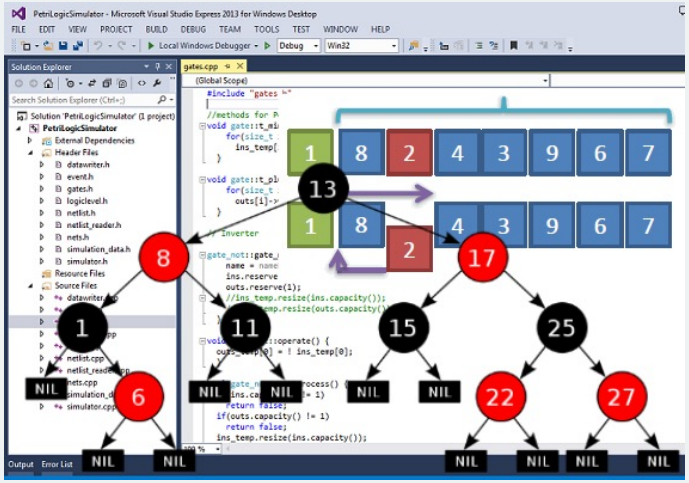
МИЭТ
Национальный исследовательский университет «МИЭТ»
Кафедра ПКИМС

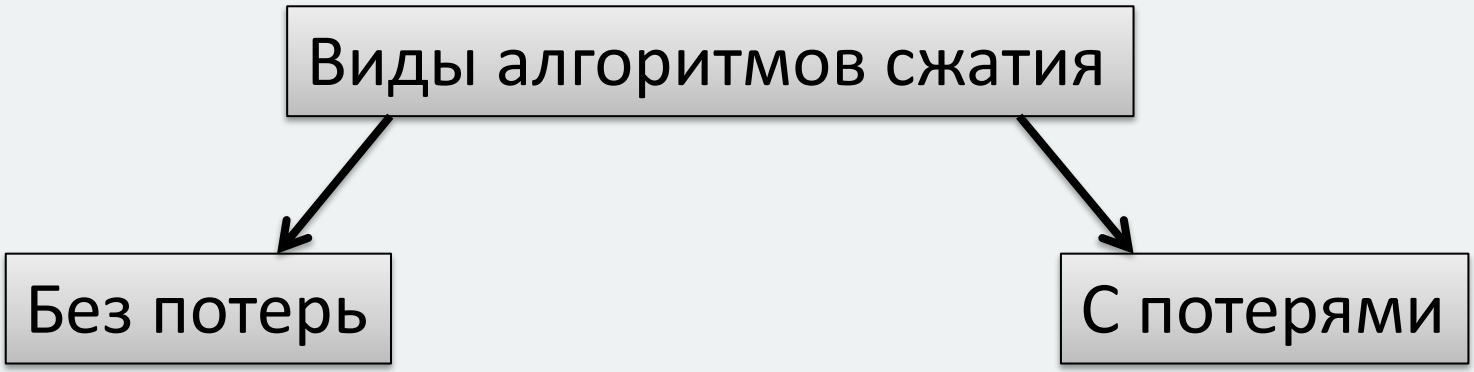
Теория алгоритмов



Лекция 5

Основы алгоритмов компрессии данных.

Виды сжатия



После декомпрессии или обработки последовательность бит **в точности соответствует** исходным данным

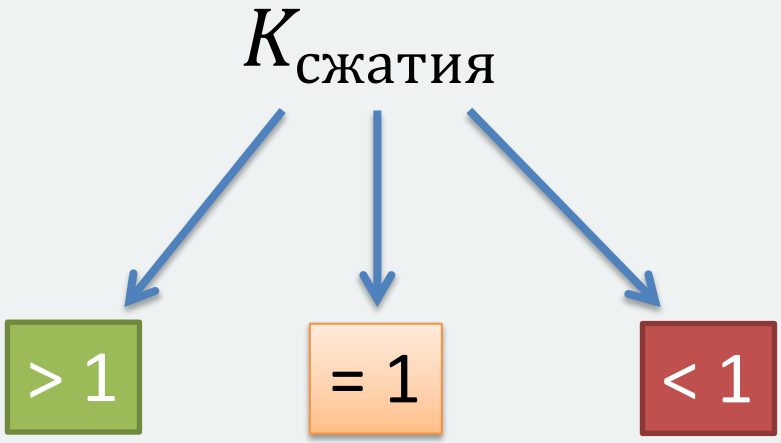
После декомпрессии или обработки последовательность бит **не соответствует** исходным данным, но это не является принципиальным моментом с точки зрения дальнейшей работы

Коэффициент сжатия

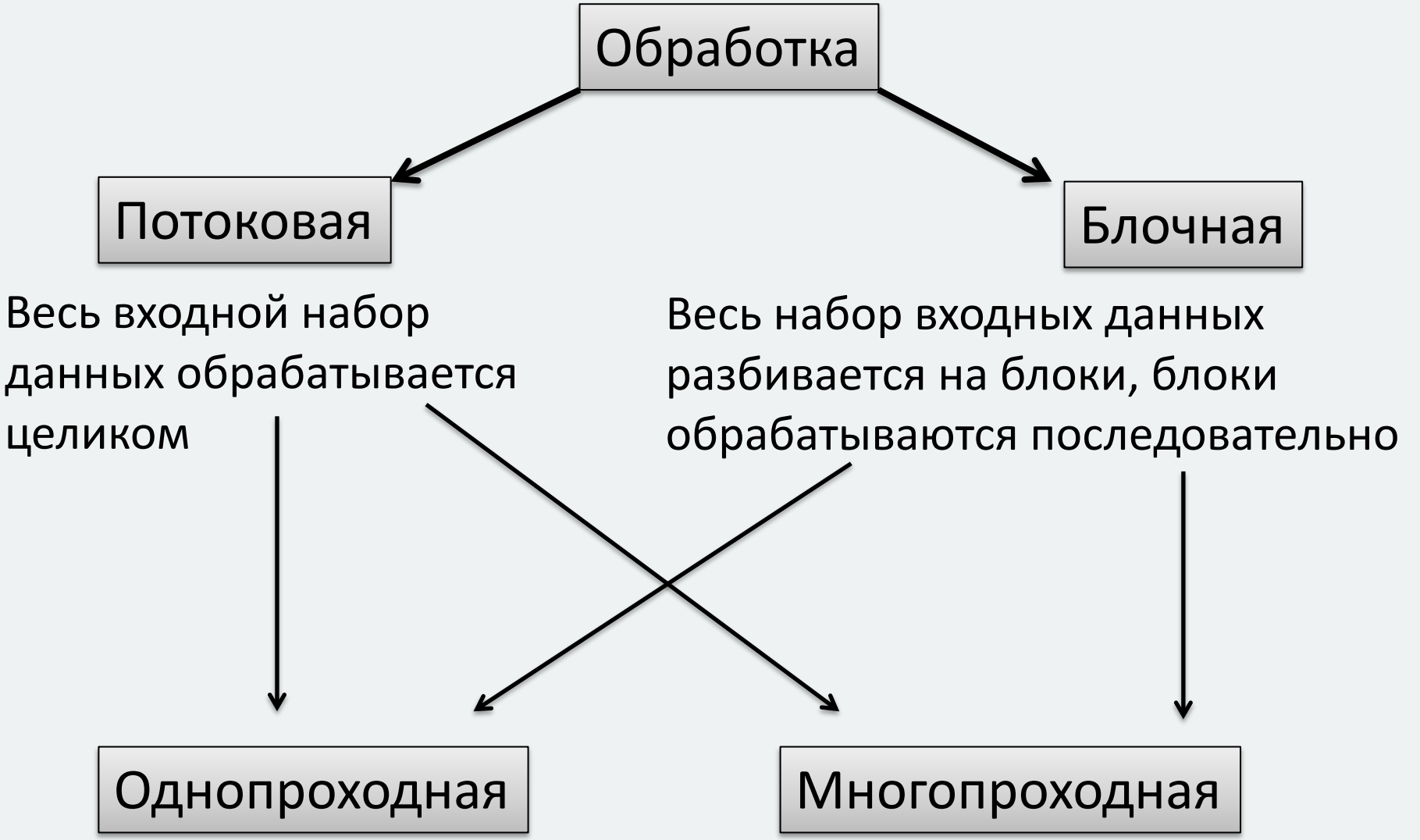
$$K_{\text{сжатия}} = \frac{L_{\text{исходных данных}}}{L_{\text{результата}}}$$

Какие значения может принимать коэффициент сжатия данных?

Коэффициент сжатия определяет эффективность алгоритма сжатия



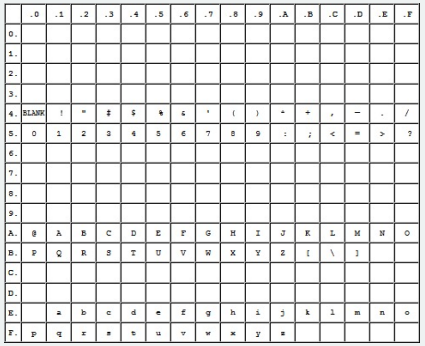
Типы обработок данных



7-битная передача текстовых данных

При 7-ми битном кодировании используются всего 7 бит из 8-ми на каждый байт.

Объём передаваемых данных: 8*(N-1) бит



	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.A	.B	.C	.D	.E	.F
0.																
1.																
2.																
3.																
4.	BLANK	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
5.	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
6.																
7.																
8.																
9.																
A.	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
B.	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]		
C.																
D.																
E.		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
F.	p	q	r	s	t	u	v	w	x	y	z					

10 байт – 80 бит – 70 бит – 9 байт, Δ=1 байт

1Кб - 8192 бита – 7168 бит – 896 байт, Δ=128 байт

1Мб – 8388608 бит – 7340032 бит – 896Кб, Δ=128 Кб

Стоит ли из-за 900Кб возиться с алгоритмами компрессии/декомпрессии?

Δ=128 Кб при скорости 14400 бод даёт сокращение в 2 минуты!

Простейший алгоритм сжатия : RLE

RLE
Run Length Encoding

Алгоритм:

1. считываем со входа символы до тех пор, пока идут повторяющиеся символы
2. считаем, сколько одинаковых символов прошло
3. пишем на выход число - сколько одинаковых символов прошло
4. пишем символ, который повторялся

Тестовая последовательность:

ААААББВГАААДДДДДДДДДЕЕЖЖЖЖЖЖЖЖЖЖ33333

Результат работы алгоритма сжатия RLE:

3А2Б1В1Г2А9Д2Е11Ж53

Алгоритм Хаффмана (6)

Тестовая последовательность:

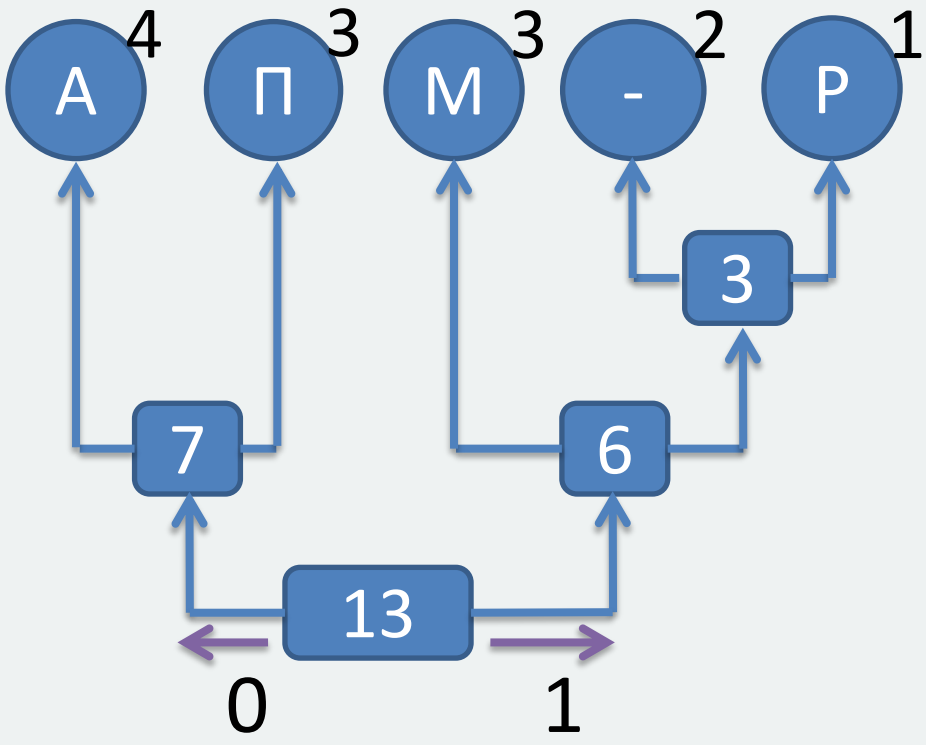
ПАРАМ-ПАМ-ПАМ

Алгоритм:

1. проходим по всему набору данных, посчитаем частоты вхождения символов
2. сортируем символы по частоте, с которой они встречаются в наборе данных
3. начинаем строить дерево
   3-а. выбираем 2 символа с минимальными частотами, заносим в дерево
   3-б. повторяем операцию, наращиваем глубину (высоту) дерева, добавляя к созданному элементу те, которые имеют минимальную частоту.
4. Выполняем проверку: убеждаемся, что итоговое значение равно сумме частот появлений символов

5. Строим таблицу путей по дереву для каждого из символов

А	00
П	01
М	10
-	110
Р	111



Алгоритм LZW (1)

Тестовая последовательность:
ПАРАМ-ПАМ-ПАМ

Лемпель, Зив, Уэлч

Алгоритм:

```
STR = data[0];
i = 1;
while(i < data.length) {
   SYM = data[i];

   if(LUT.find(STR + SYM))
      STR = STR + SYM;
   else {
      out << LUT.code_for(STR);
      LUT.add(STR + SYM);
      STR = SYM;
   }
   ++i;
}
```

Потоковые алгоритмы vs Блочные алгоритмы

Последовательность данных

$K_{\text{сжатия}} < 1$

Подпоследовательность данных-1

$K_{\text{сжатия\_блок1}}$

Подпоследовательность данных-2

$K_{\text{сжатия\_блок2}}$

Подпоследовательность данных-3

$K_{\text{сжатия\_блок3}}$

```
data.block[i].compress();

if(data.block[i].koeff > 1)
  write(data.block[i].compressed)
else
  write(data.block[i].uncompressed);
```

Блочный алгоритм: архиватор ZIP

Блочный алгоритм: архиватор bz2

Работа алгоритма основана на преобразовании Барроуза-Уиллера (BWT)

Алгоритм компрессии:

1. для входной последовательности организуется циклическая перестановка со сдвигом влево, все возможные комбинации;
2. получившаяся последовательность строк сортируется лексикографически;
3. запоминаем последний столбец – это наш блок данных;
4. запоминаем номер исходной строки;
5. кодируем блок обычным алгоритмом, например LZW;

Блочный алгоритм: архиватор bz2 - декомпрессия

Алгоритм декомпрессии:

1. декодируем блок данных;
2. создаём пустую квадратную матрицу по числу символов;
3. выписываем декодированную последовательность в первый пустой столбец матрицы, начиная справа;
4. сортируем матрицу построчно;
5. повторяем пункты 3-4, пока не заполним матрицу;
6. в строке с запомненным номером хранится исходная строка;

Алгоритмы сжатия

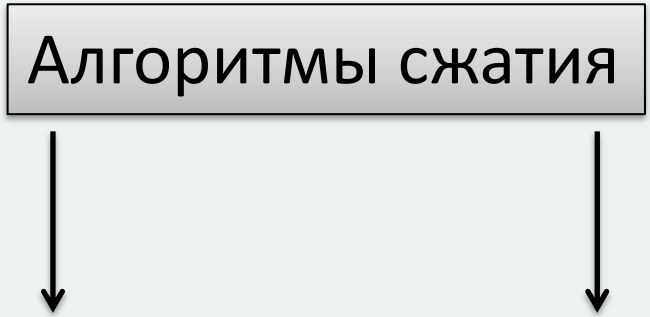
Виды алгоритмов сжатия

Без потерь

- RLE (PCX, BMP, TGA, TIFF)
- Huffman (PNG)
- LZW (GIF)

С потерями

Алгоритмы сжатия видеоданных



Пространственные

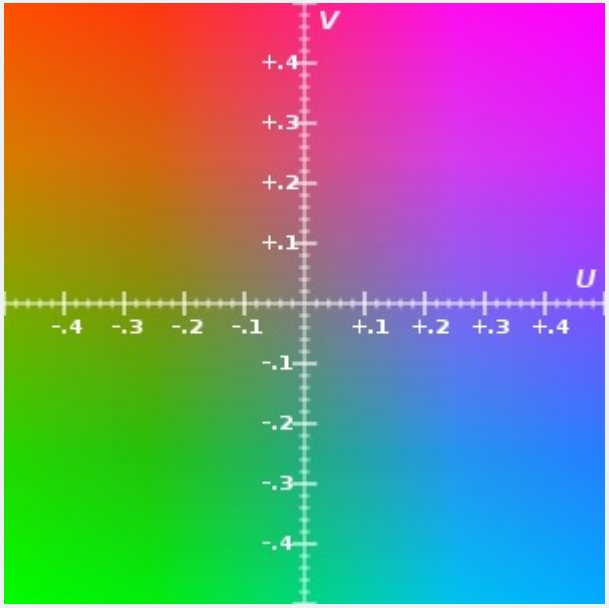
Сжатие производится в пределах одного кадра

Временные

Сжатие производится для нескольких кадров

Сжатие с потерями: JPEG

JPEG
Joint Photographic Experts Group



Основан на переводе RGB в YUV

Y – яркость
U – синяя цветоразностная компонента
V – красная цветоразностная компонента

```
Y = 0.299 * R + 0.587 * G + 0.114 * B;
U = -0.14713 * R - 0.28886 * G + 0.436 * B + 128;
V = 0.615 * R - 0.51499 * G - 0.10001 * B + 128;
```

```
R = Y + 1.13983 * (V - 128);
G = Y - 0.39465 * (U - 128) - 0.58060 * (V - 128);
B = Y + 2.03211 * (U - 128);
```

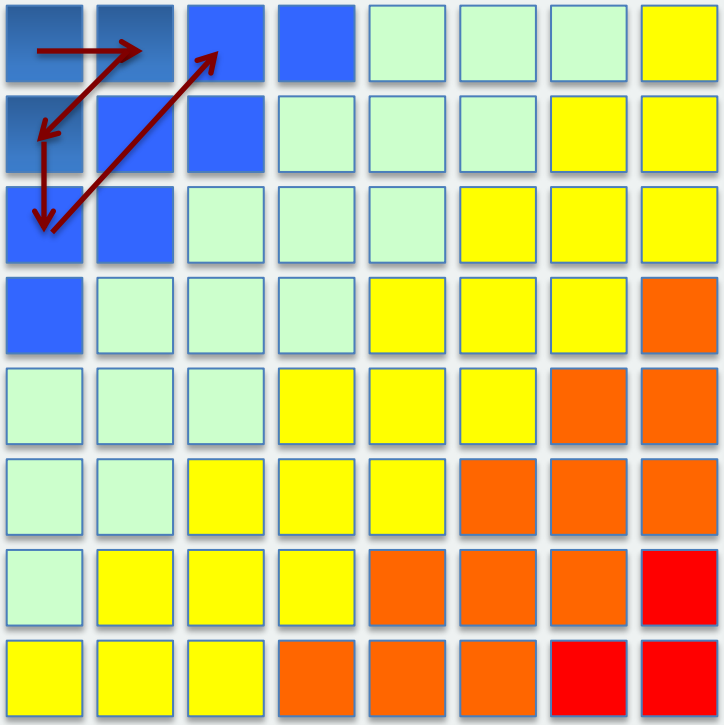
Алгоритм компрессии JPEG

перевод RGB в YUV

паковка группами по 8x8 пикселей

применяется дискретное косинусное преобразование – группировка по частотам в изображении

применяется алгоритм кодирования Хаффмана



Примеры формата JPEG с разной степенью сжатия (1)

Сжатие видеопотока: деинтерлейсинг

Сжатие видеопотока: временное сжатие

Разница кадров с компенсацией

Сжатие видеопотока: временное сжатие