Алгоритмы сжатия

Множество различных алгоритмов сжатия данных без потерь подразделяются на несколько основных групп.

1.Кодирование повторов (RLE – Run-Length Encoding).

2.Вероятностные методы сжатия. К ним относятся алгоритмы Шеннона-Фано и Хаффмена. В основе этих методов лежит идея построения «дерева», в котором положение символа на ветвях определяется частотой его появления. Каждому символу присваивается код, длина которого обратно пропорциональна частоте появления этого символа.

3.Арифметические методы. В результате арифметического кодирования строка символов заменяется действительным числом больше нуля и меньше единицы.

4.Метод словарей. Алгоритм, положенный в основу метода словарей, был впервые описан в работах

Алгоритм Шеннона — Фано

Алгоритм использует коды переменной длины: часто встречающийся символ кодируется кодом меньшей длины, редко встречающийся — кодом большей длины.

Коды Шеннона — Фано ***префиксные***, то есть никакое кодовое слово не
является префиксом любого другого.
Это свойство позволяет однозначно декодировать любую последовательность кодовых слов.

Основные этапы

1. Символы первичного алфавита выписывают в порядке убывания вероятностей.

2. Символы полученного алфавита делят на две части, суммарные вероятности символов которых максимально близки друг другу.

3. В префиксном коде для первой части алфавита присваивается двоичная цифра «0», второй части — «1».

4. Полученные части рекурсивно делятся и их частям назначаются соответствующие двоичные цифры в префиксном коде.

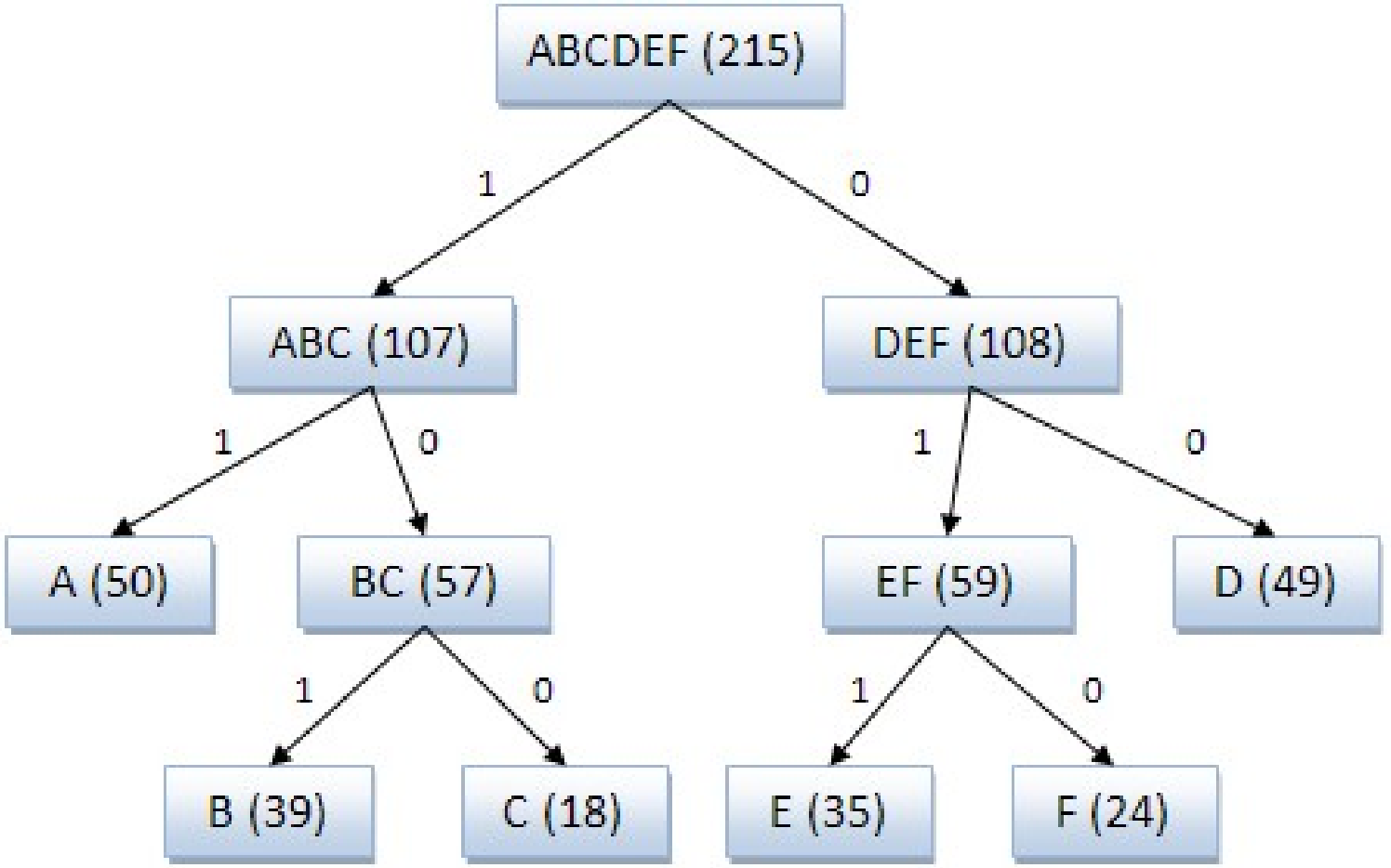
Алгоритм вычисления кодов Шеннона — Фано

Код Шеннона — Фано строится с помощью **дерева**. Построение этого дерева начинается от корня. Всё множество кодируемых элементов соответствует корню дерева (вершине первого уровня). Оно разбивается на два подмножества с примерно одинаковыми суммарными вероятностями. Эти подмножества соответствуют двум вершинам второго уровня, которые соединяются с корнем. Далее каждое из этих подмножеств разбивается на два подмножества с примерно одинаковыми суммарными вероятностями. Если подмножество содержит единственный элемент, то ему соответствует концевая вершина кодового дерева; такое подмножество разбиению не подлежит. Подобным образом поступаем до тех пор, пока не получим

Пример кодового дерева

Исходные символы:

A (частота встречаемости 50)
B (частота встречаемости 39)
C (частота встречаемости 18)
D (частота встречаемости 49)
E (частота встречаемости 35)
F (частота встречаемости 24)



Полученный код:
A — 11, B — 101, C — 100, D — 00, E — 011, F — 010.

Алгоритм Хаффмана

Алгоритм Хаффмана — адаптивный жадный алгоритм оптимального префиксного кодирования алфавита с минимальной избыточностью. Был разработан в 1952 году аспирантом Массачусетского технологического института Дэвидом Хаффманом при написании им курсовой работы.

В отличие от алгоритма Шеннона — Фано, алгоритм Хаффмана остаётся всегда оптимальным и для вторичных алфавитов с более чем двумя символами.

Этот метод кодирования состоит из двух основных этапов:

1.Построение оптимального кодового дерева.

2.Построение отображения код-символ на

Кодирование Хаффмана

Идея алгоритма 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.

Классический алгоритм Хаффмана на входе получает таблицу частот встречаемости символов в сообщении. Далее на основании этой таблицы строится дерево кодирования Хаффмана (H-дерево).

1. Символы входного алфавита образуют список свободных узлов. Каждый лист имеет вес, который может быть равен либо вероятности, либо количеству вхождений символа в сжимаемое сообщение.
2. Выбираются два свободных узла дерева с наименьшими весами.
3. Создается их родитель с весом, равным их суммарному весу.
4. Родитель добавляется в список свободных узлов, а два его потомка удаляются из этого списка.
5. Одной дуге, выходящей из родителя, ставится в соответствие бит 1, другой — бит 0.
6. Шаги, начиная со второго, повторяются до тех пор, пока в списке свободных узлов не останется только один свободный узел. Он и будет считаться корнем дерева.

A	B	C	D	E
10	5	8	13	10

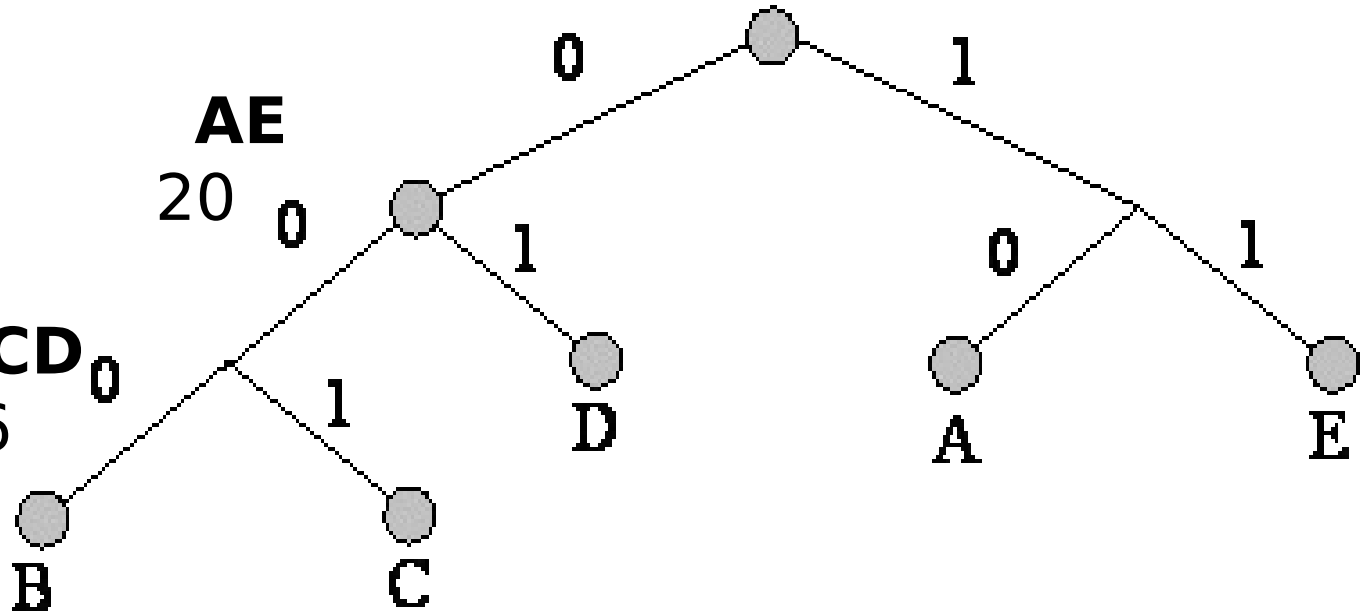
B	C	A	E	D
5	8	10	10	13

A	E	BC	D
10	10	13	13

BC	D	AE
13	13	20

AE	BCD
20	26

AEBCD



Заголовок

Полученный код: A — 11, B — 101, C — 100, D — 00, E — 011, F — 010.