## Лабораторная работа №6

## Кодирование с динамическим словарем

***Определение 1*** (Словарь, индекс) Словарь  основанный на алфавите  – это последовательность отдельных слов множества : , Мы говорим, что индекс  это .

В арифметическом кодировании мы наложили «алфавитный порядок» на множество символов , и создали соответствующий «лексикографический порядок» в . Например, если  содержит три символа в порядке , то множество  является словарем, упорядоченным следующим образом ,

а индекс элемента  – 7. Это пример статического словаря.

В этом параграфе мы рассматриваем метод кодирования, в котором словарь является динамическим – другими словами, он построен таким образом, что содержит те строки, которые действительно появляются в данном сообщении. Предполагается, что источник стационарный, поэтому строки символов, которые возникают в начале сообщения типичны для тех, которые будут появляться позже, но никакий конкретной информации о распределении символов не требуется.

Опишем систему Лемпеля-Зива-Велча, также известную как **LZW**. Алфавит  имеет размер , содержит символаы . И кодировщик и декодировщик начинаются со словаря , где .

Если сообщение, где , кодировщик кодирует сообщение, , и в то же время создает словарь , добавляя к  определенные строки , которые встречаются в . Правило кодирования заменяет каждую такую строку по ее индексу в . Декодер использует последовательность чисел , а также исходный словарь , чтобы реконструировать  и (в данном случае) .

***Определение 2*** (**LZW** кодирование) Предположим, что дано сообщение  в алфавите . Пусть , где. Правила кодирования LZW создают  серией шагов.

* Шаг 1 Первый символ  (называется ) – это запись  в . Кодируется  путем определения . Строка  (называется ) не входит в . Определим .
* Шаг  () Полагаем, что Шаги  завершены. Это означает, что код  для начального сегмента  – , и словарь , созданы. Найдем самую длинную строку  вида  такую, что  принадлежит множеству , можно сказать . Такая строка, безусловно, существует, потому что  единственный символ, и он уже входит в исходный словарь . По определению  не входит в множество . Кодируем сегмент  –  с помощью , и определяем .
* Повторять эту процедуру, пока не будет достигнут конец сообщения.

***Пример 1*** Возьмем алфавит  и применим правила кодирования LZW к сообщению .

*Решение* Начнем со словаря . Правило на Шаге 1 предписывает кодировать А, которое имеет индекс 1, и добавить АВ в словарь: , .

На Шаге 2, B находится в, но BR – нет, поэтому мы кодируем B индексом 2 и добавляем BR: , .

Шаги 3, 4, 5, 6, 7 продолжаем таким же образом, пока не достигнем стадии:

, .

На Шаге 8 AB уже в , с индексом 6, но ABR нет в . Таким образом, результат ,

.

На Шаге 9 RA входит в словарь, с индексом 8, и сообщение заканчивается, таким образом

.

На первый взгляд это не очевидно, что код LZW однозначно декодируемый. Декодировщик должен повторять кодировщик путем добавления новой записи в словарь на каждом шаге, и всегда, кажется, что один шаг позади. Эта проблема может появиться на Шаге 1, и стоит обратить внимание на этот случай.

Предположим, что декодировщик задан  и кодированным сообщением  .

Шаг 1 в правилах кодирования LZW говорит, что , где  в , поэтому  декодируется путем установки . Чтобы завершить Шаг 1, декодировщик должен также принять решение о том, как увеличить  путем добавления новой строки , и это требует рассмотерния .

Пусть . Если  процедура проста, потому что  в  и . Таким образом, сообщение должно начинаться с , а декодировщик повторяет кодировщик, добавив эту строку в словарь. Но что если ? В этом «тяжелом случае»  может быть только , потому что  кодировано с использованием словаря . Согласно правилу для Шага 1, , где , . Сейчас декодировщик может определить , поскольку  – это кодированный вид . Так, , а  и новая запись в словаре это фактически .

***Теорема 1*** Код LZW, созданный на основе определения 4.22 – однозначно декодируемый.

***Пример 2*** Дано: словарь , декодировать сообщение 214468331 .

*Решение* На Шаге 1 и, поэтому. Также  и , поэтому новая запись в словаре – MI. Таким образом

.

На Шаге 2  и , поэтому. Также  и , поэтому новая запись в словаре – IS. Таким образом,

.

Шаги 3 и 4 подобны, в результате чего

.

На 5 шаге, и , так . Также  и , так новая запись в словаре ISS. Таким образом,

.

На 6 шаге, и , так . Также  и , так новая запись в словаре IP. Таким образом,

.

На 7 шаге, и , так . Также  и , так новая запись в словаре PP. Таким образом,

.

На 8 шаге, и , так . Также  и , так новая запись в словаре PI. Таким образом,

.

На 9 шаге и , так . Таким образом,

.

На данный момент читатель вероятно спросит, достигают ли по факту LZW правила значительного сжатия. В шуточных примерах, приведенных выше, только малая степень сжатия достигается в какой-то мере. Например, сообщение  имеет длину 11 символов, а кодированный вид 125131468 имеет длину 9 символов, но уменьшение достигается за счет нетривиальных вычислений. Однако примеры однозначно полагают, что сообщение, увеличиваясь в длине, с неизбежным большим повторением строк, дает возможно некоторое улучшение.

На практике LZW кодирование работает хорошо, и его широкое применение подтверждает, что принцип правилен. Изложение правил, приведенных выше, предназначено для того, чтобы сделать их легкими для понимания, но можно упростить расчеты с использованием более эффективных структур данных.

***Задание.*** Закодировать и декодировать по алгоритму LZW строку текста 10 символов, в котором существуют повторяющиеся символы, используя алфавит  из 4-5 символов.