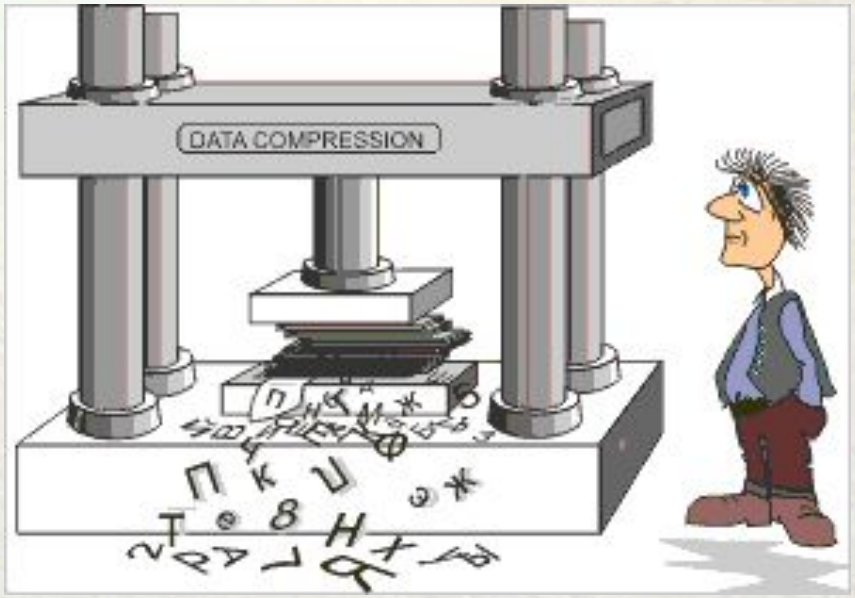
Практическая работа

Алгоритм Хаффмана

Сжатие информации



Сжатие данных – сокращение объема данных при сохранении закодированного в них содержания.

Сжатие информации

Сжатие происходит за счет устранения избыточности кода, например, за счет упрощения кодов, исключения из них постоянных битов или представления повторяющихся символов в виде коэффициента повторения.

Важнейшая характеристика процесса сжатия – коэффициент сжатия.

Коэффициент сжатия – отношение объема исходного сообщения к объему сжатого.

Алгоритмы сжатия

1. Равномерное сжатие с использованием кодов одной длины.

Этот метод используется, если в записи сообщения присутствует небольшая часть алфавита.

2. Сжатие с использованием кодов переменной длины.

Сокращение объёма данных достигается за счёт замены часто встречающихся данных короткими кодовыми словами, а редких — длинными.

Сжатие с использованием кодов переменной длины

В этом случае возникает проблема отделения кодов символов друг от друга.

Решить эту проблему позволяет условие, достаточное для однозначного декодирования сообщений с переменной длиной кодовых слов, *условие Фано*:

Никакое кодовое слово не является началом другого кодового слова.

По-другому условие Фано называют *свойством префиксности*, а код, удовлетворяющий этому условию, называют *префиксным кодом*.

Префиксные коды

Чтобы понять, как строятся префиксные коды, рассмотрим, как построить ориентированный граф, определяющий этот код.

Например, кодовые слова 00, 01, 10, 011, 100, 101, 1001, 1010, 1111, кодируют соответственно буквы: *a*, *b*, *c*, *d*, *e*, *f*, *g*, *h*, *i*.

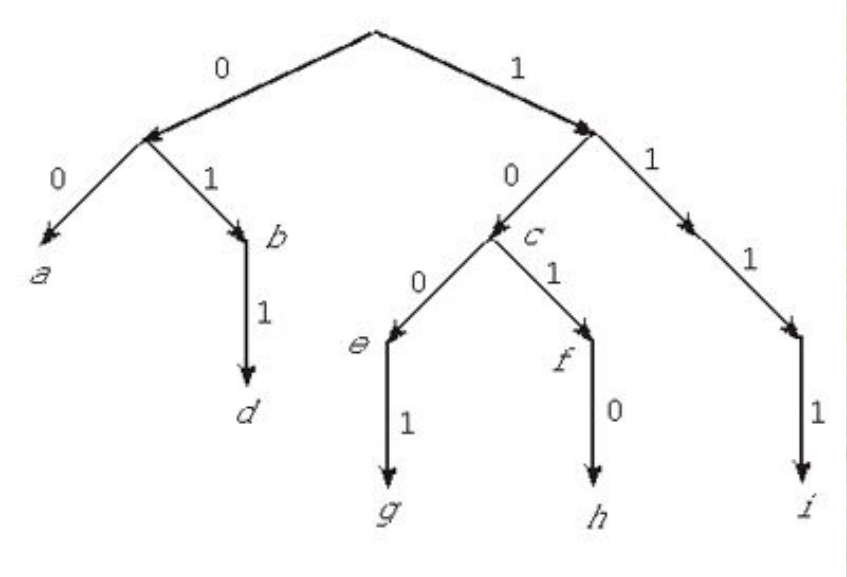
Префиксные коды

Построим граф этого кода.

Из начальной вершины выходят две дуги, помеченные 0 и 1. Затем из конца каждой такой дуги входят новые дуги, помеченные 0 и 1 так, чтобы, идя по этим дугам от корня, читалось начало какого-либо кодового слова.

Префиксные коды

Если при этом какое-то последовательность оказывается прочитанным полностью, то у конца последней дуги пишется кодируемый символ.



Из получившихся вершин снова проводятся дуги — и так далее, до тех пор, пока не будут исчерпаны все коды.

Префиксные коды

Если известен граф, созданный по префиксному коду, то по этому графу легко восстанавливается код каждого символа – надо просто, идя от корня к листу, помеченному данным символом, выписать 0 и 1 в порядке их прочтения.

Идея префиксного кодирования была использована американским ученым Д. Хаффманом для создания эффективного алгоритма сжатия символьной информации.

Алгоритм Хаффмана

Алгоритм Хаффмана — адаптивный алгоритм оптимального префиксного кодирования алфавита с минимальной избыточностью.

Был разработан 1952 году аспирантом Массачусетского технологического института Дэвидом Хаффманом при написании им курсовой работы. В настоящее время используется во многих программах сжатия данных.

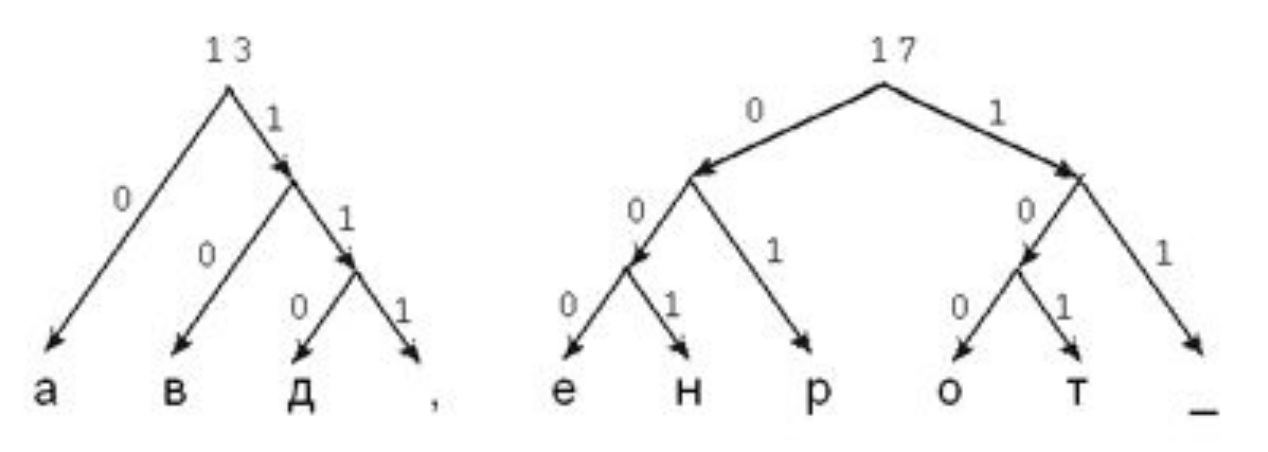
1. Символы исходного алфавита образуют вершины. Вес каждой вершины вес равен количеству вхождений данного символа в сжимаемое сообщение.
2. Среди вершин выбираются две с наименьшими весами (если таких пар несколько, выбирается любая из них).
3. Создается следующая вершина графа, из которой выходят две дуги к выбранным вершинам; одна дуга помечается цифрой 0, другая — символом 1.

Вес созданной вершины равен сумме весов, выбранных на втором шаге вершин.

4. К новым вершинам применяются шаги 2 и 3 до тех пор, пока не останется одна вершина с весом, равным сумме весов исходных символов.

НА_ДВОРЕ_ТРАВА,_НА_ТРАВЕ_ДРОВА

а	в	д	,	е	н	р	о	т	_
6	4	2	1	2	2	4	2	2	5



Составим таблицу кодов символов:

а	в	д	,	е	н	р	о	т	_
00	010	0110	0111	1000	1001	101	1100	1101	111
6	4	2	1	2	2	4	2	2	5

Найдем объем сообщения после кодирования кодом Хаффмана: 2·6 + 3·4 + 4·2 + 4·1 + 4·2 + 4·2 + 3·4 + 4·2 + 4·2 + 3·5 = 95 бит.

Теперь подсчитаем объем этого сообщения, если каждый его символ кодировать цепочкой из 0 и 1 равной длины. Т.к. в сообщении 10 различных символов вес одного символа 4 бита. Поэтому после кодирования получится сообщение объемом 4·3 = 120 бит.

Коэффициент сжатия равен 120/95 =1,26.

Сообщение в памяти компьютера закодировано с помощью ASCII-кодов, каждый символ весит 8 бит. Значит, объем исходного сообщения 240 бит.

Коэффициент сжатия равен 240/95 = 2,53.

Математики доказали, что среди алгоритмов, кодирующих каждый символ по отдельности и целым количеством бит, алгоритм Хаффмана обеспечивает наилучшее сжатие.

Пример 1. Используя код Хаффмана, закодировать следующий текст, состоящий из 29 знаков:

WENEEDMORESNOWFORBETTERSKIING

Используя табл. 1.8, закодируем строку:

011101 100 1100 100 100 11011 00011 1110 1011 100 0110 1100 1110 011101 01001 1110 1011 011100 100 001 001 100 1011 0110 110100011 1010 1010 1100 00001

После размещения этого кода в памяти побайтно, он примет вид:

01110110 01100100 10011011 00011111 01011100 01101100 11100111 01010011 11010110 11100100 00100110 01011011 01101000 11101010 10110000 001

Декодирование

Для декодирования можно воспользоваться построенным деревом. Начинаем с корня дерева и в качестве текущего бита берем начало текста.

В цикле делаем следующее

1) Смотрим, какое значение у текущего бита, и идем в низ по дереву по дуге, на которой указано такое же значение.

2) Переходим к следующему биту.

3) Если сейчас находимся в листе дерева: читаем символ находящийся в этом листе и записываем его в результат декодирования, переходим снова в корень дерева.

Пример 2. Раскодировать следующий двоичный код, полученный по алгоритму Хаффмана (пробелами код разделен на байты):

01010001 00100101 00100011 11111100

Двигаясь по дереву Хаффмана, начиная от первого слева разряда, получим следующую расшифровку:

0101 → H; 00010 → U; 01001 → F; 01001 → F;
00011 → M; 1111 → A; 1100 → N.

Получилось слово HUFFMAN. Упакованный код занимал 4 байта, исходный код — 7 байтов. Следовательно, коэффициент сжатия был равен $4/7 \approx 0{,}57$.

Практическая работа
Алгоритм Хаффмана

Цель: закрепить знания о сжатии текстовой информации с помощью алгоритма Хаффмана.

Ход работы

Задание 8, 9 с. 208 учебника.

Домашнее задание

- Повторить п. 6 с.43-44