High Throughput and Low Latency Compression Engine

Sudarshan Kumar
Ratandeep S. Bhangal
Rohit Ranjan
Shreyank Amartya
Data Compression

- **SSD controller**
  - Increase capacity
  - Increase life/reliability:
    - Reduced write amplification
    - Less wear out
  - Performance: Improve read/write speed

- **Enterprise Server**
  - Increase capacity
  - Performance: Improve read/write speed

- **WAN/LAN, SERDES…**
  - Increased bandwidth: 2X
  - Less network slowness due to congestion

- **BIGDATA Analytics**
  - Lots of compression & decompression
  - Reduce disk space (Req 3x disk write)
  - Performance: Improve IOPs

Cost:
- Latency
- Area
- Power
High throughput Data Compression

Single instance instead of multiple instance

- Smaller area
- Lower power,
- Lower cost
- Very important in a price sensitive disk market

- Enables compression in
  - high speed enterprise servers
  - high speed networking (40G, 100G)
Data Compression Techniques

LOSSLESS

- Statical
  - Huffman coding
  - Arithmetic coding
  - LZ78
- Sequential
  - LZ(LEMPLEL ZIV) family
    - Dictionary Based
    - Faster
    - Lower compression: 1.5X worse than LZ77
    - Dictionary gets full
    - Eg LZW etc

LOSSY

- Sliding Window Based
  - Slower
  - Higher compression
  - Eg LZS, LZSS, LZO, LZ4 etc

For Picture & video

LZ advantage
- One pass
- Good compression
- Easy to implement
- No explicit Table
LZ77 Search Example

- Consists of
  - Search (compute intensive)
  - Encode
  - Eg: LONGS LONG AWAY FIND LONG LONGS!!!!!!!!!!!!!!

LZ77 (4,6) AWAY FIND (6,15)(5,26)(12,1)

Similar to RLE
Length > Offset
Multiple match
Offset points to nearest match
Encoding After Search

- **SERCH**
  - Literals, (length, offset)

- **ENCODE**
  - **Byte Boundry**
    - Less efficient
    - More bits wasted as flag
    - Faster decoding in software
    - Eg LZO, LZ4, Snappy
  - **Byte Boundry**
    - More efficient
    - 1 bit used as flag
    - Slower decoding in software
    - Eg LZSS, LZS
  - **Statistical Encode**
    - Deflate (LZ77 + Huffman)
    - More Compression
    - More efficient
    - Slower decoding in software (drawback)
    - Hardware decoding: fast
Implementation

Low Latency:
- Four pipeline stages

Variable “Sliding Window” Size:
- 2K to 8K

High Throughput:
- 4byte/clock, 8byte/clock
- Easy to increase throughput

“LAZY” match:
- Yields longer match

Output format:
- LZSS, LZS
- Output can be fed to Hoffman coder

Pipeline Stages

1. INPUT MEMORY
2. LZ77 SEARCH
3. ENCODE(LZSS, LZS, LZO)
4. OUTPUT MEMORY
5. PIPED To Other blocks: ENCRYPTION

Flash Memory Summit
Result

Latency:
4 cycle
Independent of input data size/type

Compression ratio:
Similar to published numbers.

Throughput @ 4byte/clock:
ASIC at 500Mhz=16Gbit/s
Custom@2Ghz=64Gbit/s

Throughput @ 8byte/clock:
ASIC at 500Mhz=32Gbit/s
Custom@2Ghz=128Gbit/s
FPGA at 200 Mhz=12.8Gbit/s

Compresssion Result
Calgary Corpus tests
Paper2=3.64 bits/byte
Bib=3.45 bits/byte
Progp=2.48 bits/byte
(LZSS)

Can increase throughput from 8byte/clock to 10byte/clock or 12byte/clock if needed
Thanks You