

Coordinate Scheme of Coding.

КСК относится к АС без потери качества и, как у других АС, у него возможны различные реализации, которые наиболее подходят в том или ином случае. Для того, чтобы понять принцип работы КСК полезно рассмотреть структуру архива, получающегося после работы алгоритма.

Структура архива.

$$A_1 [fin_bit, Ad_1^1, fin_bit, Ad_2^1, fin_bit, \dots, Ad_{m-1}^1, fin_bit], \dots, A_n [fin_bit, Ad_1^n, fin_bit, \dots, Ad_{k-1}^n, fin_bit]$$

A_i - i -ая буква алфавита.

Ad_k^i - адрес $k+1$ – го экземпляра i – ой буквы (это не опечатка: именно $k+1$ – го, а не k – го – почему: будет разъяснено позже).

fin_bit - этот бит нужен для того, чтобы распаковщик знал все ли экземпляры буквы A записаны в файл.

Принцип работы распаковщика.

Все буквы в архиве записаны в том порядке, в котором первые вхождения этих букв встречаются в исходном файле (распакованном файле), если его просматривать с начала до конца: первый экземпляр каждой буквы записывается по ближайшему от начала файла не занятому адресу.

Условия получения положительного результата.

Допустим: у нас файл размером 4Гб и размер каждой буквы 4 байта – для прямой адресации нам нужно $\log_2 \frac{\text{длина_файла}}{\text{длина_буквы}} = 30$ бит.

Выведем формулу для подсчёта сколько бит тратится на букву с n вхождениями в архиве: (кол-во $fin_bit=n$)+(кол-во адресов= $n-1$)*(длина поля адреса= 30)+(длина буквы= 32) = $= n + 30(n-1) + 32$. Из формулы видно, что на букве с одним вхождением: идёт потеря одного бита; с двумя: результат нейтральный; начиная с трёх: получаем эффект сжатия. Кстати, заметим, если бы мы писали b в архиве адреса первых вхождений – для положительного эффекта, при данных условиях, потребовалось бы не менее 33 вхождений буквы.

Dynamic Scheme of Coordinate Coding.

DSCC является расширенной версией CSC, в которой содержатся две нижеследующие методики или какая – либо из них.

Сужение поля адресации.

При заполнении файла на процент равный $\sum_i \frac{1}{2^i}, i = 0, \dots, n$ можно сократить длину поля на 1

бит: $i=0$ – длина_поля=длина_поля; (50%) $i=1$ – длина_поля --; (75%) $i=2$ – длина_поля -- и.т.д. Возникает закономерный вопрос как(?!?) – как при сужении обеспечить адресацию, ведь объём

файла прежний. Использование битовой карты позволяет сводить адресацию из общего диапазона к диапазону свободных адресов.

Привязка вторичных кодов.

По мере заполнения файла, образуется перечень использованных кодов, каждый из которых больше встречаться не будет – все эти коды заносятся в массив. Итый элемент массива привязывается к i -му свободному адресу в файле. Таким образом можно получить доп. процент сжатия за счёт экономии на некотором количестве финальных бит. Ну, и ещё, что стоит отметить – массив должен переопределяться заново на каждом шаге сужения, впрочем, это очевидно.

Dual Scheme of Coding.

Все буквы разбиваются на две категории: одна кодируется координатной схемой, а другая словарной. При распаковке сначала декодируется координатка, - затем все свободные ячейки заполняются из второй части архива (словарной).

ABC System With Limit of Losses.

Все буквы разбиваются на две категории: в первой каждая буква получает уникальный код – во второй все имеют один и тот же код минимальной длины. В архиве буквы второй категории должны располагаться в той же очередности, что и в исходном файле.

Ремарка.

Координатные схемы чрезвычайно гибки: могут быть адаптированы как под текстовые файлы, так и под различные бинарные файлы с заведомо малой избыточностью. Мной разрабатываются, на основе расширенного определения избыточности, более совершенные алгоритмы сжатия.

Licence:

Free usage of these algorithms is possible only in the free software.

Usage in the commercial apps/hardware must be licenced.

Any publication of these algorithms must have a reference to developer's web.

Date: 12.03.2007

WWW: <http://xproject-all.narod.ru/prgsale.htm>
(C) Evgeney Knyazhev

2007