УДК 004.627

МОДИФИКАЦИЯ АЛГОРИТМА СЖАТИЯ ИНФОРМАЦИИ RLE

Семионов А.А.
Гомельский государственный технический университет им. П.О. Сухого, г. Гомель
Научный руководитель: Мурашко И.А., д.т.н., доцент

Бурный рост объемов информации, циркулирующих в компьютерных системах, приводит к возрастанию времени отклика системы. Сжатие данных снижает требования к компьютерным системам и каналам передачи данных. Поэтому актуальным является разработка новых или повышение эффективности существующих алгоритмов сжатия данных.

В алгоритме RLE достаточно несложно разобраться, если посвятить этому некоторое время. Суть метода заключается в кодировании повторов. Это значит, что последовательность одинаковых элементов записывается в виде пары количество×значение.

Например, строка вида «TTTKGGGGG» будет преобразована в «3×T, K, 5×G». Это и есть суть метода.

Пример реализации:

Допустим, есть массив целочисленных чисел от 0 до 255:

{0, 0, 0, 0, 0, 0, 7, 9, 3, 4, 4, 4, 4, 4, 4, 4, 70,
70, 70, 70, 0, 4, 4, 4, 4, 255, 255, 255, 255, 0}

Переведем его в HEX форму:

{00, 00, 00, 00, 00, 00, 07, 09, 03, 04, 04, 04, 04, 04, 04,
04, 46, 46, 46, 46, 00, 04, 04, 04, 04, FF, FF, FF, FF, 00}

Нашей целью является сжатие набора данных. Для этого необходимо выявить закономерность: очень часто попадаются последовательности из одинаковых элементов. Алгоритм RLE будет как нельзя кстати для сжатия такого набора.

Зашифруем данный набор с помощью уже описанного RLE алгоритма:

6×0, 7, 9, 3, 7×4, 4×70, 0, 4×4, 4×255, 0

Чтобы представить результат в понятном для компьютера виде, нужно как-то отделять одиночные байты от кодируемой цепочки. Поскольку весь диапазон значений байта используется данными, то просто так выделить какие-либо диапазоны значений под эти цели не удастся.

Существует довольно простой способ разделять одиночные байты от кодируемой цепочки. Для этого надо указывать количество не только для повторяемых, но и последующих далее одиночных элементов. Тогда можно будет заранее узнать, где какие данные.

Давайте выделим один бит в служебных байтах под тип последовательности: 0 – одиночные элементы, 1 – одинаковые. Возьмём для этого старший бит байта.

Оставшиеся семь бит будут использоваться для хранения длины последовательности. То есть максимальная длина кодируемой последовательности будет 127 байт. Можно, конечно, выделить для служебной информации два байта. Но настолько длинные последовательности в нашем примере не встречаются.

Получается, в выходной поток вначале пишется длина последовательности, затем одно повторяемое значение или цепочка неповторяемых элементов указанной длины.

Сделаем наш алгоритм более эффективным. Не тяжело заметить, что при таком алгоритме есть несколько неиспользуемых значений. Не может быть последовательностей

Репозиторий БРГТУ

с нулевой длиной. Поэтому максимальная длина может быть увеличена до 128 байт, отнимая от длины единицу при кодировании и прибавляя при декодировании. Таким образом, можно кодировать длины от 1 до 128 вместо длин от 0 до 127.

Можно также заметить, что не бывает последовательностей единичной длины. Поэтому от значения длины таких последовательностей при кодировании можно отнимать единичку, увеличив тем самым их максимальную длину до 129. То есть цепочки одинаковых элементов могут иметь длину от 2 до 129.

Теперь можно закодировать данные снова, но уже в компьютерном виде. Служебные байты будут записываться в виде [T|L], где T – тип последовательности, а L – длина. Учитывается, что длины записываются в изменённом виде: при T=0 отнимается от L единица, при T=1 – двойка.

[1|4], 0, [0|2], 7, 9, 3, [1|5], 4, [1|2], 70, [0|0], 0, [1|2], 4, [1|2], 255, [0|0], 0

Последним шагом надо сохранить полученный результат как массив байт. Например, пара [1|4], упакованная в байт, будет выглядеть следующим образом:

[1|4] = 10000100 = 0x84

Конечный вариант кодирования:

84 00 02 07 09 03 85 94 82 70 00 00 82 04 82 FF 00 00

Таким образом, при помощи алгоритма сжатия RLE, входной массив, состоящий из 30 байт, поместился в 18-ти байтах.

Эффективность алгоритма зависит от способа его реализации. Для разных данных можно разрабатывать разные варианты кодирования. Например, при кодировании изображений можно сделать цепочки переменной длины: выделять один бит под индикацию длинной цепочки, и если он выставлен в единицу, то хранить длину и в следующем байте тоже. Так длина коротких цепочек уменьшается до 65, но зато длина длинных расширяется до 16385 элементов.

УДК 004.413:687.1/.4

АВТОМАТИЗАЦИЯ УЧЕТА ПОСТУПЛЕНИЯ ГОТОВОЙ ПРОДУКЦИИ НА СКЛАД НА ПРИМЕРЕ ШВЕЙНОГО ПРОИЗВОДСТВА

Стельмашук Н.П., Гучко И.М.

Брестский государственный технический университет, г. Брест

Цель данной работы – создание прикладной разработки для автоматизации отдельного участка работы на складе готовой продукции (обработка сопроводительных листов) частного швейного предприятия, учет по которому велся «вручную», и данные учета поступали в бухгалтерию на бумажных носителях.

Склад готовой продукции является важнейшим логистическим звеном, органически связывающим производство и сбыт продукции предприятия. В условиях производства продукции с номенклатурой, обычно составляющей более сотни товарных позиций и цикличности технологических процессов, обеспечение на складах быстроты оформления операций позволяет снизить трудозатраты и ускорить их выполнение. Более того, при количестве обрабатываемых документов до тысячи в год, а также когда руководство предприятия считает, что данный рабочий процесс еще не нуждается в более сложной автоматизации, но, тем не менее, уже занимает значительную часть рабочего времени,

Репозиторий БрГТУ