ТЕМА 2.1.2 СЖАТИЕ ДАННЫХ

Сжатие данных

Алгоритмическое преобразование данных, производимое с целью уменьшения занимаемого ими объёма

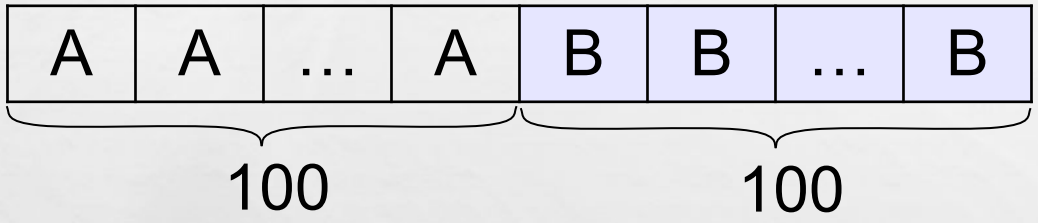
КОЭФФИЦИЕНТ СЖАТИЯ

СООТНОШЕНИЕ ИСХОДНОГО И СЖАТОГО ФАЙЛА

АЛГОРИТМ RLE

КОДИРОВАНИЕ ЦЕПОЧЕК ОДИНАКОВЫХ СИМВОЛОВ

Файл qq.txt



200 байтов

Файл qq.rle (сжатый)

100	A	100	B

4 байта

КОДИРОВАНИЕ ШЕННОНА — ФАНО

АЛГОРИТМ ПРЕФИКСНОГО НЕОДНОРОДНОГО КОДИРОВАНИЯ. ОТНОСИТСЯ К ВЕРОЯТНОСТНЫМ МЕТОДАМ СЖАТИЯ.

ИСПОЛЬЗУЕТ ИЗБЫТОЧНОСТЬ СООБЩЕНИЯ, ЗАКЛЮЧЁННУЮ В НЕОДНОРОДНОМ РАСПРЕДЕЛЕНИИ ЧАСТОТ СИМВОЛОВ ЕГО (ПЕРВИЧНОГО) АЛФАВИТА, ТО ЕСТЬ ЗАМЕНЯЕТ КОДЫ БОЛЕЕ ЧАСТЫХ СИМВОЛОВ КОРОТКИМИ ДВОИЧНЫМИ ПОСЛЕДОВАТЕЛЬНОСТЯМИ, А КОДЫ БОЛЕЕ РЕДКИХ СИМВОЛОВ — БОЛЕЕ ДЛИННЫМИ ДВОИЧНЫМИ ПОСЛЕДОВАТЕЛЬНОСТЯМИ.

ОСНОВНЫЕ ЭТАПЫ АЛГОРИТМА ШЕННОНА — ФАНО

1. СИМВОЛЫ ПЕРВИЧНОГО АЛФАВИТА M1 ВЫПИСЫВАЮТ ПО УБЫВАНИЮ 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.

ПРИМЕР КОДОВОГО ДЕРЕВА

ИСХОДНЫЕ СИМВОЛЫ:

A (ЧАСТОТА ВСТРЕЧАЕМОСТИ 50)

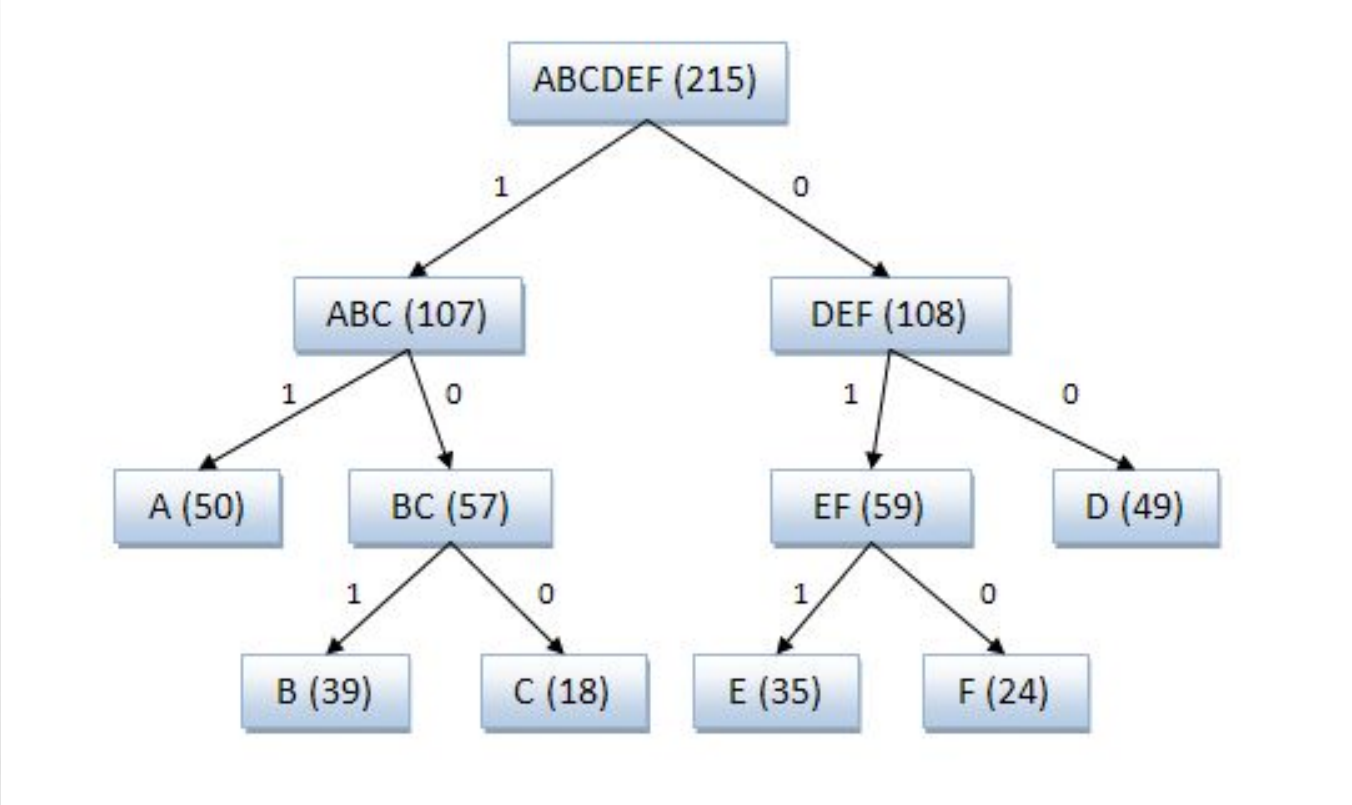
B (ЧАСТОТА ВСТРЕЧАЕМОСТИ 39)

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.

КОД ШЕННОНА-ФАНО

ДОСТОИНСТВА И НЕДОСТАТКИ

- учитывается частота символов
- не нужен символ-разделитель
- код префиксный – можно декодировать по мере поступления данных

- нужно заранее знать частоты символов
- код неоптимален
- при ошибке в передаче сложно восстановить «хвост»
- не учитывает повторяющиеся последовательности символов

АЛГОРИТМ ХАФФМАНА

АЛГОРИТМ ОПТИМАЛЬНОГО ПРЕФИКСНОГО КОДИРОВАНИЯ АЛФАВИТА С МИНИМАЛЬНОЙ ИЗБЫТОЧНОСТЬЮ

ЭТОТ МЕТОД КОДИРОВАНИЯ СОСТОИТ ИЗ ДВУХ ОСНОВНЫХ ЭТАПОВ:

1. ПОСТРОЕНИЕ ОПТИМАЛЬНОГО КОДОВОГО ДЕРЕВА.
2. ПОСТРОЕНИЕ ОТОБРАЖЕНИЯ КОД-СИМВОЛ НА ОСНОВЕ ПОСТРОЕННОГО ДЕРЕВА.

КЛАССИЧЕСКИЙ АЛГОРИТМ ХАФФМАНА НА ВХОДЕ ПОЛУЧАЕТ ТАБЛИЦУ ЧАСТОТ ВСТРЕЧАЕМОСТИ СИМВОЛОВ В СООБЩЕНИИ. ДАЛЕЕ НА ОСНОВАНИИ ЭТОЙ ТАБЛИЦЫ СТРОИТСЯ ДЕРЕВО КОДИРОВАНИЯ ХАФФМАНА (Н-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. БИТОВЫЕ ЗНАЧЕНИЯ ВЕТВЕЙ, ИСХОДЯЩИХ ОТ КОРНЯ, НЕ ЗАВИСЯТ ОТ ВЕСОВ ПОТОМКОВ.
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.

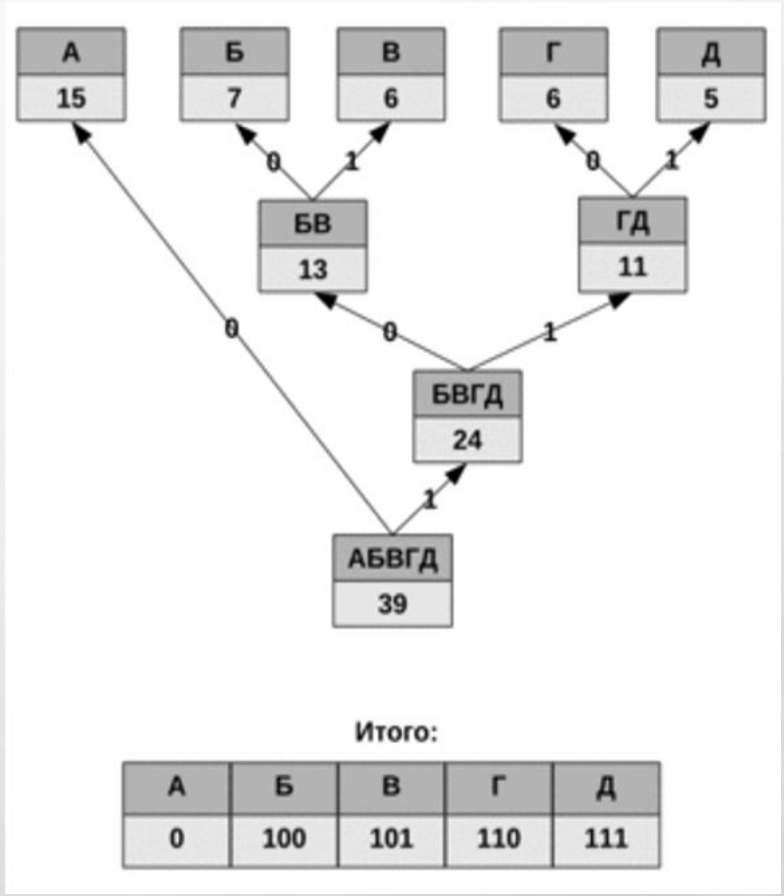
ПРИМЕР

ИСХОДНЫЕ СИМВОЛЫ:

Символ	А	Б	В	Г	Д
Частота	15	7	6	6	5

Полученный код:

Символ	А	Б	В	Г	Д
Код	0	100	101	110	111



АЛГОРИТМ ХАФФМАНА ДОСТОИНСТВА И НЕДОСТАТКИ

- код оптимальный среди алфавитных кодов

- нужно заранее знать частоты символов
- при ошибке в передаче сложно восстановить «хвост»
- не учитывает повторяющиеся последовательности символов

АЛГОРИТМ LZW

Этот метод позволяет достичь одну из наилучших степеней сжатия среди других существующих методов сжатия графических данных, при полном отсутствии потерь или искажений в исходных файлах. В настоящее время используется в файлах формата TIFF, PDF, GIF, PostScript и других, а также отчасти во многих популярных программах сжатия данных (ZIP, ARJ, LHA).

КОДИРОВАНИЕ

Начало.

Шаг 1. Все возможные символы заносятся в словарь. Во входную фразу X заносится первый символ сообщения.

Шаг 2. Считать очередной символ Y из сообщения.

Шаг 3. Если Y — это символ конца сообщения, то выдать код для X, иначе:

Если фраза XY уже имеется в словаре, то присвоить входной фразе значение XY и перейти к **Шагу 2** ,

Иначе выдать код для входной фразы X, добавить XY в словарь и присвоить входной фразе значение Y. Перейти к **Шагу 2.**

Конец.

ПРИМЕР

Пусть мы сжимаем последовательность: **abacabadabacabae**

Текущая строка	Текущий символ	Следующий символ	Вывод	Словарь	
			Код		
ab	a	b	0	5:	ab
ba	b	a	1	6:	ba
ac	a	c	0	7:	ac
ca	c	a	2	8:	ca
ab	a	b	-	-	-
aba	b	a	5	9:	aba
ad	a	d	0	10:	ad
da	d	a	3	11:	da
ab	a	b	-	-	-
aba	b	a	-	-	-
abac	a	c	9	12:	abac
ca	c	a	-	-	-
cab	a	b	8	13:	cab
ba	b	a	-	-	-
bae	a	e	6	14:	bae
e	e	-	4	-	-

Символ	Код
a	0
b	1
c	2
d	3
e	4
ab	**5**
ba	**6**
ac	**7**
ca	**8**
aba	**9**
ad	**10**
da	**11**
abac	**12**
cab	**13**
bae	**14**

Ответ: 01025039864

ДЕКОДИРОВАНИЕ

Начало.

Шаг 1. Все возможные символы заносятся в словарь. Во входную фразу X заносится первый код декодируемого сообщения.

Шаг 2. Считать очередной код Y из сообщения.

Шаг 3. Если Y — это конец сообщения, то выдать символ, соответствующий коду X, иначе:

Если фразы под кодом XY нет в словаре, вывести фразу, соответствующую коду X, а фразу с кодом XY занести в словарь.

Иначе присвоить входной фразе код XY и перейти к **Шагу 2** .

Конец.

ПРИМЕР

Пусть мы декодируем последовательность: 01025039864.

Код	На выходе	Новая запись			
		Полная		Частичная	
0	a	-	-	5:	a?
1	b	5:	ab	6:	b?
0	a	6:	ba	7:	a?
2	c	7:	ac	8:	c?
5	ab	8:	ca	9:	ab?
0	a	9:	aba	10:	a?
3	d	10:	ad	11:	d?
9	aba	11:	da	12:	aba?
8	ca	12:	abac	13:	ca?
6	ba	13:	cab	14:	ba?
4	e	14:	bae	-	-

Символ	Код
a	0
b	1
c	2
d	3
e	4
ab	**5**
ba	**6**
ac	**7**
ca	**8**
aba	**9**
ad	**10**
da	**11**
abac	**12**
cab	**13**
bae	**14**

ОТНОШЕНИЕ СЖАТИЯ К ФОРМАТУ ФАЙЛА

Хорошо сжимаются:

- тексты (**`*.txt`**)
- документы (**`*.doc`**)
- несжатые рисунки (**`*.bmp`**)
- несжатый звук (**`*.wav`**)
- несжатое видео (**`*.avi`**)

Плохо сжимаются:

- случайные данные
- сжатые данные в архивах (**`*.zip`**, **`*.rar`**, **`*.7z`**)
- сжатые рисунки (**`*.jpg`**, **`*.gif`**, **`*.png`**)
- сжатый звук (**`*.mp3`**, **`*.aac`**)
- сжатое видео (**`*.mpg`**, **`*.mp4`**, **`*.mov`**)