

# Методы сжатия цифровой информации

# АЛГОРИТМЫ ОБРАТИМЫХ МЕТОДОВ

## Свойства алгоритмов сжатия

Алгоритм	Выходная структура	Сфера приме- не-	Примечание
RLE (Run-Length Encoding)	Список (вектор данных)	Графические дан- ные	Эффективность алгоритма не за- висит от объема данных
KWE (Keyword En- coding)	Таблица данных (сло- варь)	Текстовые данные	Эффективен для массивов боль- шого объема
Алгоритм Хафмана	Иерархическая структу- ра (дерево кодировки)	Любые данные	Эффективен для массивов боль- шого объема

# **АЛГОРИТМ кодирования по ключевым словам**

# АЛГОРИТМ KWE

В основу алгоритмов кодирования по ключевым словам (Keyword Encoding) положено кодирование лексических единиц исходного документа группами байтов фиксированной длины. Примером лексической единицы может служить слово (последовательность символов, справа и слева ограниченная пробелами или символами конца абзаца). Результат кодирования сводится в таблицу,

Эффективность данного метода существенно зависит от длины документа, поскольку из-за необходимости прикладывать к архиву словарь длина кратких документов не только не уменьшается, но даже возрастает.

Данный алгоритм наиболее эффективен для англоязычных текстовых документов и файлов баз данных. Для русскоязычных документов, отличающихся увеличенной длиной слов и большим количеством приставок, суффиксов и окончаний, не всегда удается ограничиться двухбайтными токенами, и эффективность метода заметно снижается.

# алгоритм LZ и алгоритм LZW

Существует довольно много реализаций этого алгоритма, среди которых наиболее распространенными являются:

- алгоритм Лемпеля-Зіва (**алгоритм LZ**) и его модификация
- алгоритм Лемпеля-Зіва-Велча (**алгоритм LZW**).

# алгоритм LZ и алгоритм LZW

- Словарем в данном алгоритме является потенциально бесконечный список фраз. Алгоритм начинает работу с почти пустым словарем, который содержит только одну закодированную строку, так называемая NULL-строка. При считывании очередного символа входной последовательности данных, он прибавляется к текущей строке. Процесс продолжается до тех пор, пока текущая строка соответствует какой-нибудь фразе из словаря.
- Алгоритм LZW построен вокруг таблицы фраз (словаря), которая заменяет строки символов сжимаемого сообщения в коды фиксированной длины.

Алгоритмы сжатия этой группы наиболее эффективны для текстовых данных больших объемов и малоэффективны для файлов маленьких размеров (за счет необходимости сохранения словаря).

# LZ78

Кодируются не отдельные буквы, а слова.

**Общая схема алгоритма LZ77 такова** (это не точное описание алгоритма):

- входные данные читаются последовательно, текущая позиция условно разбивает массив входных данных на прочитанную и непрочитанную части;
- для цепочки первых байтов непрочитанной части ищется наиболее длинное совпадение в прочитанной части. Если совпадение найдено, то составляется комбинация {смещение, длина}, где смещение указывает, на сколько байтов надо сместиться назад от текущей позиции, чтобы найти совпадение, а длина — это длина совпадения;
- если запись комбинации {смещение, длина} короче совпадения, то она записывается в выходной массив, а текущая позиция перемещается вперед (на длину совпадающей части);
- если совпадение не обнаружено или оно короче записи комбинации {смещение, длина}, то в выходной массив копируется текущий байт, текущая позиция перемещается вперед на 1, и анализ повторяется.

- Алгоритмы Лемпела Зива - Яндекс.Видео (yandex.ru)

Недостатки LZ77:

- с ростом размеров словаря скорость работы алгоритма-кодера пропорционально замедляется;
- кодирование одиночных символов очень неэффективно.

Пример. **КРАСНАЯКРАСКА**  
**КРАС(7,4)КА**

Пример. Фраза **КОЛОКОЛ\_ОКОЛО\_КОЛОКОЛЬНИ**

закодировается алгоритмом LZ77 как \_\_\_\_\_



# Алгоритм Лемпела-Зива - Яндекс.Видео (yandex.ru)

**Общая схема алгоритма LZ78** такова (это не точное описание алгоритма):

- алгоритм во время сжатия текста создает специальный *словарь повторяющихся цепочек*, в словаре каждой цепочке соответствует короткий код;
- для цепочки первых байтов непрочитанной части ищется наиболее длинное совпадение в словаре. Код совпадения записывается в выходной массив, туда же заносится первый несовпавший символ, и текущая позиция перемещается вперед на длину совпадения + 1;
- в словарь добавляется новое слово: «совпадение» + «несовпавший символ», и процесс повторяется до тех пор, пока не будет сжат весь входной массив.

Алгоритмы Лемпеля—Зива тем лучше сжимают текст, чем больше размер входного массива. Характерной особенностью обратных алгоритмов LZ77 и LZ78 является то, что, кроме самих сжатых данных, никакой дополнительной информации им не требуется.

- В следующей лекции мы рассмотрим  
АЛГОРИТМЫ СЖАТИЯ С ПОТЕРЯМИ.