Entropy Adaptive Coding project requirements

Introduction

Entropy Adaptive Coding (EAC) is an algorithm proposal that can improve effeciency of any compression algorithm that does not respond to entropy change in data that is being compressed. EAC breaks data into blocks of fixed length and compresses each block using some compression algorithm (as an example we will use LZ77 as underlying algorithm). By changing parameters of compression after each block we can achieve better compression ratio when source data entropy changes. For example LZ77 can be adapted by adjusting it's window. When entropy is high (i.e. data is more structured) larger window will give better result. But when entropy is low (i.e. data is not structured, randomized) shorter window can provide better result. Since simple LZ77 can not change it's window during operation, EAC can provide better result by adjusting window size between blocks. This project implements both simple LZ77 compression algorithm and EAC algorithm. The goal is to test EAC algorithm by comparing it's result against simple LZ77 on real world and synthetic (generated with given entropy) files. Generator of files with given entropy is part of this project.

Environment requirements

1. Programming language - C
2. Operating system - Linux distribution
3. Binaries that must be compiled:
    1. Encoder - binary that can be run from command line and is able to compress using EAC and LZ77 (configurable by parameters)
    2. Decoder - binary that can be run from command line and is able to decompress using EAC and LZ77 (configurable by parameters)
    3. Generator - binary that can be run from command line and is able to generate random data of specified size and entropy.
4. Test files set that must include:
    1. Simple text files
    2. Raw video files
    3. MS Office files
    4. Synthetic random data files with known entropy
5. Scripts that should be written:
    1. Sanity test - simple test script that compresses a few files, decompresses them and than compares to original files. This script should be used to check that encoder/decoder work correctly.
    2. Performance test - script that should run a set of compressions/decompressions on each test file and record results in some file (for example CSV or XLS).
6. Additional - result files simple graphical viewer.

Functional requirements

Encoder

1. Two modes of operation switched by command line option:
    1. Simple LZ77 compression
        1. In this mode window size can be adjusted
        2. Block size is ignored in this mode
        3. Implemented in single thread
    2. EAC compression
        1. In this mode block size can be adjusted
        2. Window size is ignored in this mode
        3. For performance reasons encoding using EAC *must* be implemented multithreded.
2. Logging options
    1. By default outputs only information needed for performance test scripts
        1. Source file size
        2. Compressed file size
        3. Compression ratio
        4. Size of longest match and average size of match per file and per block
    2. Additionally verbose logging can be used that shows high level details of operation
    3. Additionally debug logging can be used that outputs result of every function execution

Decoder

1. Two modes of operation switched by command line option:
    1. Simple LZ77 decompression
        1. In this mode window size should be specified
    2. EAC decompression
        1. In this mode block size must be specified
2. Logging options
    1. By default outputs only information needed for performance test scripts
    2. Additionally verbose logging can be used that shows high level details of operation
    3. Additionally debug logging can be used that outputs result of every function execution

Generator

1. Simple program that generates files of given entropy
2. Accepts as parameters
    1. Size of file
    2. One of

1. Probabilities of 0 and 1 bits
2. Probability of bit flip (10 or 01 sequence)

Other requirements

Test files

1. Following types of files must be used to test and compare EAC and LZ77 algorithms:
    1. Raw video file
    2. Simple text file
    3. Fully random data generated by random generator initialized with truly random data
    4. MS office files
    5. JPEG files
2. Files should not be too large to be able to run all tests in a day or two

Documentation

1. C source code
    1. Every function, module and source file must be documented
    2. Choise of documentation system is Doxygen - it allows generating HTML and PDF documentation automatically from source files.
2. Scripts
    1. Every test script and helper script that is used by test script must be documented.
3. Simple results graphical viewer
    1. Must be written in HTML+CSS+JavaScript.
    2. Since it is very simple, the only documentation it will have is some basic annotations in source code in case someone later wants to modify it or understand how it works.

Test runs and results

1. After binaries and scripts are ready all tests must be run on some computer
2. The goal is to have results of all tests along with sources of all programs. This is to both be able later improve the program or adjust it to algorithm changes and be able to check that program does work exactly according to algorithm.

Glossary

1. **EAC** - Entropy Adaptive Coding
2. **EAC block** - Fixed length block of data
3. **LZ77** - Lempel Ziv 77
4. **LZ77 window** - number of bits (or bytes) of previous data where matches are searched.

5. **C language** - Standard C99 language
6. **Multithreaded program** - program that has more than 1 execution thread. Each thread can be executed in parallel to other threads.
7. **PDF** - Portable Document Format
8. **JPEG** - commonly used method of lossy compression for digital photography
9. **HTML** - Hyper Text Markup Language
10. **CSS** - Cascading Style Sheets
11. **JavaScript** - scripting language used primarily in HTML pages