**Лабораторная работа № 10**

**СЖАТИЕ/РАСПАКОВКА ДАННЫХ МЕТОДОМ ЛЕМПЕЛЯ – ЗИВА**

**Цель:** приобретение практических навыков использования метод Лемпеля − Зива (Lempel-Ziv) для сжатия/распаковки данных.

**Теоретические сведения**

**Сжатие данных** осуществляется за счет замены записей соответствующими кодами из словаря.

*«если в проанализированном (сжатом) ранее выходном потоке уже встречалась подобная последовательность байт, причем запись о ее длине и смещении от текущей позиции короче, чем сама эта последовательность, то в выходной файл записывается ссылка (смещение, длина), а не сама последовательность»*

**Суть метода LZ77** (как и последующих его модификаций) состоит в следующем: упаковщик постоянно хранит некоторое количество последних обработанных символов в буфере.

По мере обработки входного потока вновь поступившие символы попадают в конец буфера, сдвигая предшествующие символы и вытесняя самые старые.

Размеры этого буфера, называемого также скользящим словарем, варьируются в разных реализациях систем сжатия.

Скользящее окно имеет длину n, т. е. в него помещается n символов, и состоит из двух частей:

* последовательности длины n1 = n − n2 уже закодированных символов (словарь);
* упреждающего буфера (буфера предварительного просмотра, lookahead) длиной n2 – буфера кодирования

Словарем будут являться n1 предшествующих символов. В буфере находятся ожидающие кодирования (сжатия) символы.

Нужно найти самое длинное совпадение между строкой буфера кодирования, начинающейся с символа St + 1, и всеми фразами словаря.

Полученная в результате поиска фраза кодируется с помощью двух чисел: 1) смещения (англ. offset) от начала буфера p; 2) длины соответствия, или совпадения (англ. match length) q.

Ссылки (p и q − указатели) однозначно определяют фразу. Дополнительно в выходной поток записывается символ s, следующий за совпавшей строкой буфера.

Длина кодовой комбинации (триады – p, q, s) на каждом шаге определяется соотношением

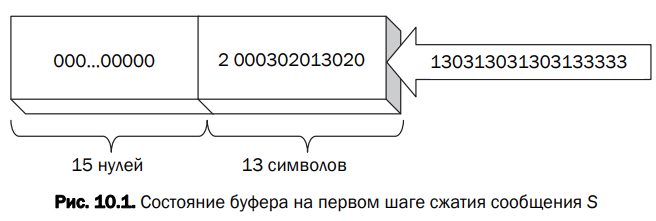


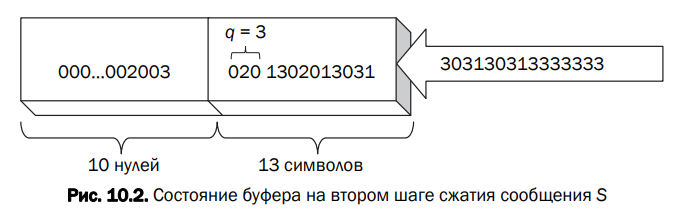
где N – мощность алфавита.

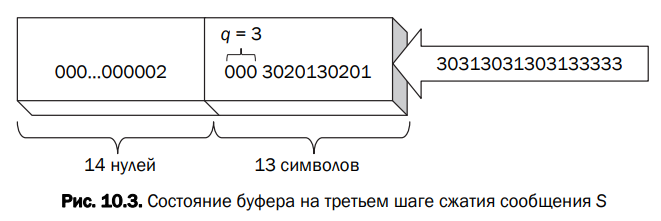
После каждого шага окно смещается на q + 1 символов вправо и осуществляется переход к новому циклу кодирования. Величина сдвига объясняется тем, что мы реально закодировали именно q + 1 символов: q – с помощью указателя и 1 − с помощью тривиального копирования. Передача одного символа в явном виде (s) позволяет разрешить проблему обработки еще ни разу не встречавшихся символов, но существенно увеличивает размер сжатого блока.

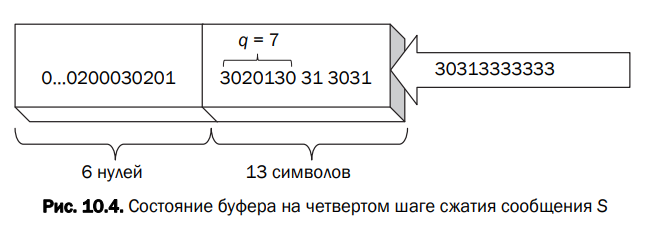
Как работает:

1. Используется скользящее окно (размер n), состоящее из:
   * Словаря (n1 символов уже закодированных),
   * Буфера предварительного просмотра (n2 символов на входе).
2. Алгоритм ищет самое длинное совпадение между началом буфера и фразами из словаря.
3. Совпадение кодируется в виде триады (p, q, s):
   * p – смещение назад в словаре,
   * q – длина совпадения,
   * s – следующий символ после совпадения.
4. Эта триада записывается в выходной поток вместо повторяющейся строки.
5. Окно сдвигается на q + 1 символ, и процесс повторяется.









**Практическое задание**

1. Разработать авторское приложение в соответствии с целью лабораторной работы. При этом предусмотреть возможность оперативного изменения размеров окон (n1, n2).

2. С помощью приложения выполнить прямое и обратное преобразования произвольного текста длиной несколько килобайт. Формат представления параметров p и q выбрать по указанию преподавателя.

3. Изменяя размеры окон, оценить скорость и эффективность (используя соотношения на с. 76) выполнения операций сжатия/ распаковки.

4. Результаты оформить в виде отчета по установленным правилам.

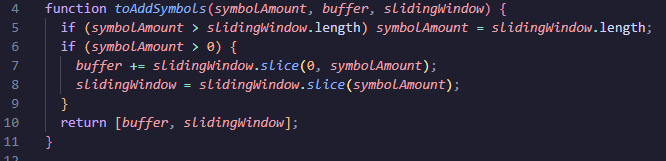


Рисунок 1.1 - Добавление символов в буфер и удаление из скользящего окна

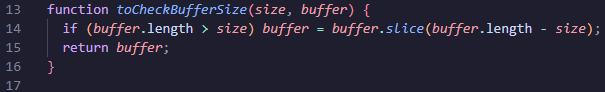


Рисунок 1.2 - Проверка и обрезка буфера до нужного размера

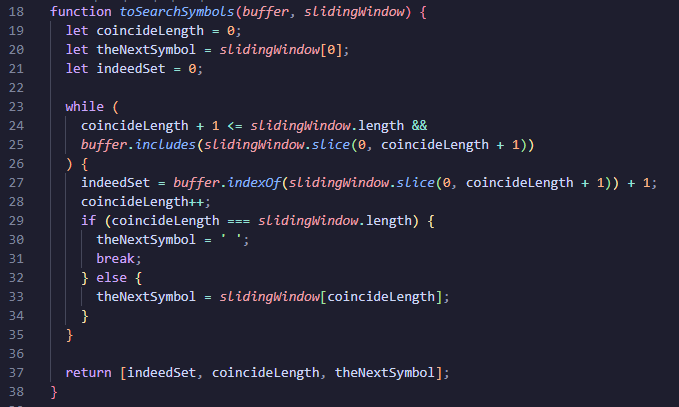


Рисунок 1.3 - Поиск совпадающей подстроки в словаре

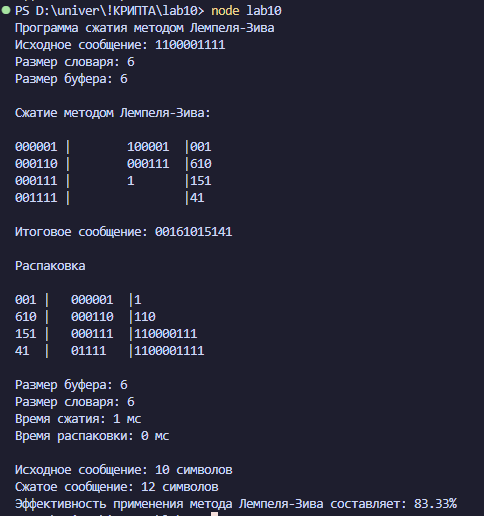


Рисунок 1.4 – Результат работы

**Вывод**

Метод Лемпеля – Зива демонстрирует высокую эффективность при работе с повторяющимися последовательностями символов и позволяет существенно снизить объём передаваемых/сохраняемых данных. Алгоритм хорошо подходит для реализаций, ориентированных на реальное применение в системах сжатия. Было установлено, что увеличение размера словаря способствует лучшему сжатию за счёт увеличения шансов найти более длинные совпадения, однако при этом возрастает время обработки. Оптимальный баланс между скоростью и степенью сжатия достигается при правильно подобранных параметрах окон.