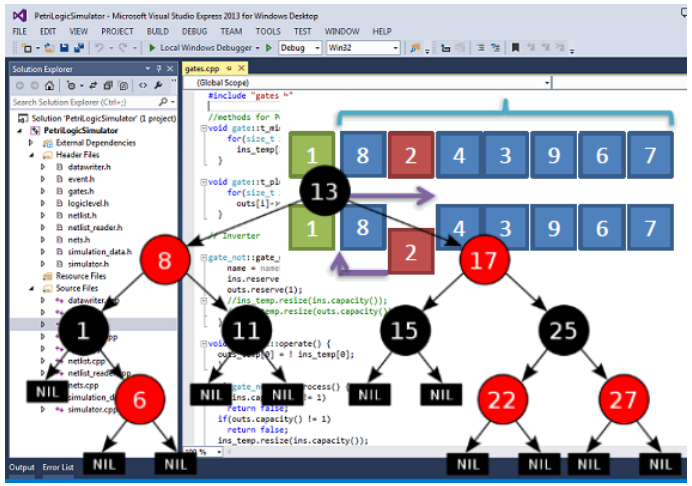
Национальный исследовательский университет «МИЭТ»

Институт интегральной электроники (группы ЭН-24-25, каф. ПКИМС)

Теория алгоритмов



Лекция 5

Основы алгоритмов компрессии данных

Виды сжатия

Виды алгоритмов сжатия

Без потерь

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

С потерями

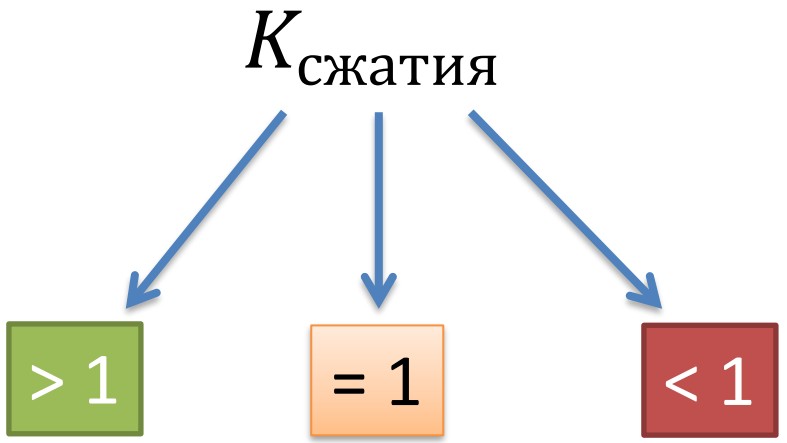
После декомпрессии или обработки последовательность бит **не соответствует** исходным данным, но это не является принципиальным моментом с точки зрения дальнейшей работы

Коэффициент сжатия

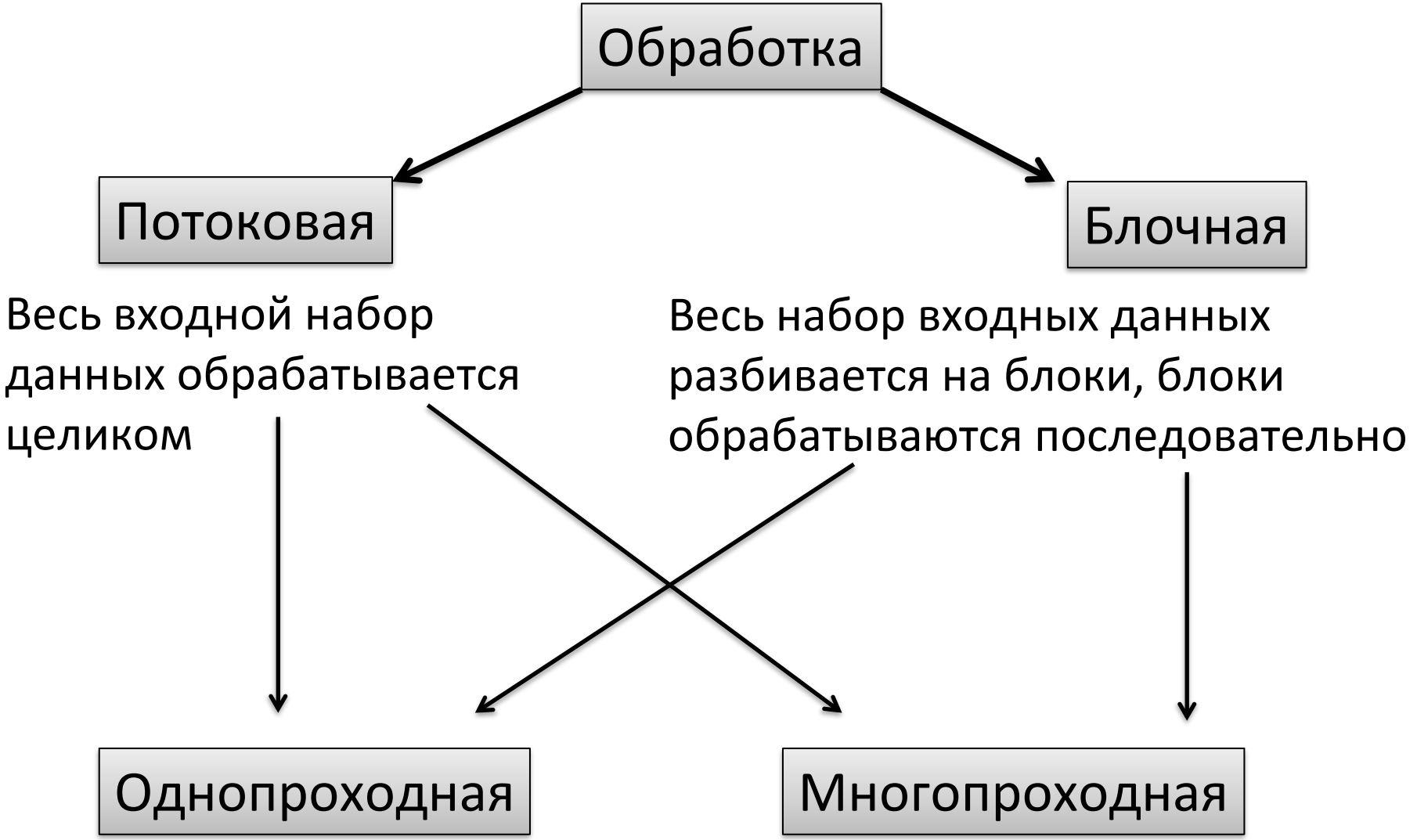
$$K_{\text{сжатия}} = \frac{L_{\text{исходных данных}}}{L_{\text{результата}}}$$

Коэффициент сжатия определяет эффективность алгоритма сжатия

Какие значения может принимать коэффициент сжатия данных?



Типы обработок данных



Простейший алгоритм сжатия : RLE

RLE - Run Length Encoding

Кодирование длин серий

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
5. повторяем пункты 1-4, пока есть входные данные

Тестовая последовательность:

ААААББВГАААДДДДДДДДДЕЕЖЖЖЖЖЖЖЖЖЖ33333

Результат работы алгоритма сжатия RLE:

4А2Б1В1Г2А9Д2Е11Ж5З

Эффективность алгоритма RLE

Алгоритм эффективен для любых типов файлов, имеющих большое число повторяющихся последовательностей байт

Алгоритм эффективен для типов данных, имеющих большое число повторяющихся типов заранее предопределённых последовательностей

Алгоритм не требует информации о всём наборе данных, может использоваться при передаче данных по каналам связи

Модификации алгоритма RLE: PackBits

АБББББВГДЕЖЖЖЗЗИИКЛЛЛЛМНОПРРРРСТУ → 1А4Б1В1Г1Д1Е3Ж2З2И1К4Л1М1Н1О1П4Р1С1Т1У

32 байта — 38 байт

[19] {01000011101000001000} А4БВГДЕ3Ж2З2ИК4ЛМНОП4РСТУ

1 байт — 3 байта — 25 байт

[1]А[-4]Б[4]ВГДЕ[-3]Ж[-2]З[-2]И[1]К[-4]Л[4]МНОП[-4]Р[3]СТУ

30 байт

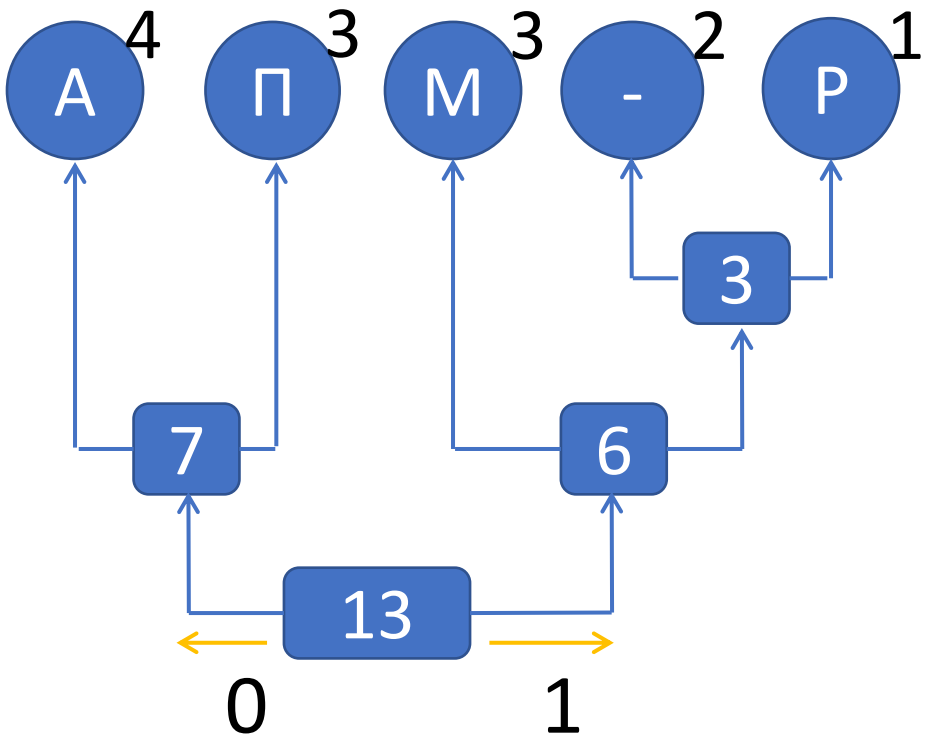
Алгоритм Хаффмана

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

Тестовая последовательность:

ПАРАМ-ПАМ-ПАМ



А	00
П	01
М	10
-	110
Р	111

Алгоритм LZW

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;
}
```

Алгоритм LZW (2)

Тестовая последовательность:

ПАРАМ-ПАМ-ПАМ

Выход: П А Р А М - «256»

```
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;
}
```

	Таблица	Строка	Символ
256	ПА	П	А
257	АР	А	Р
258	РА	Р	А
259	АМ	А	М
260	М-	М	-
261	-П	-	П
		П	А
262	ПАМ	ПА	М

Алгоритм LZW (3)

Тестовая последовательность:

ПАРАМ-ПАМ-ПАМ

Выход: П А Р А М - «256»«260» «262»

```
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;
}
```

Таблица	Строка	Символ
256 ПА	П	А
257 АР	А	Р
258 РА	Р	А
259 АМ	А	М
260 М-	М	-
261 -П	-	П
	П	А
262 ПАМ	ПА	М

Таблица	Строка	Символ
261 -П	-	П
262 ПАМ	ПА	М
	М	-
263 М-П	М-	П
	П	А
	ПА	М
	ПАМ	

Эффективность и характеристики алгоритма LZW

Эффективность

Исходная последовательность:

ПАРАМ-ПАМ-ПАМ

Длина исходной последовательности:
104 бита

Сжатая последовательность:

ПАРАМ- «256» «260» «262»

Длина сжатой последовательности: 75 бит

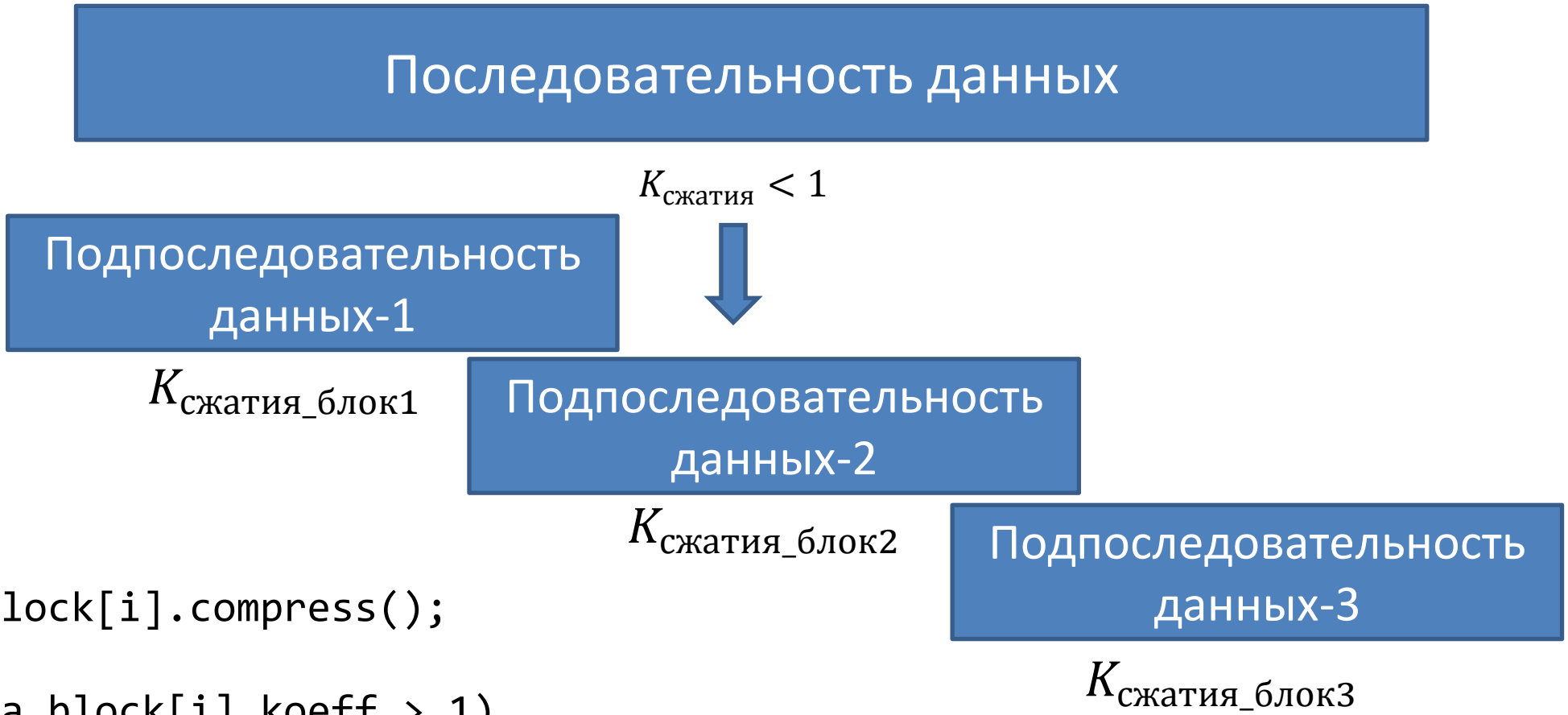
Характеристики

алгоритм является однопроходным

размер поля таблицы всегда одинаков (для набора) и зависит от размера «сканирующего окна»

не требуется хранить информацию о словаре – очевидно, что его можно формировать динамически, читая сжатое сообщение

Блочные и потоковые алгоритмы




```
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

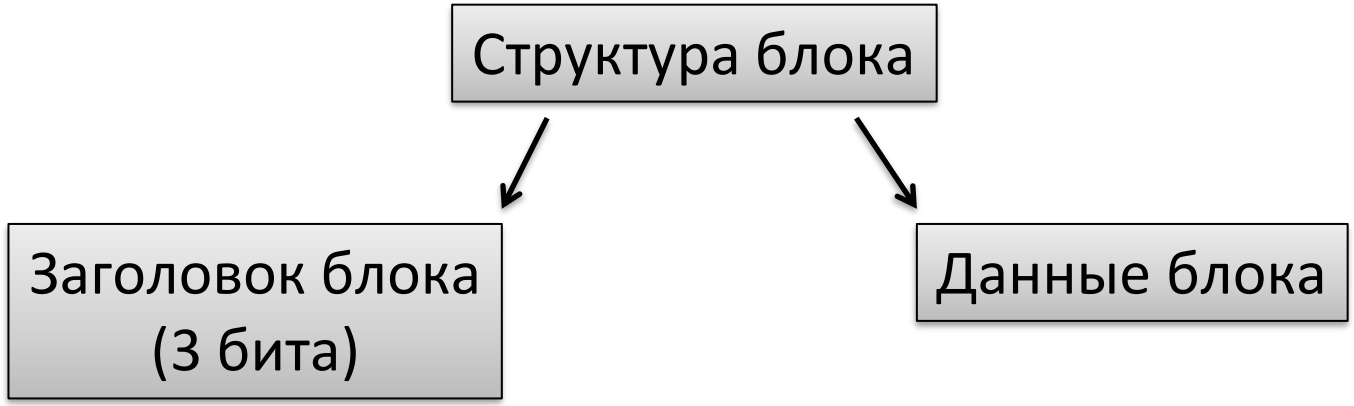
...

Блок N-1

Блок N

Блок N+1

...



метод сжатия данных блока (2 бита)
00 – исходные данные
01 – алгоритм Хаффмана
10 – алгоритм Хаффмана
11 – зарезервировано

Блочный алгоритм: архиватор bz2

Алгоритм компрессии:

1. для входной последовательности организуется циклическая перестановка со сдвигом влево;
2. получившаяся последовательность сортируется лексикографически;
3. запоминаем последний столбец;
4. запоминаем номер исходной строки;
5. кодируем последний столбец;

Тестовая последовательность:

ПАРАМПАМ

ПАРАМПАМ	АМПАМПАР	
АРАМПАМП	АМПАРАМП	
РАМПАМПА	АРАМПАМП	
АМПАМПАР	МПАМПАРА	
МПАМПАРА	МПАРАМПА	
ПАМПАРАМ	ПАМПАРАМ	
АМПАРАМП	ПАРАМПАМ	
МПАРАМПА	РАМПАМПА	

1Р2П2А2М1А,7

Алгоритмы сжатия

Виды алгоритмов сжатия

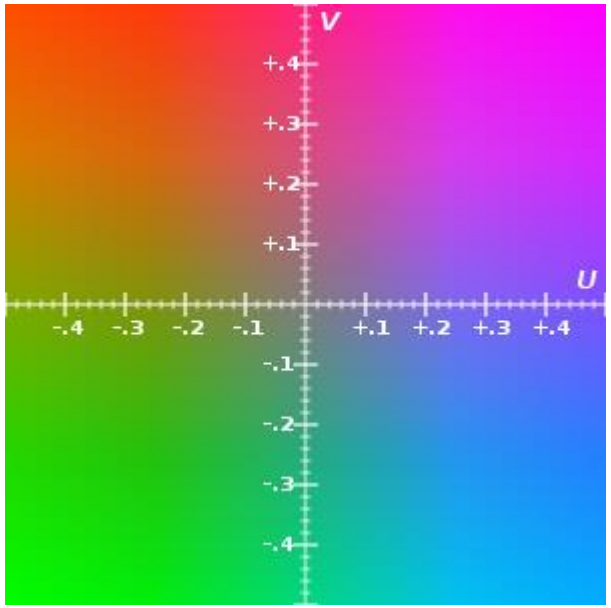
Без потерь

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

С потерями

Сжатие с потерями: формат JPEG

JPEG
Joint Photographic Experts Group



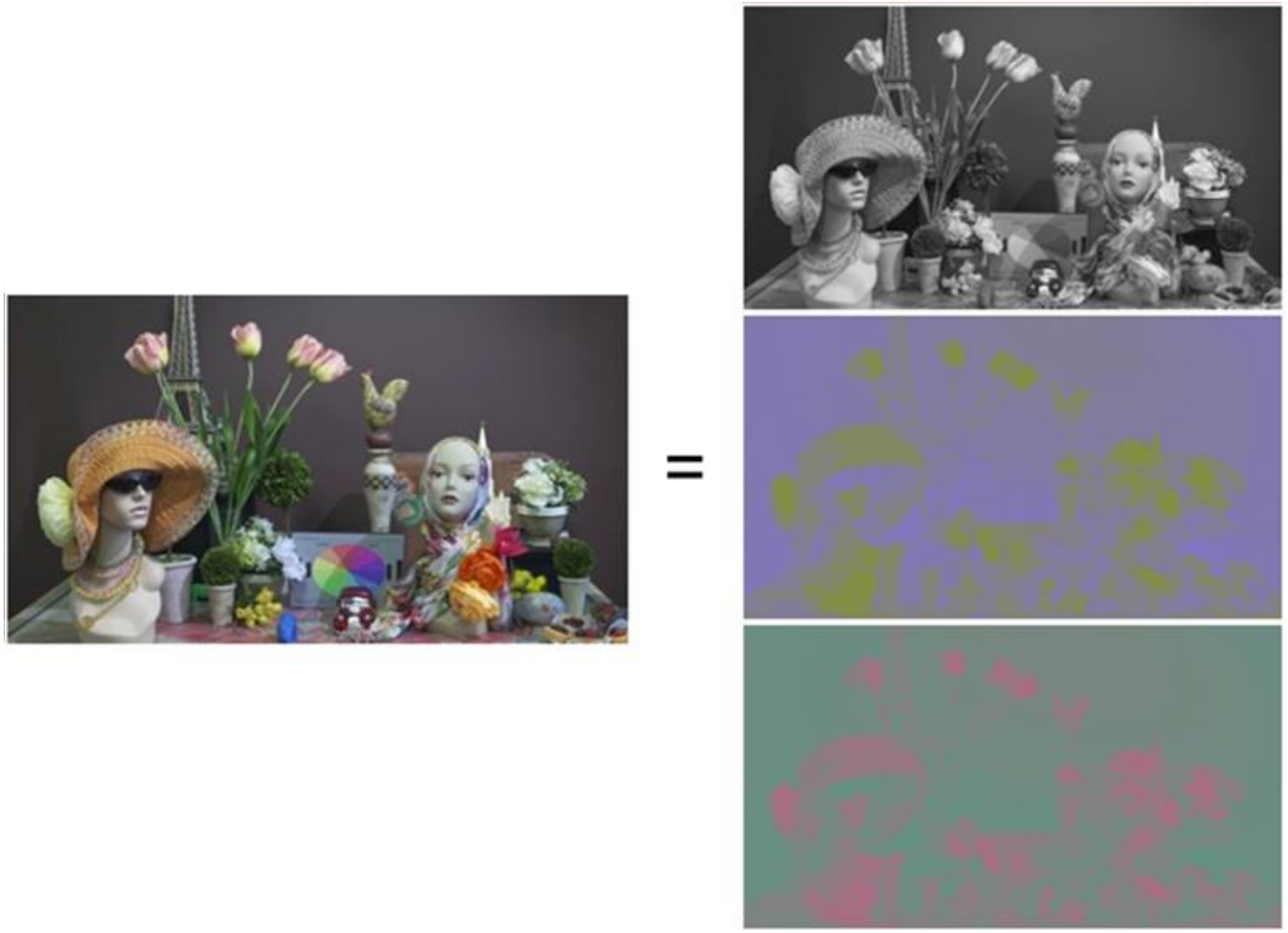
Основан на переводе RGB в YUV

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

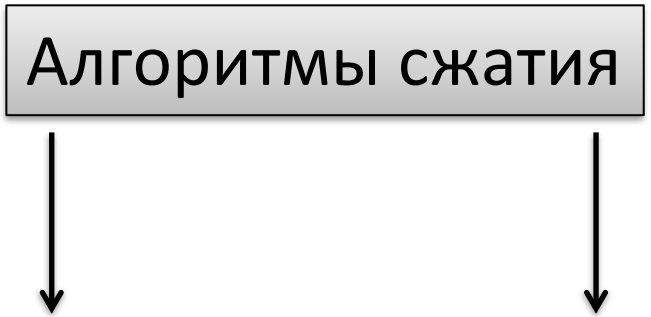
```
Y = 0.299 * R + 0.587 * G + 0.114 * B;
U = -0.14713 * R - 0.28886 * G + 0.436 * B + 128;
V = 0.615 * R - 0.51499 * G - 0.10001 * B + 128;
```

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

Цветовые пространства YUV и RGB



Алгоритмы сжатия видеоданных



Пространственные

Сжатие производится в пределах одного кадра

Временные

Сжатие производится для нескольких кадров

Сжатие видеопотока: деинтерлейсинг

Определение макроблоков и векторов движения

Разница кадров с компенсацией

Сжатие видеопотока: временное сжатие