**Архивация данных для MS DOS.**

История развития теории сжатия информации 1

Архиваторы MS DOS. 6

PKZIP 8

RAR 8

QUARK 9

GZIP 10

ARJZ 10

Сравнение вресий архиваторов MS DOS и Windows. 14

Литература. 15

# История развития теории сжатия информации

История развития MS DOS весьма похожа на историю развития всех информационных технологий. Более того, концепция MS DOS непосредственно заимствована из операционной системы UNIX. UNIX, в свою очередь, базируется на самых первых операционных системах типа OS/360 и даже IBM704. Основные алгоритмы архивации данных вначале были опробованы на UNIX, а затем (иногда процесс развития тех или иных алгоритмов осуществлялся параллельно) в MS DOS. В принципе, большинство используемых в настоящее время утилит для архивации начали свою жизнь начиная с MS DOS 2.0.

Итак, в сороковых годах ученые, работающие в области информационных технологий, ясно поняли, что можно разработать такой способ хранения данных, при котором пространство будет расходоваться более экономно. Клод Шеннон, изучая нюансы различий между семантикой (semantics) (что некая сущность значит) и синтаксисом (syntax) (как некая сущность выражается), разработал большинство базовых понятий этой теории. Понимание того, что одно и то же значение (семантика) может быть реализовано различными способами (синтаксис), приводит к закономерному вопросу: "Какой способ выражения чего-либо является наиболее экономичным?" Поиск ответа на этот вопрос привел Шеннона к мысли об энтропии, которая, проще говоря, соотносится с количеством, содержащейся в файле полезной информации. Методы сжатия пытаются увеличивать энтропию файла, то есть уменьшать длину файла, сохраняя при этом всю информацию.

Однако, Шеннон не был первым, кто задумывался о сущности информации и определении ее количества. Первый шаг на этом пути сделал в 1928 г. Хартли. Основной полученный им результат можно сформулировать примерно так: если в заданном множестве, содержащем N элементов, выделен некоторый элемент x, о котором известно лишь, что он принадлежит этому множеству, то, чтобы найти x, необходимо получить количество информации, равное log2 N. Эту формулу обычно называют формулой Хартли.
Формула Хартли является частным случаем более общей формулы Шеннона, позволяющей найти количество информации в случайном сообщении фиксированного алфавита. Пусть X1, ..., Xn - символы этого алфавита, P1, ..., Pn - вероятности их появления в тексте сообщения, тогда формула Шеннона принимает вид:

H = P1\*log2(1 / P1) + ... + Pn\*log2(1 / Pn),

где H - количество бит информации в одном символе сообщения, или энтропия символа сообщения. Это число показывает минимальное среднее число бит, необходимых для представления одного символа алфавита данного сообщения.

 В некоторых случаях алфавит сообщения может быть неизвестен, тогда выдвигаются гипотезы об алфавите сообщения. Имея разные алфавиты, можно достичь разных коэффициентов сжатия. Например, текстовый файл, если его рассматривать как последовательность битов, имеет энтропию порядка 0.7 - 0.9, если как последовательность байтов, - 0.5 - 0.7, хотя популярные программы сжатия уменьшают размеры текстовых файлов до 0.3 - 0.4 от исходного размера.

Доказательство Шеннона не было конструктивным, т.е. не содержало способа построения этих оптимальных кодов, а лишь показывало их существование. До появления работы Шеннона, кодирование символов алфавита при передаче сообщения по каналам связи осуществлялось одинаковым количеством бит, получаемым по формуле Хартли. С появлением этой работы начали появляться способы, кодирующие символы разным числом бит в зависимости от вероятности появления их в тексте. Например, часто в файлах некоторые значения байта встречаются чаще других. Таким образом, за счет использования для каждого значения байта кода различной длины можно значительно уменьшить общий размер данных. Эта базовая идея лежит в основе алгоритмов сжатия Шеннона-Фано (Shannon-Fano) и Хаффмана (Huffman). Подобные алгоритмы выбирают более короткие коды для часто встречающихся и более длинные для редко встречающихся значений байта. Обычно текстовые файлы (в которых одни значения байтов повторяются гораздо чаще других) они сжимают довольно хорошо.

Более тридцати лет алгоритм сжатия Хаффмана и его варианты оставались наиболее популярными методами. Однако в 1977 два исследователя из Израиля предложили совершенно другой подход к этой проблеме. Абрахам Лемпел и Якоб Зив выдвинули идею формирования "словаря" общих последовательностей данных. При этом сжатие данных осуществляется за счет замены записей соответствующими кодами из словаря. Существуют два алгоритма, в настоящее время известные как LZ77 и LZ78. Они уже не требуют включения словаря данных в архив, так как если вы формируете ваш словарь определенным способом, программа декодирования может его восстанавливать непосредственно из ваших данных. К сожалению, LZ77 и LZ78 тратят много времени на создание эффективного словаря. Лемпел был приглашен фирмой Sperry для оказания им помощи в разработке способа наиболее эффективной упаковки данных на компьютерных лентах. В этой же фирме Терри Велч (Terry Welch) расширил алгоритм LZ78, создав новый вариант, широко известный, как LZW.

На работу Велча обратила внимание группа программистов Unix и использовала его алгоритм в их приложении LZW, получившем вполне естественное название compress. Они добавили несколько усовершенствований и опубликовали общедоступную версию этой программы в телеконференции Internet, благодаря чему многие пользователи смогли начать с ней работать.
Популярность алгоритма LZW в значительной степени связана с успехом программы compress. Исходный текст последней версии программы, осуществляющей как сжатие, так и декомпрессию, занимает всего 1200 строк. Ядро кода сжатия занимает не более сотни строк, а код декомпрессии не намного больше. Программисты считают, что это облегчает чтение и понимание алгоритма, а также позволяет адаптировать его для самых разных целей.
Алгоритмы LZ-стиля (включая LZW, LZ77, LZ78 и многие другие варианты) очень популярны везде, где требуется универсальное сжатие. LZW используется в стандарте модема V.42bis, протоколе передачи данных ZModem, форматах GIF, TIFF, ARC и других прикладных программах. Другие алгоритмы LZ используются в дисковых утилитах сжатия типа DoubleSpace и Stacker, графических форматах типа PNG, а также в универсальных утилитах архивирования и сжатия, включая ZIP, GZIP и LHA.
Помимо пользующихся большим вниманием алгоритмов, базирующихся на словаре, существуют и другие подходы. Алгоритм сжатия Хаффмана (Huffman), основанный на статистических колебаниях распределения некоторых значений байтов, лег в основу нескольких очень эффективных методов сжатия, известных, как арифметическое кодирование (arithmetic encoding), энтропийное кодирование (entropy coding) или Q-кодирование (Q-coding). Арифметическое кодирование улучшает сжатие Хаффмана двумя путями. Первое усовершенствование заключается в том, что оно не требует, чтобы выбранные коды были целым числом бит. В то время как сжатие Хаффмана могло выбирать двух- и четырехбитовые коды, программа арифметического кодирования может использовать код длиной 6,23 бит. (Что такое 0,23 бит - чисто философский вопрос, если Вас это заинтересовало, то в отдельном разделе Вы найдете другое объяснение арифметического кодирования.) Второе усовершенствование (которое может также использоваться в сжатии Хаффмана) заключается в том, что арифметическое кодирование использует более сложную статистику. Она не просто следит за частотой появления байта в файле, а оценивает частоту его появления в определенном контексте. Например, при использовании исходного алгоритма сжатия Хаффмана символ "u", встречающийся не слишком часто, мог бы получать довольно длинный код. Но в сложной программе арифметического кодирования символ "u", следующий за "q", будет закодирован очень компактно, так как высока вероятность того, что "u" следует сразу за "q". Комбинация этих двух усовершенствований приводит очень к эффективному сжатию.

 Другие методы сжатия предназначены для данных определенного типа, а потому они плохо подходят для архивирования. Многие усовершенствованные методы, появлявшиеся в последнее время, основывались на синтезе этих трех методов (например, использование кодов Хаффмана для записей словаря) или выполнения сложной предварительной обработки данных, увеличивающей эффективность сжатия одним из этих методов.
Возможно, одним из наиболее существенных событий за последние несколько десятилетий в области алгоритмов сжатия стало появление патентов на программное обеспечение. С 1981 United States Patent and Trademark Office (USPTO) начал принимать заявки на патентование алгоритмов программного обеспечения. Многие из представленных патентов были по методам сжатия. Наиболее известные из них - патенты фирмы Unisys на алгоритм сжатия LZW и патенты фирмы IBM на арифметическое кодирование. К сожалению, первоначально работа по обработке заявок в USPTO была поставлена неважно. В результате чего разным людям предоставлялись различные патенты на один и тот же алгоритм (причем иногда с почти идентичной формулировкой). Некоторые из этих патентов оспаривались в судебном порядке, но высокая стоимость судебного разбирательства исков резко снижает количество таких претендентов.
Один положительный результат введения патентования вряд ли приходится оспаривать. Патентование программного обеспечения спровоцировало появление огромного количества работ по разработке новых алгоритмов сжатия (большая часть которых быстро патентуется их изобретателями). Однако другой эффект был абсолютно отрицательный. Многие из алгоритмов сжатия использовались специфическим образом, например, как часть международных стандартов (V.42bis и JPEG). Кроме того, отдельные компании и пользователи скопировали общедоступный код (так, реализация compress LZW широко копировалась для самых разных целей). Финансовые штрафы за использование этих алгоритмов (в форме авторских отчислений к владельцам патента) отвращали от поддержки этих стандартов авторов условно-бесплатного и бесплатного программного обеспечения или бесплатных библиотек. Некоторые компании публично объявили о том, что они не будут требовать авторских отчислений за использование их запатентованных алгоритмов в бесплатном программном обеспечении. Однако так поступили далеко не все. Пока неясно, как этот конфликт отразится на индустрии бесплатного программного обеспечения и на патентном законодательстве. По крайней мере, одна организация, League for Programming Freedom, борется с патентами программного обеспечения и предпринимает активные шаги по их отмене.

В данной работе будут рассмотрены только современные способы архивации данных, а именно специализированные программы архиваторы. Устаревшие программы резервного копирования типа BACKUP-RESTORE рассматриватся не будут, ввиду их отсутствия в новых версиях MS DOS, начиная MS DOS 6.22.

# Архиваторы MS DOS.

С развитием компьютера стали увеличиваться и объемы информации хранимой в нем, что в свою очередь привело к развитию технологий по хранению этой информации в сжатом виде, то есть в архивах. Для этого было придумано множество программ осуществляющих архивацию информации.

Однако в работе с этой информацией иногда нежелательно раскрывать полный архив, чтобы взять один или два требуемых файла или же просто посмотреть, что в архиве за информация.

Программы-архиваторы, за исключением единиц, не предоставляют удобных оболочек позволяющих просто, быстро и в наглядной форме разобраться с содержимым архивов.

Архиваторы -- это программы, позволяющие создавать и обрабатывать архивные копии файлов. При этом из архивные копии имеют меньший размер, чем оригиналы. С помощью специальных алгоритмов сжатия из файлов удаляется вся избыточная информация, а при применения обратных алгоритмов распаковки архивная копия восстанавливается в первоначальном виде.

Наиболее известные программы-архиваторы для MS-DOS: ARJ (разработчик -- Robert K. Jung), pkzip

(компания PKWARE Inc.), lha (Haruyasu Yoshizaki), zoo (Rahul Dhesi). Безусловным лидером во всем мире за последние 5 лет стал архиватор RAR. В настоящее время RAR активно вытесняет ZIP как основную утилиту сжатия FTP архивов в сети INTERNET. RAR я является единственной всемирно используемой программой, созданной русским программистом (за исключением TETRIS). Все архиваторы отличаются используемыми алгоритмами сжатия, форматами архивных файлов, скоростью работы и т.д.

Термины, используемые в архивации

Add file Добавление (копировние) файла в архив. Если архив не существует, то он создается.

CRC Код циклического контроля. Специальная функция от всего содержимого файла. Составляется таким образом, что изменить файл так, чтобы его CRC остался неизменным, практически невозможно.

Exclude selected files При архивации НЕ добавлять в архив определенные файлы.

Extract files Извлечение файлов из архива без сохранения структуры подкаталогов.

Extract files with pathnames Извлечение файлов из архива с сохранением структуры подкаталогов.

Fresh files Добавление в архив новых версий уже имеющихся там файлов.

Garble (или scramble) files with password Архивация файлов с паролем. Извлечь файлы из такого архива можно, лишь правильно указав пароль.

Move files Перемещение файлов в архив.

Multiple volumes Многотомные архивы -- состоящие из нескольких файлов (томов). Удобны при архивации больших комплексов файлов, когда тома архива можно помещать на отдельные дискеты.

Ratio Степень сжатия файла. Определяется по-разному в различных архиваторах (отношение исходного к сжатому либо наоборот).

Recurse subdirectories Архивация файлов из заданного каталога и всех его подкаталогов. В архиве сохраняется информация о пути к файлам, и при извлечении их можно выводить не в один каталог, а в соответствующие подкаталоги.

Self-extract (sfx)archive Самораспаковывающийся архив. Архивный файл имеет расширение .exe .или com, и после его запуска происходит автоматическое извлечение файлов из архива.

Test integrity Проверка целостности архива, т.е. проверка CRC файлов архива.

Update files Добавление в архив новых файлов.

Архиватор ARJ

Работает из командной строки. Выполняет все функции по обслуживанию архивов .arj , в т.ч. поддержку многотомных архивов.

Получить справку по ключам архиватора arj с помощью команд:
arj(обычная справка)

arj /?(подробная справка)

**Arj** имеет очень большое число ключей. Можно автоматизировать много действий -- создание резервной копии диска, архивирование начиная с какой-то даты, добавление к имени архива текущей даты (arh970821.arj), архивирование файла с конкретного места, несколько уровней сжатия и так далее. В версии 2.55 возможна работа с длинными именами.
*Достоинства:* очень большое колличество ключей, что дает возможность автоматизировать большое число функций. Защита архива от повреждений.
*Недостатки:* отсутствие диалогового режима, некоторое неудобства работы при наличии какого-то ключа в переменной окружения (ARJ\_SW) и строке запуска -- взаимное уничтожение.

## PKZIP

Работает из командной строки. Различные функции по обслуживанию архивов .zip выполняются разными программами:
pkzip - помещение файлов в архив

pkunzip - извлечение файлов из архива

zip2exe - создание самораспаковывающегося архива

pkzipfix - восстановление поврежденного архива.

Изучить справку по работе с архиватором pkzip с помощью команд:

pkzip /h

pkunzip /h

zip2exe /h

## RAR

Архиватор RAR v2.50 для DOS - Интегрированная программа управления архивами

RAR - это очень мощное средство для создания архивов и управления ими. Возможности RAR:

Полноэкранный интерактивный интерфейс (отключаемый);

Поддержка мыши и меню;

Поддержка не-RAR архивов;

'Стандартный' интерфейс командной строки;

Оригинальный высокоэффективный алгоритм сжатия данных;

Специальный алгоритм для сжатия мультимедийных файлов;

Лучшая степень упаковки, чем у аналогичных продуктов, за счет использования режима "непрерывного" сжатия;

Информация об авторе архива (только в зарегистрированной версии);

Самораспаковывающиеся (SFX) обычные и многотомные архивы;

Восстановление физически поврежденных архивов;

Язык программирования для инсталляционных SFX-архивов;

Блокировка, шифрование, список порядка файлов, метки томов и др.

## QUARK

Quark является архиватором классического типа, использующим LZ77-алгоритм для уплотнения исходных данных путем кодирования повторяющихся последовательностей байт (RSE-алгоритм) с последующим вторичным уплотнением сжатого потока кодами Хаффмана. Подобные методы используют все три лидера в области упаковки данных - архиваторы ARJ, LHA, PkZIP.
Тем не менее, Quark добивается лучших результатов в компактности данных при скорости лучшей чем LHA, не меньшей чем у ARJ и не сильно отличающейся от скорости PkZIP, при использовании им т.н. максимальной компрессии данных. Это обусловлено несколькими причинами:
1) Quark работает с плавающим размером окна от 32Kb до 64Kb (против фиксированных 16Kb у LHA, и 32Kb у PkZIP и ARJ).
2) Quark выполняет оптимизацию Первого рода (оптимальность адресов ссылок LZ77) и оптимизацию Второго рода (оптимальность ссылочного покрытия потока).
3) Quark использует текстовую редукцию для текстовых файлов.
4) Quark заносит в архив минимум служебной информации, не претендуя на иные аппаратные платформы и операционные системы.

## GZIP

Gzip сокращает размер заданных файлов используя кодирование Зива-Лемеля (LZ77). Когда возможно, каждый файл замещается файлом с расширением '.gz', при этом сохраняются владелец, режимы, доступ и времена модификации (Другие расширения '-gz' для VMS, 'z' для MSDOS, OS/2, FAT и Atari). Если никаких файлов не указано или имя файла '-', то пакуется стандартный ввод и выдается на стандартный вывод. Gzip пытается паковать только обычные файлы, в частности GZip игнорирует символические ссылки.

 Gzip использует алгоритм Зива-Лемеля также как Zip, PKZIP. Итоговый размер, полученного файла после сжатия, зависит от размера исходного файла и наличия в нем общих подстрок. Обычно, такой текст, как исходный код или английский текст сокращается на 60-70%. Паковка с использованием этого алгоритма обычно лучше, чем при использовании LZW (его использует Compress), кодирование Хаффмана (его использует Pack) или адаптированное кодирование Хаффмана (Compact).

Упаковка происходит независимо от того увеличился ли размер упакованного файл в сравнении с оригиналом или нет. Причина расширения - несколько байтов для заголовка Gzip файла, плюс 5 байтов для каждого 32К блока, или отношение расширения 0.015% от длины файла. Заметим, что фактическое число занятых на диске блоков уже никогда не возрастает. Gzip сохраняет режимы доступа, владельцев и время модификации файлов при упаковке и распаковке...

## ARJZ

ARJZ (по воле автора программы произносится как "арж-зет") - это архиватор, основанный на известной программе ARJ Роберта Юнга. В отличие от таких современных средств архивирования, как RAR и UC2, ARJZ использует формат файлов, командную строку и опции, совместимые с одной из самых популярных программ сжатия данных, а это имеет свои преимущества. В частности:
1) Практически всё программное обеспечение, рассчитанное на вызов ARJ, будет работать так же и с программой ARJZ без всякой модификации. Например, не надо будет переписывать ни ARCVIEW, ни NC 4.0, ни DN, ни тех .BAT файлов, которые вы могли создать за время пользования ARJ'ем.

2) Для того, чтобы использовать возможности ARJZ'а при работе с вашими старыми архивами, вам совсем не нужно переархивировать их заново.

3) Вы так же почти избавляетесь от необходимости изучать новый архиватор. Зная, как запускается ARJ, вы знаете, как запускается ARJZ.

Однако, следует иметь в виду, что:

1) ARJZ позволяет сжимать файлы, используя более мощные методы, чем оригинальная программа. В этом случае ARJ НЕ СМОЖЕТ ПРОВОДИТЬ ОБРАБОТКУ ПОЛУЧЕННЫХ АРХИВОВ, СВЯЗАННУЮ С РАСПАКОВКОЙ, т.е. деархивирование, тестирование и т.д. В любом случае вы сохраните возможность обновлять и сливать архивы, переименовывать или удалять файлы в них, а так же получать список файлов в архивах.

2) ARJZ/UNARJZ с одной стороны, поддерживают не все команды и опции ARJ'а, а с другой - вводят новые и это может создавать проблемы при работе. В действительности такие проблемы встречаются чрезвычайно редко и легко разрешимы.

Достоинства и недостатки

К достоинствам ARJZ можно отнести:

 1) Версии под DOS (реальный/расширенный режимы), OS/2 и NT. В программу для расширенного режима DOS встроен расширитель, поэтому она работает на компьютерах 386+ без какого-либо дополнительного программного обеспечения.

 2) Высокую скорость сжатия: ARJZ сжимает файлы с тем же качеством, что и ARJ примерно в полтора раза быстрее последнего (кроме версии, работающей в реальном режиме).

 3) Высокую степень сжатия (в этом случае полученные архивы не будут распаковываться ARJ'ем). По этому параметру ARJZ находится на уровне RAR/UC2 (в этом вы можете убедится сами - you see too ;-).

4) Так называемый "полуэкранный интерфейс". ARJZ может во время работы выводить на экран окошко с двумя индикаторами процесса, именем архива и именем пакуемого файла - это замечательная особенность предназначена специально для таких программ, как ARC- или ARJVIEW, SHEZ, ARJMENU, NC 4.0+, DN и др.

5) Здесь, конечно не место для описания преимуществ UNARJZ'а, но тем не менее... Высокая скорость распаковки. Даже на XT UNARJZ работает в среднем в 1.5-2 раза быстрее, чем ARJ, а при использовании специальной опции (см. UNARJZ.DOC) разница возрастает ещё в два раза.

Важно отметить, что процедуры деархиватора оптимизированы отдельно под процессоры 286, 386, 486 и Pentium.

ARJZ написан таким образом, что его можно использовать и как отдельный архиватор и как надстройку над ARJ'ем: если он не может распознать команд или опций командной строки, то запускает оригинальную программу. Это, фактически, означает, что, используя ARJZ, вы, тем не менее, не теряете ни одной опции ARJ'а.

Недостатки ARJZ:

1) В ARJZ (по крайней мере, пока) нет поддержки многотомных (multi volume), резервных (backup) и самораспакующихся (SFX) архивов. Заметьте, что UNARJZ распаковывает любые архивы, созданные ARJ.
2) ARJZ не является полноценным архиватором в том смысле, что он самостоятельно не удаляет и не переименовывает файлы в архивах, не может сливать архивы и т.д. Всю эту работу можно сделать с помощью оригинальной программы, поэтому нельзя говорить, что пара ARJZ/UNARJZ полностью заменяет собой ARJ.

**Далее будут кратко рассмотрены менее популярные архиваторы, которые, однако все еще используются в среде MS DOS.**

**Arhangel** -- этот архиватор базируется на архиваторе HA, но несовместим с ним. По заявлению создателей этот архиватор наиболее подходит для сжатия изображений (TIFF, BMP).
*Достоинства*: хорошее сжатие графической информации.
*Недостатки*: обилие ключей для установки максимального сжатия -- просто теряешся какой выбрать.

**JAR** -- новый архиватор из семейства ARJ. У меня он показал более плохие результаты, нежели WinRAR, хотя в документации сказано, что "это лучший архиватор в терминах компрессии". Другие тесты тоже, вроде, подтверждают, что этот архиватор лучше WinRAR и других подобных, но он, безусловно, не занимает первое место. Возможности примерно такие же, как и у ARJ.
*Достоинства:* достаточно быстрая работа, поддержка recovery record -- возможность восстановления сбойных архивов.
*Недостатки:* нет диалогового режима работы.

**HA** -- Архиватор, написанный Harry Hirvola в 1993 году. Работает медленно (или даже очень медленно), но, по результатам тестов, хорошо. В данном тесте он не на первом месте по степени сжатия. Насколько я знаю, он особенно хорошо сжимает текстовые файлы, была проведена отдельная проверка, которая подтвердила данный факт -- несмотря на солидный возраст этот архиватор неплохо сжимает: на 20% меньше, чем PKZIP. Таким образом для сжатия больших текстовых файлов лучше пользоваться HA (LGHA) чем WinRAR. Появился архиватор LGHA, который создает точно такие же архивы, но немного быстрее.
*Недостатки:* Малое количество ключей, несколько нестандартные комантды. Невозможность создания архивов на несколько дискет и самораспаковывающиеся архивов. Не поддерживает шифрование.

**UC II (Ultra compress)** -- программа, работающая из командной строки, но имеющая диалоговое меню для настройки различных параметров. Есть возможность добавлять recovery record. Интересная дополнительная программа по шифрованию архивов.
*Недостатки:* Разбивка на дискеты только фиксированной длинны и только внешней программой...

**ACB** -- программа российского программиста обеспечивающая хорошее, даже ОЧЕНЬ хорошее сжатие, НО требующая очень хорошего компьютера. По словам разработчика необходим как минимум 486 комп с 15 мегабайтами ОЗУ. Но реально требуется гораздо больше. Почему-то очень медленная распаковка -- 27 мегабайт информации программа распаковывала 3000 секунд (т.е. 50 минут) и это при том, что программа запускалась на Pentium-200 с 32 мегабайтами ОЗУ.
*Достинства:* маленький размер программы, возможность защиты архивов от повреждения.
*Недостатки:* Очень завышенные требования к машине. Не хочет архивировать большие файлы и ставить защиту от повреждения на большие архивы. Еще один формат команд -- нестандартые команды.

**Rkive** -- этот архиватор по результатам сжатия занял третье место в данном тестировании (в сходном тесте он пропустил вперед BOA и ACB). Тесты шли с бета версией.
*Достоинства:* не смотря на небольшой размер архиватора (58 к) очень приличное сжатие.
*Недостатки:* неустойчивая работа -- на одной из машин все время аварийное завершение работы архиватора (без сбоя системы в целом -- то ли машина не нравится, то ли 16 мегабайт памяти мало...). Нехватка памяти при был замечен еще один недостаток -- невозможность прибавления файлов к существующим архивам. Отсутствует возможность создания SFX архивов.

**BOA** -- этот архиватор по результатам сжатия занял второе место среди рассматриваемых архиваторов тестировании (в сходном тесте он занял первое место). Но переходить на этот архиватор, безусловно, рано, т.к. это только предварительная версия для тестирования.
*Достоинства:* достаточно быстро работает.
*Недостатки:* будем считать, что недостатки присутствуют из-за бета-версии...

**BIX** -- новый архиватор, который подает хорошие надежды. Пока только бета версия. Создан Игорем Павловым. Автор предупреждает, что **если вы будете использовать маску "\*.\***", то у вас не будут добавляться файлы без расширения! Для добавления всех файлов следует использовать маску "\*"

**DST** -построен таким образом, что сжимает только один файл. Выход из этого -- воспользоваться утилитой, которая может сжимать сздавать один файл из нескольких.

# Сравнение вресий архиваторов MS DOS и Windows.

Несмотря на кажущуюся «моральную отсталость» MS DOS, в ряде случаев использование MSDOS-архиваторов может быть гораздо более эффективным, по сравнению с аналогичными версиями для Windows 9x.

Архиваторы работающщие в среде MSDOS работают в реальном режиме процессора. Это обеспечивает не менее чем в 1.5 раза большую производительность процессора, по сравнению с защищенным режимом Windows 9x (для процессоров P-II и старше).

Все версии архиваторов для MS DOS ( по крайней мере из рассмотренных выше), весьма эффективно используют память XMS или EMS. С учетом реального режима работы процессора, это дает еще 10-15 % выиграша в производительности.

Как ни странно, версии архиваторов для MS DOS часто используют более совершенные алгоритмы сжатия. Причина проста - в большинстве случаев в версиях архиваторов для MS DOS опробуются возможные усовершенствования, переносимые далее в Windows. Причем, иногда применение некоторых новшеств в Windows оказывается технически нецелесообразным. Пример - отказ от применения в архиваторе WinZip 8.0 Beta словаря пременной длинны более 128 кБайт, из-за переполнения кеш-памяти современных процессоров Pentium-II или Celeron.

Резюмируя, можно утверждать, что при прочих равных условиях (тип вычислительной системы, степень сжатия и др.) производительность архивации (особенно упаковки с высокой степенью сжатия) в архиваторах для MS DOS примерно в 1.5-1.6 выше. Грубо говоря, в 1.5 раза быстрее. Для небольших архивов (до 40-100MB) это не является существенным. Однако, если выполнять архивирование 10000-15000 файлов общим объемом более 1-2Гб (типичный объем небольшого FTP сервера INTERNET), разница во времени может составить до 30-60 минут (Celeron-450MHz + 64MB RAM).

И наконец, в случае повреждения загрузочных файлов Windows 9x , единственным способом «спасти» архив, является использование проверенных и надежных архиваторов для MS DOS.

# Литература.

Фигурнов В.Э. «IBM PC для пользователя. Краткий курс.» - М.: ИНФРА-М, 1998. - 480 с.: ил

 PCMagazine, 1997-1999 гг.