И. А. Валежанин

СЖАТИЕ ДАННЫХ ПО АЛГОРИТМУ ХАФФМАНА

Описывается реализация алгоритма Хаффмана на языке C++ для сжатия данных, а также рассматриваются преимущества и недостатки алгоритма в сравнении с другими.

Введение

Основная идея алгоритма заключается в построении таблицы кодирования,которая присваивает уникальный код каждому символу исходного текста. При декодировании полученная таблица кодирования используется для преобразования данных в их исходную форму.

I. Алгоритм Хаффмана

Процесс кодировки на Хаффману начинается с подсчета частот встречаемости символов в исходном файле. Затем эти частоты используются для построения дерева Хаффмана(см. рис.1.), в котором символы с наименьшей частотой имеют более короткий код, а символы с более высокой частотой имеют более длинный код.

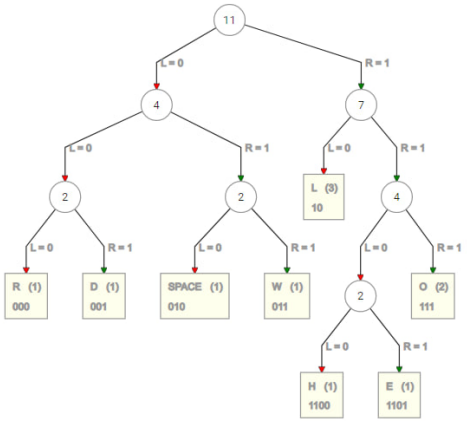

Рис. 1 – Дерево Хаффмана

Коды Хаффмана строятся по принципу "один символ - один код что обеспечивает однозначность декодирования. Закодированный файл содержит дерево Хаффмана и закодированные данные, которые занимают меньше места, чем исходные данные. При декодировании используется дерево Хаффмана, чтобы раскодировать закодированные данные и восстановить исходный файл.

II. Реализация и тесты алгоритма на C++

Для построения дерева Хаффмана на C++ подсчитывается частота встречаемости символов в данных и создаётся приоритетная очередь, отсортированная по частоте. Затем элементы с наименьшей частотой извлекаются, объединяются и добавляются в очередь, пока не останется только корень дерева. Для кодирования данных используется готовое дерево. Для каждого символа находится соответствующий код Хаффмана в виде битовой последовательности. Для декодирования данных нужно использовать дерево и читать битовый поток. При сравнении данного алгоритма с другими были получены следующие результаты:

Таблица 1 – Для данных размером 35-40 байт

	Степень сжатия	Время разархива
Haffman	75.20	1мс
7zip	420.52	4мс
nanozip	32.06	50мс
Bzip2	197.34	1мс

Таблица 2 – Для данных размером 2 Кбайа

	Степень сжатия	Время разархива
Haffman	65.18	27мс
7zip	33.19	11мс
nanozip	33.36	50мс
Bzip2	65.18	27мс

В каждой из приведенных таблиц степень сжатия означает, какой процент от исходного размера файла составляет размер сжатого файла. Полученные результаты указывают на то, что сжатие по алгоритму Хаффмана эффективно при работе с небольшими данными. Это связано с тем, что при работе с такими данными дерево Хаффмана может быть построено быстро и занимает меньше памяти по сравнению с деревьями, построенными для больших наборов данных.

III. Выводы

Алгоритм Хаффмана - мощный инструмент для сжатия данных, который может быть использован для уменьшения размеров файлов без потери качества. Полученные результаты показывают, что алгоритм Хаффмана является достаточно эффективным методом сжатия данных.

1. Huffman D. // A Method for the Construction of Minimum-Redundancy Codes. – 1952.

Валежанин Илья Александрович, студент кафедры инфокоммуникационных технологий БГУИР, ilavalezanin@gmail.com.

Научный руководитель: Кукин Дмитрий Петрович, Заведующий кафедрой вычислительных методов и программирования БГУИР, кандидат технических наук, доцент, kukin@bsuir.by.