



CATERPILLAR

A COMPRESSED_SENTENCE DUMPER, REVISION 13++

Caterpillar_HDD2RAM.bat:

```
@echo off
if exist Caterpillar.hits.Pattern1.html del Caterpillar.hits.Pattern1.html>nul
if exist Caterpillar.hits.Pattern2.html del Caterpillar.hits.Pattern2.html>nul
if exist Caterpillar.hits.Pattern3.html del Caterpillar.hits.Pattern3.html>nul
if exist Caterpillar.hits.Pattern4.html del Caterpillar.hits.Pattern4.html>nul
dir *.Okumura/b/oe>Caterpillar.lst
echo Note1: Due to heavy recursion within search function Pattern1: *underdog* is
echo      much-much slower than Pattern1: * and Pattern1_NestedPattern1: underdog
echo Note2: Since r.13++ Caterpillar.exe rejects lines longer than 960 chars.
echo.
Caterpillar Caterpillar.lst Caterpillar_HDD2RAM.ini
echo Press a key to exit.
pause>nul
```

**Caterpillar(Sentence_Dumper), revision 13++, written by Svalqyatchx,
in fact adapted from Haruhiko Okumura's excellent LZSS.C program.**

How near are these words_forms to me: Masakari, Massacre, Steel-Coloss,
Monster-Truck, Dump-Mining-Truck, Caterpillar 797, Liebherr, Komatsu. They
resemble one thing: strong-devoid-of-ambition-power(i.e. a pure work/time).

'Caterpillar' is a simple pattern searcher(from 'Masakari' family tools)
into archived english-text files, designed to achieve up to 90% higher read
speed than the HDD READ BURST i.e. 'copy hugefile nul' gaining at same time
50% compression of searched data.

Its main feature is somewhat hidden nowadays, because of
pseudo-transparent decompression used, which leads to doubling(unreachable in
fact) uploaded data for search function(written by N. Horspool, thanks a lot)
due to LZSS algorithm implemented by H. Okumura(greetings to him). Okumura's
variant(HDD2RAM) which is much faster(!!!) and needs less memory than tuned
memory-to-memory decompression(RAM2RAM) variant. I am still stunned.

In few words: feeding search function is 100-% faster with very fast
CPU-Physical_RAM subsystems, in this way reducing the ugly penalty which comes
from reading a HDD. In numbers: me IDE HITACHI 7200rpm 2MB gives up to 60MB/s
READ BURST, 'Caterpillar' almost doubles(i.e. 120-MB/s) it in case of 3+++GHz
CPU and 533+++MHz RAM.

For Windows 2003, VIA KT600, AMD XP 2500+(1836.12MHz=11x166.92MHz),
FSB 333.84MHz(2x166.92MHz), 512KB L2 cache, 1 DIMM DDR 512MB 333MHz(2x166MHz),
Caterpillar(in fact LZSS) decompresses 58,000KB per second i.e.
boost is negative: 60MB/s=61,440KB/s(READ BURST) is greater than 58,000KB/s.
But for two times faster CPU-RAM sub-system(SERVER) than described above OR
for two times slower HDD sub-system(LAPTOP) boost will be positive:
(1 - READ BURST SPEED / DECOMPRESSION SPEED) * READ BURST SPEED or
(1 - (61,440KB/s) / (2 * 58,000KB/s)) * 61,440KB/s = (0.471) * 61,440KB/s.

Since revision 5 'fread()' was changed with 'read()', for speed.

'The Monster-Dump-Truck' short notes:

Note1: Thanks a lot to N. Horspool, Dmitry Shkarin, H. Okumura, Igor Pavlov.

Note2: Run it without parameters to get usage and short notes.

Note3: Current revision searches only for case-sensitive and unexact matches.

Note4: This simple amateurish(more over I am not versed well neither in C nor
in mathematics nor in english language, but I am persistent in INDEXING
GBs of english TEXTS) tool is written in ANSI C(at least its source is
compileable for CL(Windows) and not yet for GCC(Linux) because of

'O_BINARY in open(), gets(), getch(), kbhit()', and its purpose is to create a SentenceList for a group of compressed(with it) text files(LF and CRLF) given via filelist.

Its name comes from a heavy-nopride-dumper-truck 'Caterpillar'.

Note5: By default allocated memory is 95MB i.e. decoding is HDD2RAM.

Note6: Disastrous performance in case 95MB|147MB not fully physical!

Note7: For me digital library:

where files are 54, ENcoded 6,917,425,566, DEcoded 14,419,485,826 with Windows XP, VIA KT600, AMD XP 2500+(1836.12MHz=11x166.92MHz), FSB 333.84MHz(2x166.92MHz), 512KB L2 cache, DDR 512MB 333MHz(2x166MHz), IDE HDD Maxtor 80GB 7200 8MB and

'D:\temp>dir E:\KAZEHOME\KAZUYA.O??\b>Caterpillar.lst'

'D:\temp>Caterpillar Caterpillar.lst CaterpillarRAM2RAM.ini'

result is: 282 seconds or 41000KB/s upload, 11000KB/s boost, 52000KB/s boosted upload, 56000KB/s decode.

'D:\temp>Caterpillar Caterpillar.lst CaterpillarHDD2RAM.ini'

result is: 142 seconds or 99000KB/s boosted upload!!!

Note8: Matches(hits) containing neither '<' nor '>' are written to 'Caterpillar.hits.pattern?.html' file.

Note9: Works both on UNIX(LF) and Windows(CRLF) text files.

NoteA: Never forget the importance of defragmented_AND_grouped files located at fastest area of disk - first partition is faster than second one, etc.

NoteB: In ANSI, clock is defined as '#define CLOCKS_PER_SEC 1000'.

NoteC: Since Caterpillar 13++:

- limits(just skip longer ones) lines to 960 chars; OTHERWISE: HUGE TIME DELAYS due to recursive function;
- shows hits to console too; MORE VIVID;

Noted: During execution hitting a 'Esc' causes termination(i.e. skipping rest).

NoteE: At last NON-ENCODED regime has two modes: in addition to LINE(i.e. hits are lines) there is a FILE(i.e. hits are filenames) mode.

NoteF: For all regimes files Caterpillar.HIT?.lst are created for each pattern(1,2,3 and 4) - containing hits filelist i.e. filenames containing HITS(either LINES or FILENAMES).

Below is LINE(default for DECODING ???2RAM regimes) mode pattern description:

Pattern(s) note: You may specify(four times) a main-pattern(case insensitive with wildcards '*' i.e. any character(s) or empty and '?' i.e. any character or empty) with three nested-patterns(case sensitive and unexact), all four connected with AND. Due to different line endings(CRLF in Windows; LF in UNIX) you must add a '?' wildcard in place of CR: for example in case of searching for '*.pdf' write '*.pdf?'.

Pattern(s) example: Pattern1: *take? *it*

Pattern1_NestedPattern1: you

Possible hit: ... your reason is so taken by It.

Usage: 'Caterpillar e file1 file2' encodes file1 into file2

'Caterpillar d file2 file1' decodes file2 into file1

'Caterpillar m ListOfFilesFile SolidSize'

<ListOfFilesFile>: Files to be merged into Caterpillar.??? files

<SolidSize>: Caterpillar.??? files size limit in MB.

'Caterpillar ListOfFilesFile [OptionsFile]'

<ListOfFilesFile>: Input file with files for Caterpillaring

<OptionsFile>: Optional input file with options with following format:

Optional line #1 contains method of decoding:

'DECODING HDD2RAM' | 'DECODING RAM2RAM' | 'NON-ENCODED'

'NON-ENCODED' allocates 95MB, size of biggest file must be lower;

'DECODING HDD2RAM' needs less physical memory(95MB) but is faster!

'DECODING RAM2RAM' needs more physical memory(147MB) but is slower!

Optional line #2 contains terminal hits:
 '0' | 'long integer'
 '0' means all hits are needed
 'long integer' means reaching this value termination follows
Optional line #3 contains Pattern1: 'string'
 if 'string' is specified then input from keyboard arise not
 if 'string' is not specified then input from keyboard arise
Optional line #4 contains Pattern1_NestedPattern1: 'string'
Optional line #5 contains Pattern1_NestedPattern2: 'string'
Optional line #5 contains Pattern1_NestedPattern3: 'string'

Note1: One useful way to make 'ListOfFilesFile=Caterpillar_NON.lst' is next:
D:\Caterpillar>copy con MAKElst.bat
@echo off
dir Caterpillar_tree*.lbl /s/b>Caterpillar_NON.lst
dir Caterpillar_tree*.txt /s/b>>Caterpillar_NON.lst
echo.
F6

*Have a nice Caterpillaring.
For contacts: sanmayce@hotmail.com
Sanmayce Svalqyatchx 'Kaze', 2007 Oct 25.*