NAME

pbzip2 − parallel bzip2 file compressor, v1.1.10

SYNOPSIS

pbzip2 [**−123456789**] [**−b#cdfhklm#p#qrS#tvVz**] [*filenames ...*]

DESCRIPTION

pbzip2 is a parallel implementation of the bzip2 block-sorting file compressor that uses pthreads and achieves near-linear speedup on SMP machines. The output of this version is fully compatible with bzip2 v1.0.2 or newer (ie: anything compressed with *pbzip2* can be decompressed with bzip2).

pbzip2 should work on any system that has a pthreads compatible C++ compiler (such as gcc). It has been tested on: Linux, Windows (cygwin), Solaris, Tru64/OSF1, HP-UX, and Irix.

The default settings for *pbzip2* will work well in most cases. The only switch you will likely need to use is -d to decompress files and -p to set the # of processors for *pbzip2* to use if autodetect is not supported on your system, or you want to use a specific # of CPUs.

OPTIONS

−b# Where # is block size in 100k steps (default 9 = 900k)

−c, −−stdout
Output to standard out (stdout)

−d,−−decompress
Decompress file

−f,−−force
Force, overwrite existing output file

−h,−−help
Print this help message

−k,−−keep
Keep input file, do not delete

−l,−−loadavg
Load average determines max number processors to use

−m# Where # is max memory usage in 1MB steps (default 100 = 100MB)

−p# Where # is the number of processors (default: autodetect)

−q,−−quiet
Quiet mode (default)

−r,−−read
Read entire input file into RAM and split between processors

−S# Child thread stack size in 1KB steps (default stack size if unspecified)

−t,−−test
Test compressed file integrity

−v,−−verbose
Verbose mode

−V Display version info for *pbzip2* then exit

−z,−−compress
Compress file (default)

−1,−−fast ... −9,−−best
Set BWT block size to 100k .. 900k (default 900k).

−−ignore-trailing-garbage=#
Ignore trailing garbage flag (1 - ignored; 0 - forbidden)

If no file names are given, pbzip2 compresses or decompresses from standard input to standard output.

FILE SIZES

You should be able to compress files larger than 4GB with *pbzip2*.

Files that are compressed with *pbzip2* are broken up into pieces and each individual piece is compressed. This is how *pbzip2* runs faster on multiple CPUs since the pieces can be compressed simultaneously. The final .bz2 file may be slightly larger than if it was compressed with the regular bzip2 program due to this file splitting (usually less than 0.2% larger). Files that are compressed with *pbzip2* will also gain considerable speedup when decompressed using *pbzip2*.

Files that were compressed using bzip2 will not see speedup since bzip2 packages the data into a single chunk that cannot be split between processors.

EXAMPLES

Example 1: pbzip2 myfile.tar

This example will compress the file "myfile.tar" into the compressed file "myfile.tar.bz2". It will use the autodetected # of processors (or 2 processors if autodetect not supported) with the default file block size of 900k and default BWT block size of 900k.

Example 2: pbzip2 -b15k myfile.tar

This example will compress the file "myfile.tar" into the compressed file "myfile.tar.bz2". It will use the autodetected # of processors (or 2 processors if autodetect not supported) with a file block size of 1500k and a BWT block size of 900k. The file "myfile.tar" will not be deleted after compression is finished.

Example 3: pbzip2 -p4 -r -5 myfile.tar second*.txt

This example will compress the file "myfile.tar" into the compressed file "myfile.tar.bz2". It will use 4 processors with a BWT block size of 500k. The file block size will be the size of "myfile.tar" divided by 4 (# of processors) so that the data will be split evenly among each processor. This requires you have enough RAM for pbzip2 to read the entire file into memory for compression. Pbzip2 will then use the same options to compress all other files that match the wildcard "second*.txt" in that directory.

Example 4: tar cf myfile.tar.bz2 --use-compress-prog=pbzip2 dir_to_compress/
Example 4: tar -c directory_to_compress/ | pbzip2 -c > myfile.tar.bz2

These examples will compress the data being given to pbzip2 via pipe from TAR into the compressed file "myfile.tar.bz2". It will use the autodetected # of processors (or 2 processors if autodetect not supported) with the default file block size of 900k and default BWT block size of 900k. TAR is collecting all of the files from the "directory_to_compress/" directory and passing the data to pbzip2 as it works.

Example 5: pbzip2 -d -m500 myfile.tar.bz2

This example will decompress the file "myfile.tar.bz2" into the decompressed file "myfile.tar". It will use the autodetected # of processors (or 2 processors if autodetect not supported). It will use a maximum of 500MB of memory for decompression. The switches -b, -r, and -1..-9 are not valid for decompression.

Example 6: pbzip2 -dc myfile.tar.bz2 | tar x

This example will decompress and untar the file "myfile.tar.bz2" piping the output of the decompressing pbzip2 to tar.

Example 7: pbzip2 -c < myfile.txt > myfile.txt.bz2

This example will read myfile.txt from standard input compressing it to standard output which is redirected to to myfile.txt.bz2.

SEE ALSO

bzip2(1) gzip(1) lzip(1) rzip(1) zip(1)

AUTHOR

Jeff Gilchrist

http://compression.ca