Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Защита информации и надёжность информационных систем

Студент: Скалкович С.Л.

ФИТ 3 курс 2 группа

Преподаватель: Нистюк О.А.

**Лабораторная работа №9**

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

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

Задачи:

1. Закрепить теоретические знания по алгебраическому описанию и использованию методов сжатия/распаковки (архивации/ разархивации) данных на основе метода Лемпеля − Зива.

2. Разработать приложение для реализации метода Лемпеля − Зива.

1. **Теоретические сведения**

В 1977 г. Авраам Лемпель и Якоб Зив выдвинули идею формирования «словаря» общих последовательностей анализируемых (сжимаемых) данных. При этом сжатие данных осуществляется за счет замены записей соответствующими кодами из словаря.

Классический алгоритм Лемпеля − Зива – LZ77, названный так по году представления метода, формулируется следующим образом: «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». Известный метод сжатия RLE, который заключается в записи вместо последовательности одинаковых символов одного символа и их количества, является подклассом LZ77.

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 (англ. sliding dictionary), 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.

Пусть к текущему моменту времени закодировано t символов: S1, S2, ..., St. Тогда словарем будут являться n1 предшествующих символов: St − (n1 − 1), St − (n1 − 1)+1, …, St.

В буфере находятся ожидающие кодирования (сжатия) символы St+1, St+2, …, St+n2. Если n2 ≥ t, то словарем будет являться вся уже обработанная часть входной последовательности.

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

Эти фразы могут начинаться с любого символа St − (n1 − 1), St − (n1 − 1) + 1, …, St, выходить за пределы словаря, вторгаясь в область буфера, но должны лежать в окне. Буфер не может сравниваться сам с собой. Длина совпадения не должна превышать размера буфера. Полученная в результате поиска фраза St − (р − 1), St − (р − 1) + 1, St − (р − 1) + (q − 1) 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 *l(с ) = logN n1 +logN n2 + 1*.

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

1. Практическая часть

**Задание 1**

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

Исходя из предоставленного условия необходимо разрешить изменять размер буфера словаря и буфера данных. Для этого используются две переменные dictionaryLength и wordLength.

Данные размеры должны регулироваться таким образом, чтоб «Стоимость» данных и время, которое мы затрачиваем на сжатие и распаковку были оптимальны.

**Задание 2**

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

Входными данными будет являться ФИО студента. В данном случае используется «Скалкович Станислав Леонидович». Размер составляет 8 символов в первом и во втором буфере.

На рисунке 1 отображен процесс преобразования.

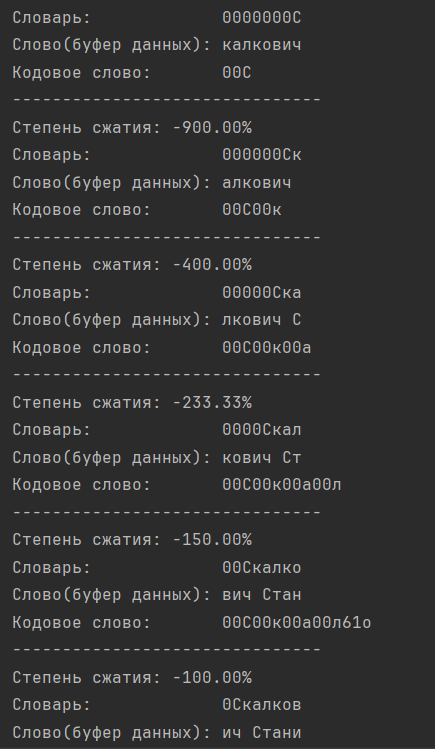


Рисунок 1 – Процесс сжатия

Для распаковки или обратного преобразования используются кодовые слова, полученные на каждом этапе сжатия. Заполняем буфер словаря нулями и будет добавлять символы и сдвигать его в соответствии с кодовым словом на каждом шаге, где p – позиция элемента, q – количество сдвигаемых символов и S – сам символ (либо следующим за ним). Процесс обратного преобразования представлен на рисунке 2.

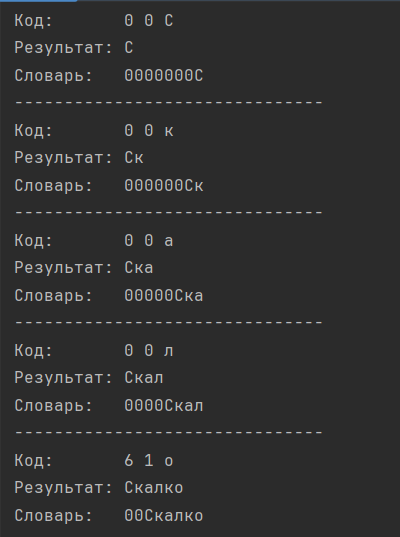
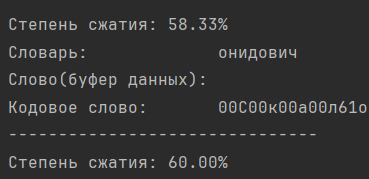


Рисунок 2 – Процесс обратного преобразования

**Задание 3**

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

Оценить эффективность сжатия можно по формулам, представленным в лабораторной работе №8. На рисунке 3 вычисляется эффективность сжатия для окон, размером 8x8 и 4x4.



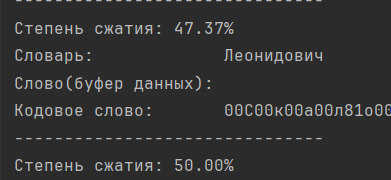


Рисунок 3 – изменение степени сжатия при разных параметрах

**Вывод**

Алгоритм Лемпеля-Зива (LZ77) является методом сжатия данных, который использует словарь для замены повторяющихся последовательностей символов. Он основан на идее о том, что при обнаружении повторяющейся последовательности байт в предыдущих данных, вместо повторной записи этой последовательности создается ссылка на уже существующий фрагмент данных.

В процессе сжатия данных алгоритм LZ77 разделяет входные данные на две части: словарь (или скользящий словарь) и буфер предварительного просмотра (lookahead buffer). Словарь содержит уже закодированные символы, а буфер предварительного просмотра содержит символы, которые еще не были закодированы. Алгоритм обрабатывает входные данные, перемещая окно кодирования по входному потоку и при необходимости записывая ссылки на уже закодированные фрагменты данных.

Для каждого символа в буфере предварительного просмотра алгоритм ищет наибольшее совпадение с уже закодированными данными в словаре. Это совпадение характеризуется смещением (offset) от начала словаря и длиной совпадения. Информация о смещении и длине совпадения записывается в выходной поток, после чего окно кодирования сдвигается на количество символов, равное длине совпадения плюс один символ.

Таким образом, алгоритм Лемпеля-Зива эффективно сжимает данные, заменяя повторяющиеся фрагменты ссылками на уже имеющиеся, что позволяет уменьшить объем передаваемой информации.