

СЕТЕВЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ



Сжатие без потерь

Сжатие без потерь (lossless compression).

Информация не теряется. Распакованные данные идентичны исходным несжатым данным.

Сжатие с потерями (lossy compression).

Распакованные данные могут быть приемлемым приближением к исходным несжатым данным.

Сжатие без потерь

Метод подавления нулей (null suppression):

BISYNC IBM 3780

Передачик сканирует данные в поиске строк пробелов и заменяет каждую такую строку двухсимвольным кодом. Код состоит из специального управляющего символа, за которым следует число пробелов в строке. Например, пусть имеется код с символами пробела, обозначенными знаком “b”:

XYZbbbb**QRX**

Эта строка заменяется следующей строкой, в которой S_c представляет собой специальный управляющий символ:

XYZS_c5QRX

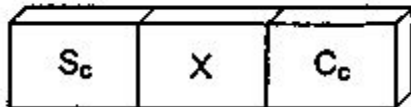
Такая схема позволяет сделать короче все строки из трех и более пробелов.

Сжатие без потерь

Групповое кодирование :

Обобщение метода подавления нулей и применяется для сжатия повторяющихся данных любого типа. На рисунке иллюстрируется применение данного метода к символьным данным. Здесь используются следующие обозначения:

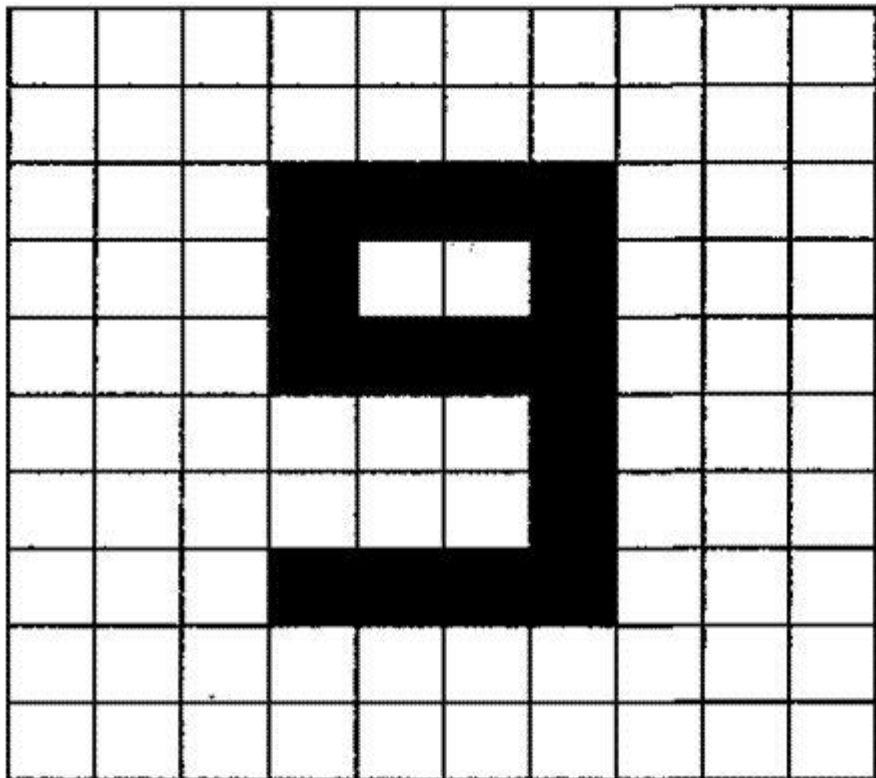
- **S_c** — специальный символ, указывающий на то, что за ним следуют сжатые данные;
- **X** — любой повторяющийся символ;
- **C_c** — счетчик символов, то есть количество повторений сжатого символа.



Пример сжатия при групповом кодировании

Исходная строка данных	Кодированная строка данных
\$*****55.72	$\$S_c^*655.72$
-----	S_c^{-9}
GunsbbbbbbbbbButter	$GunsS_c^b9Butter$

Сжатие без потерь



ИЗОБРАЖЕНИЕ

```
0000000000
0000000000
0001111000
0001001000
0001111000
0000001000
0000001000
0001111000
0000000000
0000000000
```

Длина равна 100 бит

ДВОИЧНЫЙ КОД:

**23W 4B 6W 1B 2W 1B 6W 4B 9W 1B
9W 1B 6W 4B 23W**

ИЛИ

**23 4 6 1 2 1 6 4 9 1 9 1 6 4
23**

Длина равна 15 символов или
120 бит

Сжатие без потерь

Факсимильное сжатие:

**Страница, отсканированная с разрешением 200 пел (белых или черных точек на дюйм): 3 740 000 бит (8.5" x 11" x 40 000 пелов на кв. дюйм)
ISDN (64 Кбит/с) – время передачи ~1 мин.**

Два метода сжатия данных без потерь для факсимильной связи:

- модифицированный код Хаффмана - группа 3.
- модифицированный код READ (Relative Element Address Designate – относительное назначение адресов элементов) - группа 4.

Группа 3. Кодирование черных и белых значений с плотностью в 200 точек на дюйм (гориз.) и 100...200 (верт.). Предполагается, что передача сигнала осуществляется через модем по аналоговой телефонной линии. Передача данных ускоряется в три и более раз по сравнению с группой 2.

Группа 4. Черно-белый цифровой факсимильный стандарт. Эта группа предназначена для использования в цифровых сетях со скоростями до 64 Кбит/с. Стандартизированы разрешения от 200 до 400 точек на дюйм. Время передачи одной страницы сокращено до нескольких секунд по сравнению с несколькими минутами в предыдущих стандартах.

Сжатие без потерь

Модифицированный код Хаффмана:

W7, B7, W4, B8, W4, B7, W10

ITU-T: 1728 точек на линию

Длина серии N рассматривается как сумма двух слагаемых:

$$N = 64m + n; m = 0, 1, 2, \dots, 27; n = 0, 1, 2, \dots, 63$$

P: строка из 200 черных точек: $64 \cdot 3 + 8$

EOL (End Of Line — конец строки)

Сжатие без потерь

Арифметическое кодирование. Основная концепция.

$$\log\left(\frac{1}{P_i}\right) \leq L_i \leq \log\left(\frac{1}{P_i}\right) + 1 \quad H(X) \leq E[L] < H(X) + 1$$

$H(X)$ - энтропия множества символов X ,

$E[L]$ – средняя длина кода для множества X .

В схеме сжатия без потерь $H(X)$ всегда будет нижней границей объема сжатых данных, так как $H(X)$ определяет среднее количество информации, содержащейся в символьном наборе. Эффективность алгоритма сжатия может быть повышена путем объединения символов и одновременного кодирования блоков по K символов каждый. В результате мы получаем следующее неравенство:

$$H(X) \leq \frac{E[L]}{K} < H(X) + \frac{1}{K}$$

Сжатие без потерь

Арифметическое кодирование. Основная концепция.

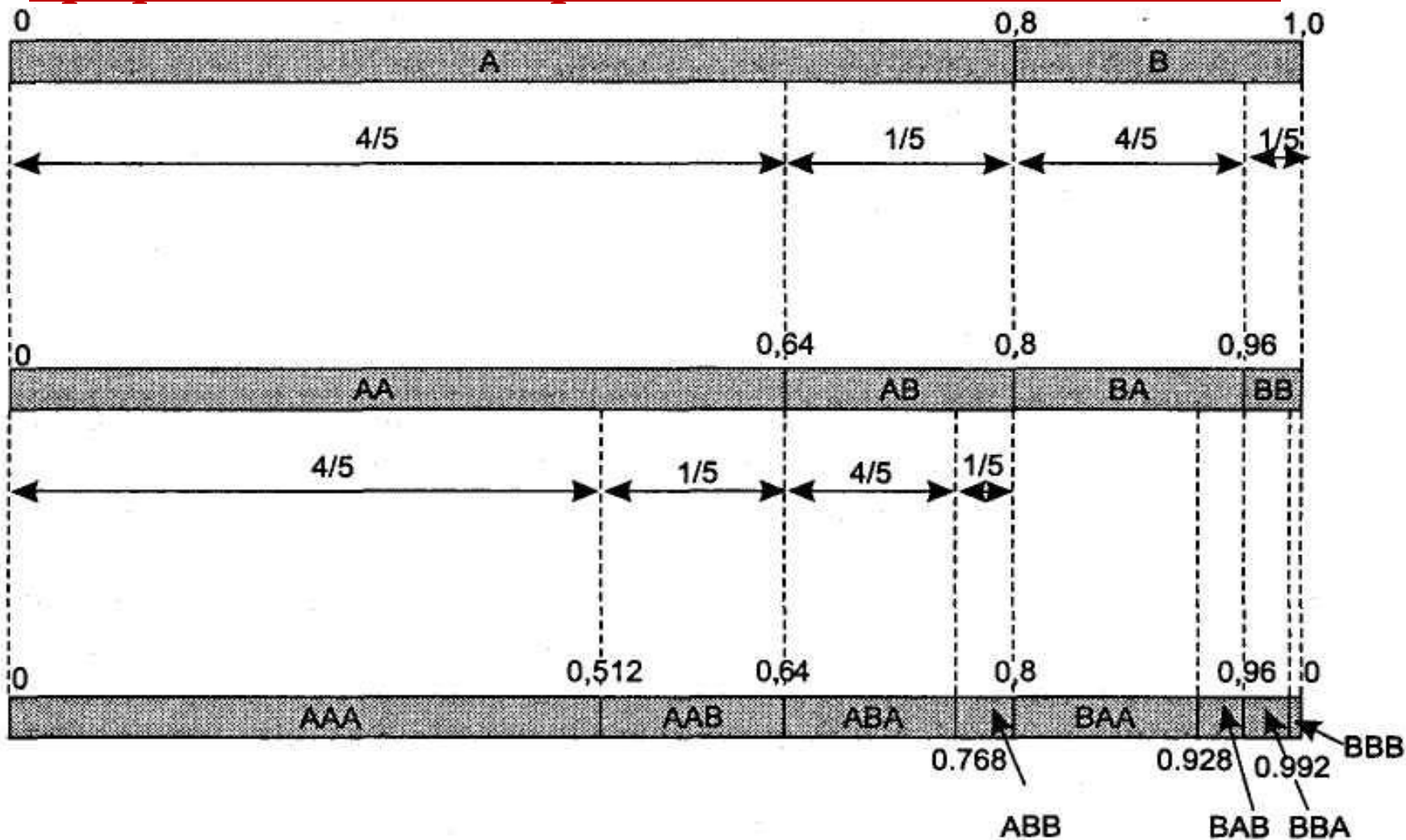
Таким образом, существует способ повышения эффективности алгоритма сжатия. Если нам нужно отправить сообщение длиной в N символов, мы можем использовать блок длиной N . Таким образом, мы обращаемся с каждым сообщением как с одним из M^N возможных результатов, где M представляет собой количество атомарных символов, а N — длину сообщения.

$$\begin{array}{ll}
 P_{AAA} = 0.512, & P_{BAA} = 0.128, \\
 P_{AAB} = 0.128, & P_{BAB} = 0.032, \\
 P_{ABA} = 0.128, & P_{BBA} = 0.032, \\
 P_{ABB} = 0.032, & P_{BBB} = 0.032.
 \end{array}$$

Для сообщения с восемью возможными результатами это разумно, но при очень длинных сообщениях использование этого метода приведет к большим вычислительным затратам.

Сжатие без потерь

Арифметическое кодирование. Основная концепция.



Сжатие без потерь

Арифметическое кодирование. Основная концепция.

Теперь результату каждого сообщения поставлен в соответствие интервал, пропорциональный вероятности сообщения. Таким образом, каждый результат может быть представлен двоичным дробным числом, равным некоторой точке на интервале.

Последовательность	P_i	Кумулятивная вероятность	Интервал	Двоичное представление нижней границы	Код	$P_i L_i$	$P_i \log(1/P_i)$
AAA	0.512	0.512	[0,0.512]	0.00000	0	0.512	0.494
AAВ	0.128	0.64	[0.512,0.64]	0.10000	100	0.384	0.38
АВА	0.128	0.768	[0.64,0.768]	0.10100	101	0.384	0.38
АВВ	0.032	0.8	[0.768,0.8]	0.11000	11000	0.16	0.159
ВАА	0.128	0.928	[0.8,0.928]	0.11001	11001	0.64	0.38
ВАВ	0.032	0.96	[0.928,0.96]	0.11101	111	0.096	0.159
ВВА	0.032	0.992	[0.96,0.992]	0.11110	11110	0.096	0.159
ВВВ	0.008	1.0	[0.992,1.0]	0.11111	11111	0.04	0.056
						2.408	2.167

Сжатие без потерь

Чистое арифметическое кодирование:

БАЗОВЫЙ АЛГОРИТМ:

Каждый шаг алгоритма начинается с полуоткрытого интервала

$[L, H)$, вначале инициализируемого как $[0,1)$

- Текущий интервал разбивается на подынтервалы по одному для каждого возможного символа. Каждый подынтервал пропорционален вероятности, с которой соответствующий символ встречается в тексте.
- Выбирается подынтервал, соответствующий текущему символу. Этот подынтервал становится текущим интервалом.
- Если обработано все сообщение, выполняется следующий шаг, если нет, возвращаемся к шагу 1.
- Выводится количество битов, достаточное для того, чтобы можно было отличить данный интервал от двух соседних интервалов.
- Выводится некий специальный код конца сообщения, чтобы уведомить получателя.

Сжатие без потерь

Арифметическое кодирование :

ПРИМЕР1:

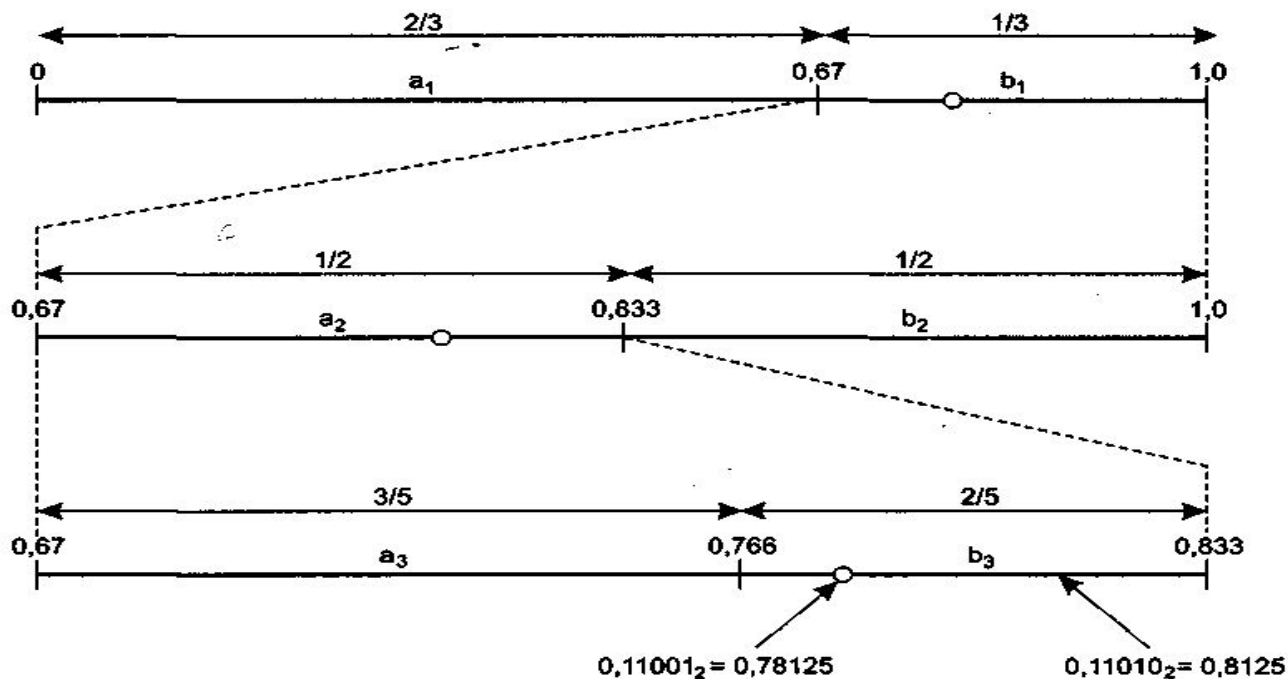
Пусть имеются два символа, a и b , вероятности которых зависят от положения в трехсимвольном сообщении:

$$\Pr(a_1) = 2/3, \quad \Pr(b_1) = 1/3,$$

$$\Pr(a_2) = 1/2, \quad \Pr(b_2) = 1/2,$$

$$\Pr(a_3) = 3/5, \quad \Pr(b_3) = 2/5.$$

Здесь $\Pr(a_i)$ представляет собой вероятность того, что символ a встретится в i -й позиции сообщения. В данном случае кодируется сообщение «bab».



Сжатие без потерь

Инкрементное арифметическое кодирование

Недостатки арифметического кодирования:

- уменьшающиеся размеры текущего интервала требуют все более точных вычислений;
- код не может быть передан, пока не закодировано все сообщение.

$$[L, H) = [0, 1)$$

Используется счетчик следования (FC)

