ESE532 Project P2 Report

Ritika Gupta, Taylor Nelms, and Nishanth Shyamkumar

November 6, 2019

1 Design Space Axes

1. **Axis:** S, Number of SHA-256 hardware units
   Challenge: Improving throughput of hashing step
   Opportunity: Send chunks to rotating SHA unit index to allow for parallel execution
   Continuum: Anywhere from 1 to however many of our hardware SHA units will fit on the FPGA
   Equation for Benefit: $\texttt{Throughput}(S) = S * \texttt{singleSHAUnitThroughput}$

2. **Axis:** L, Number of LZW hardware units
   Challenge: Improving throughput of LZW step
   Opportunity: Send chunks to rotating LZE unit index to allow for parallel execution
   Continuum: Anywhere from 1 to however many of our hardware LZW units will fit on the FPGA (BRAM likely limiting factor)
   Equation for Benefit: $\texttt{Throughput}(L) = S * \texttt{singleLZWUnitThroughput}$

3. **Axis:** Z, Design choice for LZW hash table unit
   Challenge: Allow for efficient access of code-table for LZW step while fitting within hardware specifications
   Opportunity: Use trees or associative memories (or both) to allow for low cycle count for finding relevant table entry
   Continuum: $Z \in$ {Tree with Dense RAM, Tree with Fully Associative Memory, Tree with Tree, Tree with Hybrid}
   Equation for Benefit: Slide 65 from Day 17 has the relevant tradeoff chart, with implied `implementation_complexity` parameter to consider.

4. **Axis:** H, Number of bits in hash of SHA value for storing SHA values
   Challenge: Effectively storing mapping between SHA values of previous chunks and the chunk index
   Opportunity: Tune hash table size to reduce conflicts but also remain compact
   Continuum: Could be any small number of bits (call it 5 as a low value) through 256 for the full SHA value.
   Equation for Benefit:

$$\texttt{numRows } C = 2^H$$

$$\texttt{probCollision} = \binom{N}{m}\left(\frac{1}{C}\right)^m\left(1-\frac{1}{C}\right)^{N-m}$$

5.

6.

7.

8. **Axis:** II_L, Pipelining II for LZW hardware implementation
   Challenge: Allow for quick compression algorithm
   Opportunity: Loosen pipelining constraints for LZW to reduce computational load
   Continuum: 1 to `MAX_CHUNK_SIZE`
   Equation for Benefit: $\texttt{Throughput}(\texttt{LZW}) = \frac{1byte}{II_L}$

9. **Axis:** W_L, LZW compression window size
   Challenge: Cut down on LZW memory requirements
   Opportunity: Restructure how encoding/decoding interprets data to reduce conceptual table depth from `MAX_CHUNK_SIZE` rows down to some smaller W_L
   Continuum: `MAX_CHUNK_SIZE` to 1 (the latter of which would make it stop being compression)
   Equation for Benefit: $\texttt{memRequirements}_{LZW} * = \frac{W_L}{\texttt{MAX\_CHUNK\_SIZE}}$
   Note: there are a number of things this change would affect, which is also highly dependent on Z (defined above). We will likely not change this, but it is a parameter that could be tuned.

10. **Axis:** N, Number of bytes at a time transfered to CDC unit
    Challenge: Balance memory transfer overhead against memory storage for incoming data
    Opportunity: Loosen pipelining constraints for LZW to reduce computational loa
    Continuum: 1 to $1MB$ (this could be a fake limit)
    Equation for Benefit: $\texttt{memTransferTime}_{total} = \frac{\texttt{totalInput}}{N} * (\texttt{memTransferOverhead} + N * \texttt{memTransferRate})$
11.
12.

2 Teamwork