
2. Я. Гэгэрин. Имплементация Хаффмана 2018 – Кембридж. URL: https://github.com/NikolayTach/huffman/blob/master/huffman(rus).pdf