Служебное ПО

Архивация данных

Сергеева Е.Ю.

Как хранение, так и передача информации обходятся участникам информационного процесса недешево. Зная стоимость носителя и его емкость, нетрудно подсчитать, во что обходится хранение единицы информации, а зная пропускную способность канала связи и стоимость его аренды, можно определить затраты на передачу единицы информации. Полученные результаты обычно составляют вполне значимые величины как для корпоративных пользователей, так и для индивидуальных. В связи с этим, регулярно возникает необходимость сжимать данные перерд тем, как размещать их в архивах или передавать по каналам

- **Архивация** - процесс сжатия файлов.
- **Архиватор** - программа, позволяющая архивировать и разархивировать файлы

Тогда и сейчас…

Много лет назад такие программы были очень актуальны, т.к. объем жестких дисков был невелик и временно не используемые программы и данные приходилось хранить на диске в сжатом виде. Применялись архиваторы и для переноса информации с одного компьютера на другой на дискете, если объем переносимых файлов превышал вместимость дискеты.

В настоящее время, казалось бы, ситуация другая - объемы жестких дисков у рядовых пользователей настолько велики, что многие просто не представляют, как использовать весь потенциал таких больших хранилищ данных, появились и используются компакт-диски и другие сменные носители информации большой вместимости, у многих работает электронная почта. Стоит ли в таких условиях задумываться об архиваторах и следует ли вообще их использовать?

Стоит, и для этого есть целый ряд причин.

Зачем?

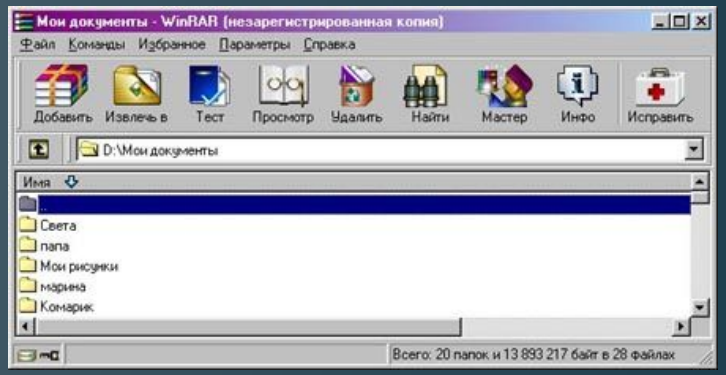
- Во-первых, при передаче файлов по электронной почте критичным является каждый килобайт, да и, кроме того, при пересылке большого числа файлов (особенно со сложной структурой каталогов) проще в письмо вложить всего один файл - архивный (сжатый).
- Во-вторых, пропускная способность локальных сетей ограниченна. При пересылке больших массивов информации по сети рекомендуется использовать архиваторы не только для уменьшения объема передаваемой информации, но и с целью упрощения конечной проверки результатов передачи: после копирования проще проверить корректность архива, нежели проверять сотни файлов на целостность, просматривая их содержимое.

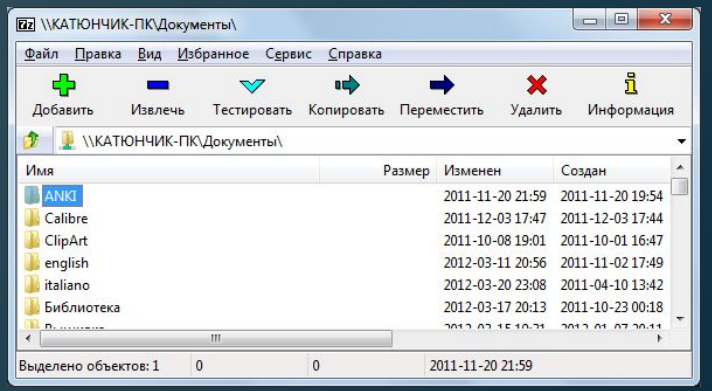
Зачем?

- В-третьих, привод компакт-дисков с возможностью записи (пишущий CD-ROM) стал стандартом для персональных компьютеров среднего класса, но часто встает проблема, когда на компакт-диск необходимо перенести очень сложную структуру каталогов, тогда как файловая система компакт-диска не допускает более чем восьмикратный уровень вложенности папок.
- В-четвертых, многие архиваторы являются очень эффективными кодировщиками, позволяющими скрыть конфиденциальную информацию от чужих глаз, "запаковав" файлы и установив на них пароль.

Архиваторы

- На сегодняшний день популярными являются следующие программы-архиваторы: WinRar, WinZip, 7-Zip. Как видите интерфейсы этих программ схожи – основные команды – Добавить и Извлечь.





Отличия

- Программы-архиваторы отличаются друг от друга методами сжатия и эффективностью их работы.
- Коэффициент сжатия — основная характеристика алгоритма сжатия. Она определяется как отношение объёма исходных несжатых данных к объёму сжатых, то есть:
- $k = S_o / S_c$, где k — коэффициент сжатия, S_o — объём исходных данных, а S_c — объём сжатых. Таким образом, чем выше коэффициент сжатия, тем алгоритм эффективнее.

Виды сжатия

Без потери качества

можно обратить процесс сжатия (вернуть информацию в исходном виде)

С потерей качества

качество теряется навсегда

Сжатие графики

основано на том, что человеческий глаз способен воспринимать лишь ограниченное число градаций основных цветов. Например, формат GIF содержит всего лишь до 256 цветов, а значит для хранения одного пикселя отводится не более 1 байта, тогда как полноцветные изображения отводят - 3-4 байта.

Сжатие звука

в рамках наиболее популярного ныне формата MP3 основывается на удалении наименее значимых деталей звучания (согласно психоакустической модели). Например, человек способен воспринимать звук узкого частотного диапазона, не воспринимаются сигналы, мощность которых ниже определенного пограничного значения и т.

Метод Хаффмана

Одним из самых эффективных способов кодирования без потери качества является метод Хаффмана. Рассмотрим его на примере.

Известно, что для кодирования одного символа требуется 8 бит .
Закодируем слово "*архив*". Вот как выглядят коды этих символов в кодовой таблице ASCII:
а - 11100000
в - 11100010
и - 11101000
р - 11110000
х - 11110101
Используя эти данные слово "архив" можно закодировать следующим образом:

1110000011110000111101011110100011100010

Это слово будет занимать на диске 40 бит (5 символов по 8 бит каждый).
Попробуем уменьшить объем информации. Для этого нужно каждому символу поставить в соответствие код покороче.

В некотором тексте, который предстоит закодировать, подсчитали сколько раз встречается каждый символ:

а - 42 в - 29 и - 19 р - 12 х - 3

Шаг 1.

Запишем эти буквы в порядке убывания частот:

$\{(а, 42), (в, 29), (и, 19), (р, 12), (х, 3)\}$

Шаг 2.

На следующей строке запишем набор, полученный из предыдущего набора следующим образом:

- вместо двух последних символов с их частотами запишем новый элемент, у которого вместо названия символа будет записаны названия двух последних элементов, а вместо частоты - сумма частот.
- отсортируем полученный набор по убыванию.

В результате на второй строке будет записано:

$\{(a, 42), (b, 29), (u, 19), (p, 12), (x, 3)\}$

$\{(a, 42), (b, 29), (u, 19), (p\,x, 15)\}$

Шаг 3.

Переходим на шаг 2 до тех пор, пока набор не будет состоять только из одного элемента.

В результате будет записано:

{(а , 42), (б , 29), (и , 19), (р , 12), (х , 3)}

{(а , 42), (б , 29), (и , 19), (р х , 15)}

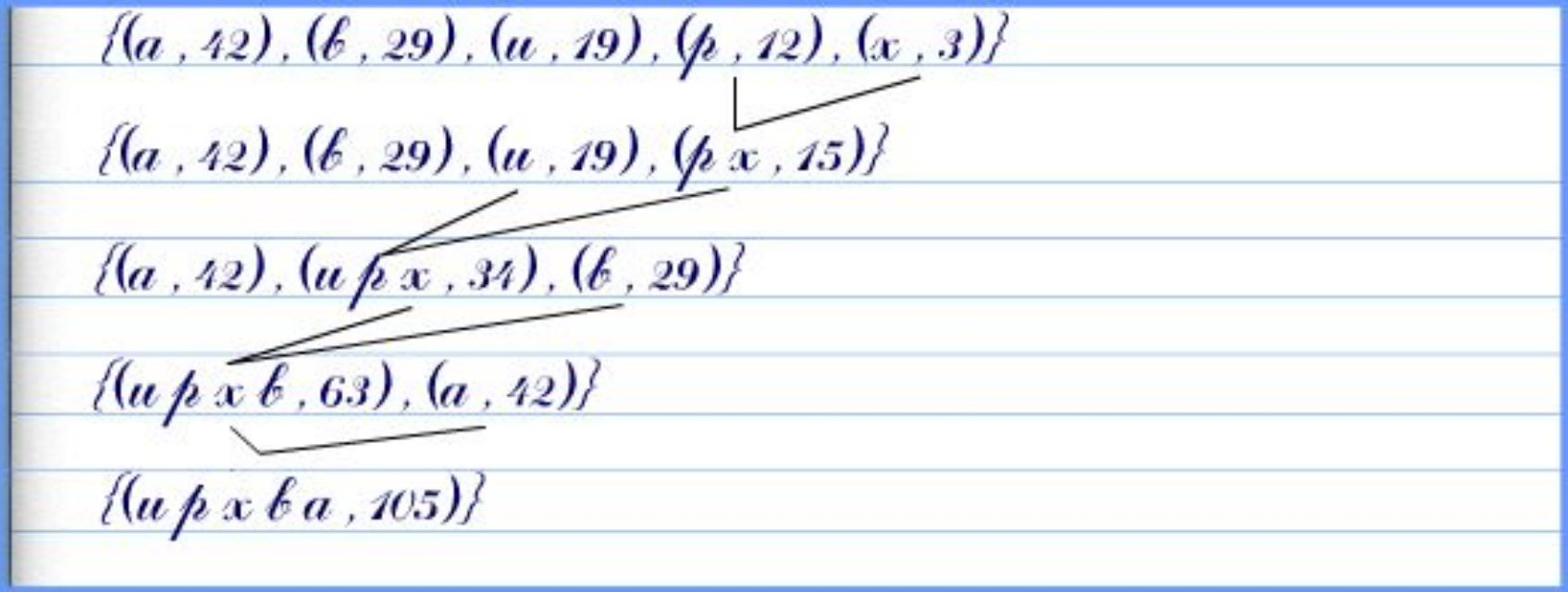
{(а , 42), (и р х , 34), (б , 29)}

{(и р х б , 63), (а , 42)}

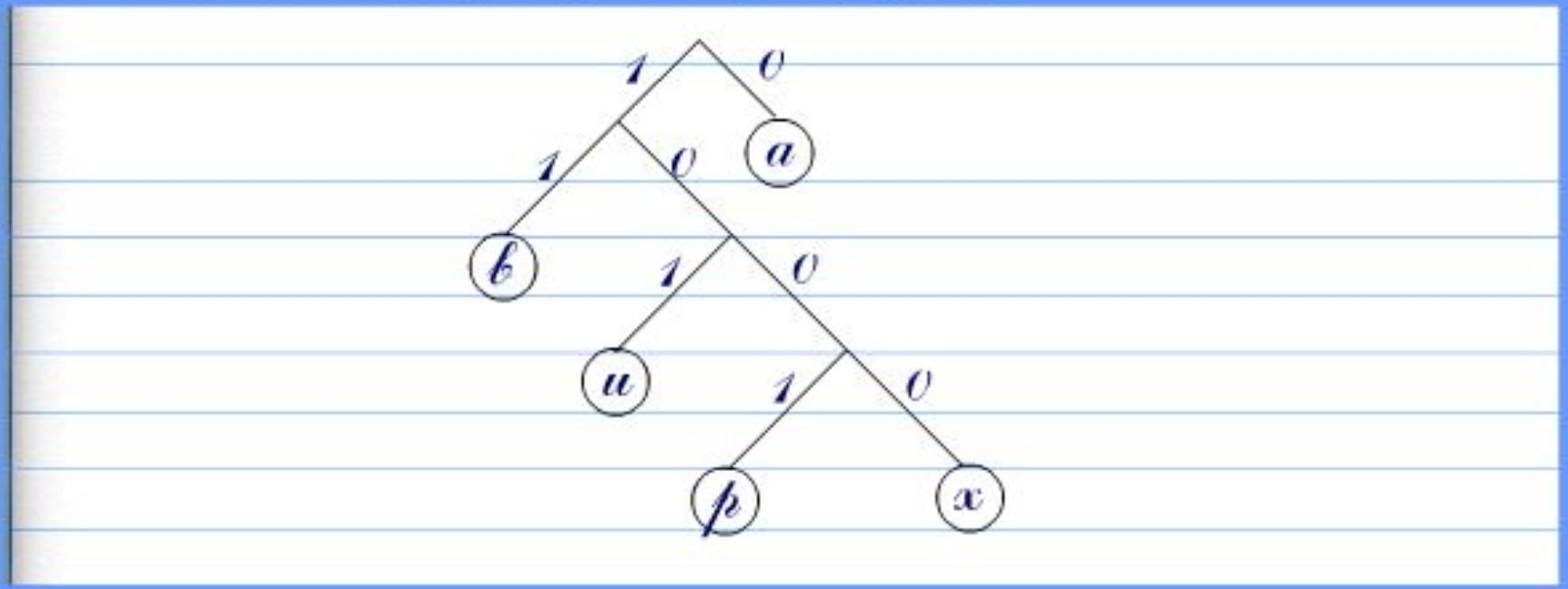
{(и р х б а , 105)}

Что же в итоге?

Если соединить чертой каждый из элементов, состоящих из более чем одного символа с теми элементами предшествующих наборов, то мы получим своего рода дерово:



Перепишем это дерево еще раз, только сверху вниз. Каждой паре линий припишем числа 1 (например, слева) и 0 (справа):



Двигаясь к каждой букве вниз от вершины дерева выписываем встречающиеся 0 и 1 - это и будет новый код символа. Итог:

а - 0 в - 11 и - 101 р - 1001 х - 1000

Новый вид кодировки слова "архив" имеет вид:

01001100010111

Его длина - 14 бит.

Задание

- Выполните этот алгоритм для задания вашего варианта.