Generating Compressors with LC

Martin Burtscher
Department of Computer Science
TEXAS STATE UNIVERSITY

LC Framework

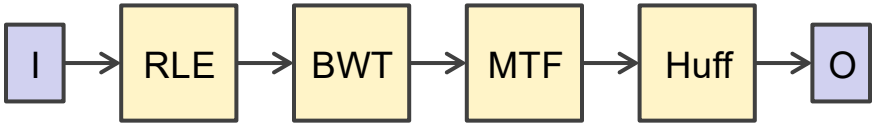
- LC generates data compressors & decompressors
  - Automatically searches for effective compressor
  - Customizes algorithm to given data
  - Targets high speed and good compression ratio
- LC supported features
  - Lossless and guaranteed-error-bounded lossy versions
  - Full CPU and GPU compatibility

LC Successes

- LC helped create several leading compressors
  - FPcompress: lossless floating-point compressor for CPUs and GPUs
  - PFPL: guaranteed-error-bounded lossy floating-point compressor for CPUs and GPUs
  - LICO: lossless image compressor for CPUs
  - SLEEK: ultra-fast lossy and lossless floating-point compressor for GPUs

Background

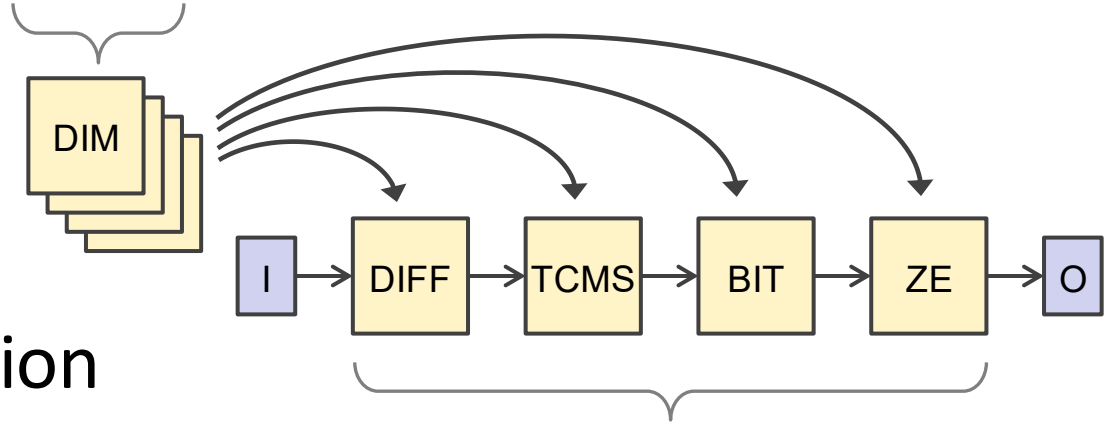
- Compression algorithms
  - Tend to be chains of data transformations
  - Examples
    - gzip = LZ77 + Huffman
    - bzip2 = RLE + BWT + MTF + Huffman + …
  - Also called pipelines



- Operation
  - Input is transformed by each stage to produce output

LC Origin

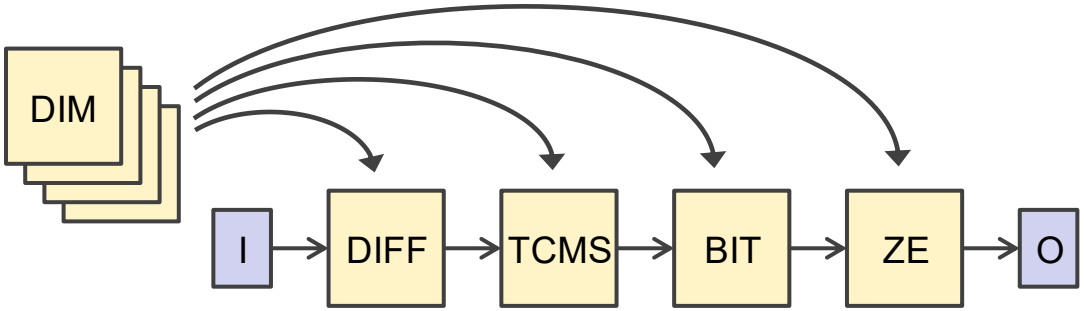
- Idea
  - Extract transformations from various compressors
  - Build library of these transformations (components)



- Operation
  - Generate all possible combinations of *k* components
  - Evaluate on training inputs to find best algorithm

LC Search Space

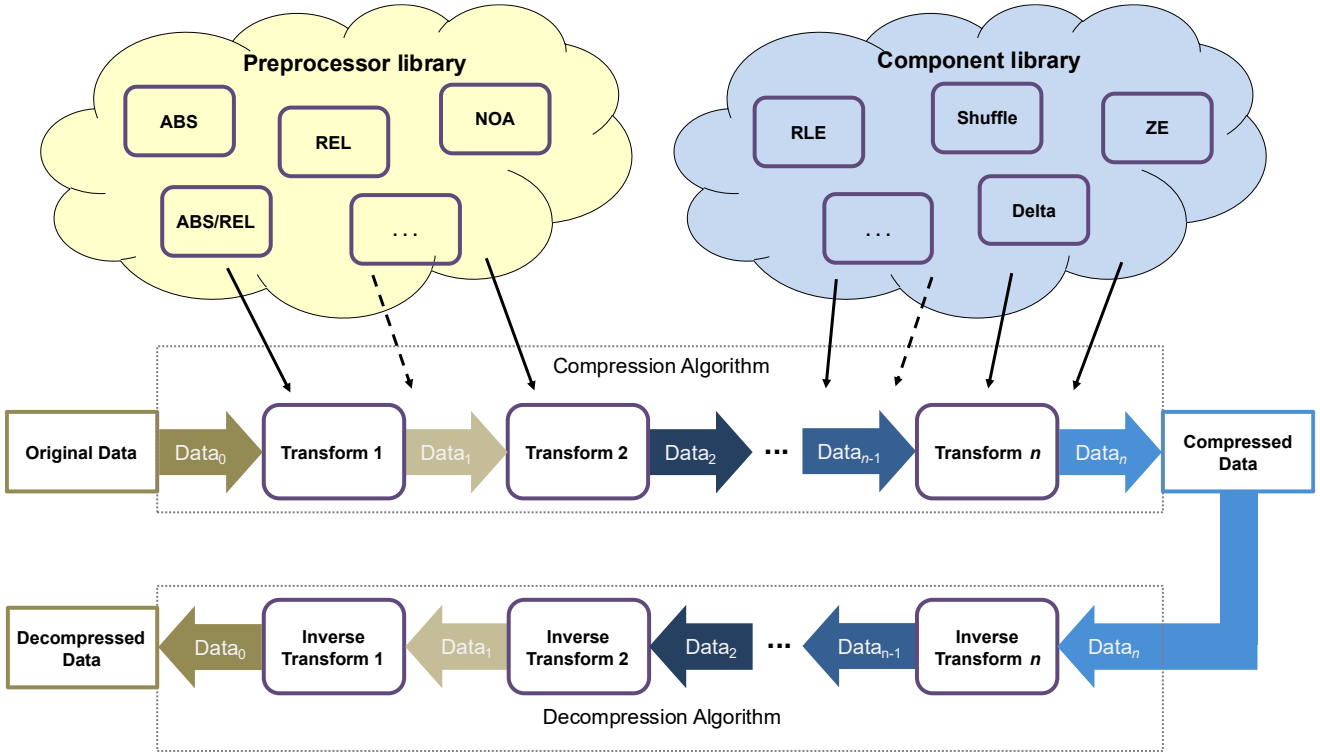
- Can generate huge number of compression algorithms
  - p^k for library with p components and pipelines with k stages
    - Ex: $p = 40$, $k = 4 \rightarrow$ 2,560,000 pipelines in search space
  - Vast space for discovering new compression algorithms



- LC incorporates genetic algorithm
  - ML approach to find good algorithms quickly

LC Operation

- LC also uses preprocessors for lossy compressors



LC Installation (CPU Version)

- Download LC using the following commands
  - git clone https://github.com/burtscher/LC-framework.git
  - cd LC-framework/
- Generate the framework as follows
  - ./generate_Host_LC-Framework.py
- Run the following command to compile LC
  - g++ -O3 -march=native -fopenmp -mno-fma -ffp-contract=off -DUSE_CPU -I. -std=c++17 -o lc lc.cpp

LC Usage

- Compress *input.dat* with "BIT_4 RLE_4" algorithm
  - ./lc input.dat CR "" "BIT_4 RLE_4"
- Have LC search for the best 2-stage algorithm
  - ./lc input.dat CR "" ".+ .+"
- Have LC search for the best 3-stage algorithm
  - ./lc input.dat CR "" ".+ .+ .+"
- If this takes too long, use the genetic algorithm
  - ./scripts/ga_search.py -s 3 input.dat

LC Usage (continued)

- If you also want throughput information, use
  - ./lc input.dat EX "" ".+ .+"
- Check out the LC tutorial for more information
  - GPU operation
  - Lossy compressors
  - Generating standalone compressor
  - Description of all supported transformations
  - Adding your own transformations to LC
  - URL: https://github.com/burtscher/LC-framework/