Лабороторная №1

Выполнил

Кочетков Никита Олегович, M4139

Реализовано

`BWT + MTF + монотонный код`

Алгоритм кодирования

Преобразования Барроуза-Уиллера

Перед тем как закодировать исходный файл, мы сначала преобразуем данные для лучшего сжатия.
Первым алгоритмом преобразования будет: **алгоритм преобразования Барроуза-Уиллера**
Сначала мы прочитаем файла и разобьём его на блоки размером 250. Для больших блоков это было бы затратно по времени и памяти. Для каждого блока сделаем следующее: рассмотрим все циклические сдвиги этого блока и запишем в массив после чего отсортируем в лексикографическом порядке, найдём номер исходного блока в этом массиве и запомним его. Возьмём в виде преобразованной последовательности последний столбец в отсортированном массиве. Так для каждого блока у нас будет новый преобразованный блок и индекс.

Запишем в начало выходного файл суммарное число блоков (4 байта) + по 1 байту на каждый блок.

MTF

После предыдущего преобразования преобразуем теперь данные методом стопки книг, так как после предыдущего преобразования у данных появляется больше одинаковых символов подряд, поэтому этот метод будет хорошо сжимать.

Возьмём упорядоченный алфавит. Теперь для каждого элемента последовательности полученной в предыдущем шаге мы сделаем поочереди следующее: найдём этот элемент в алфавите и выпишем его индекс, символ из алфавита перемещается в голову алфавита (остальные символы левее него делают сдвиг направо). В результате у нас будет список индексов с которыми мы теперь будем дальше работать.

Монотонное кодирование

Для последовательности индексов мы применим монотонное кодирование. Для каждого индекса мы получим набор бит однозначно задающих его. Для маленьких чисел число бит будет меньше байта. Поэтому мы все преобразованные числа в биты запишем в последовательность бит. Потом мы преобразовали массив бит в массив байт и записали в выходной файл.

Алгоритм декодирования

Монотонное декодирование

Для начала мы считаем первые 4 байта - это число N блоков преобразованных Барроуза-Уиллера. Потом считаем N бит - это числа для каждого блока. Остальные данные пойдут в декодирование из монотонного кода.

Произведём над этими данными преобразование, обратное кодированию монотонным кодом.

MTF

После предыдущего шага у нас имеет последовательно индексов. Сделаем так же как и с прямым преобразованием. Только по индексу мы будем возвращаем соответствующий символ алфавита.

Преобразования Барроуза-Уиллера

Разобьём последовательность на блоки по 250. Теперь восстановим изначальный блок

Повторим N раз в зависимости от размера блока: начальная строка блока записывается в первый столбец массива блоков, после чего массив сортируется. В результате имеем такую же матрицу, как при кодировании. Возьмём блок по индексу, который лежал соответственно в начале файла.

Таким образом мы получим изначальную последовательность, которую запишем в выходящий файл.

Результаты

Файл	H(X)	H(X | X)	H(X | XX)	Средние затраты на символ (в битах)	Размер сжатого файла (байт)	Размер исходного файла (байт)	Среднее время на декодирование
bib	5.2007	3.3641	2.3075	5.930847287009824	82484	111261	96 секунд
book1	4.5272	3.5845	2.8141	5.476210731154011	526244	768771	398 секунд

Файл	H(X)	H(X I X)	H(X I XX)	Средние затраты на символ (в битах)	Размер сжатого файла (байт)	Размер исходного файла (байт)	Среднее время на декодирование
book2	4.7926	3.7452	2.7357	5.3109079717642125	405525	610856	432 секунды
geo	5.6464	4.2642	3.4578	7.134453125	91321	102400	78 секунд
news	5.1896	4.0919	2.9228	5.84367914846901	275463	377109	210 секунд
obj1	5.9482	3.4636	1.4006	5.780877976190476	15539	21504	21 секунда
obj2	6.2604	3.8704	2.2654	5.343619081575599	164860	246814	210 секунд
paper1	4.9830	3.6460	2.3318	5.4548259062094395	36248	53161	90 секунд
pic	1.2102	0.8237	0.7052	1.8372615039281706	117864	513216	204 секунд
paper2	4.6014	3.5223	2.5137	5.3324249686735845	54790	82199	75 секунд
progc	5.1990	3.6033	2.1341	5.178763474792356	25642	39611	40 секунд
progl	4.7701	3.2116	2.0436	4.284509951707004	38371	71646	65 секунд
progp	4.8688	3.1875	1.7552	4.468458251483424	27581	49379	20 секунд
trans	5.5328	3.3548	1.9305	5.0177277336037145	58767	93695	33 секунды

Суммарное значение всех сжатых файлов

Суммарное значение размера всех сжатых файлов в байтах: 1 920 699