ESE534: Computer Organization

Day 21: April 4, 2012
Lossless Data Compression

Today

- Basic Idea
- Example
- Systolic
- Tree
- Reclaiming space in tree
- CAMs

Lossless vs. Lossy

- Lossless – can reconstruct source perfectly (bit-identical)

uncompress(compress(x))=x

- Examples
  - Huffman
  - Run Length Coding
  - Lempel-Ziv
  - Unix compress/gzip
- Lossy – capture important elements of original, but maybe not all bits
- Examples
  - MP3
  - JPEG
  - MPEG

Dictionary Idea

- Send id for long string rather than all the characters

Dictionary Example

- "the instruction controls the behavior of the ALU, data memory, and interconnect on each cycle."
- Characters?
  - Bits at 8b/character
- Encoding with dictionary? Bits?

Code	Word
000	ALU
001	memory
010	interconnect
011	instruction
100	data
101	cycle
110	control
111	the

Dictionary Usability

- When can we do this?
- What might prevent us from pulling this trick?

Big Idea

- Use data already sent as the dictionary
  - Don't need to pre-arrange dictionary
  - Adapt to common phrases/idioms in a particular document

Example

- First line of Dr. Suess's *Green Eggs and Ham*
  - I AM SAM SAM I AM
- Recurring substrings?

Example

- An encoding:
- I AM S<2,3> <5,4><0,4>
- Decode.
- Characters in original?
  - Bits based on 8b characters?

Example

- An encoding:
- I AM S<2,3> <5,4><0,4>
- Encode:
  - Add 1 bit to identify character vs <x,y>
    - 9b characters
  - <x,y>: 1b says this + 4b for x, 4b for y
    - Also 9b
- How many bits?

Technical Issue

- How many bits assign to x and y?
- Issues?
- What if the document is huge?
  - What problems might that pose?

Windows

- Pragmatic solution
  - Only keep the last D characters
  - D is window size
  - Need $\log_2(D)$ bits to specify a position
  - Parameterize encoder based on D
  - Typically larger D → Greater compression

Encoding

- Greedy simplification
  - Encode by successively selecting the longest match between the head of the remaining string to send and the current window

Algorithm Concept

- While data to send
  - Find largest match in window
  - If length=1
    - Send character
  - Else
    - Send <x,y> = <match-pos,length>
  - Shift data encoded into window

Run Algorithm

- Use D=8
- I AM SAM SAM I AM

- How many bits?

What's challenging to implement?

- While data to send
  - Find largest match in window
  - If length=1
    - Send character
  - Else
    - Send <x,y> = <match-pos,length>
  - Shift data encoded into window

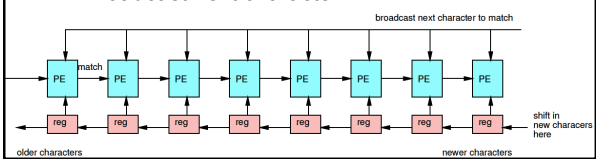
Systolic Algorithm

Give character in window a PE

- Broadcast characters to PE
- While (some PE has match-out=true)
  - match-out=new-search*match
    OR cont-search*match*match-in
  - Len=len+1
  - Broadcast next character
- Send <pos-last-match-out,len>
- Shift last set of characters into window

Systolic Hardware

- While (some PE has match-out=true)
  - match-out=new-search*match
    OR cont-search*match*match-in
  - Len=len+1
  - Broadcast next character



Simulate Systolic

- Each student is a PE in the window
  - Identify left and right neighbors
  - Raise right hand for match-out
  - Note left neighbor's hand at end of previous cycle to know match-in
- I AM SAM SAM I AM

Contemplate Solution

- How complicated is each PE?
- How fast PE?
- How fast does encoding operate?
- How much area do we need?
- How much energy?

Contemplate Solution

- What's inefficient or unsatisfying about this solution?

Tree Based

Idea

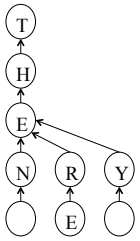
- Avoid need to track multiple substrings
- Compress storage

BY

- Storing common prefixes together in a tree

Tree Example

- THEN AND THERE, THEY STOOD…



Idea

- Avoid need to track multiple substrings
- Compress storage

BY

- Storing common prefixes together in a tree

Tree Algorithm

Root for each character

- Follow tree according to input until no more match
- Send <name of last tree node>
- Extend tree with new character
- Start over with this character

Run Algorithm

- I AM SAM SAM I AM

Encoding

- Encoding bits assuming D=512
  - So, 9b to encode tree node

Finite Window

- How can we maintain a finite window in this case?

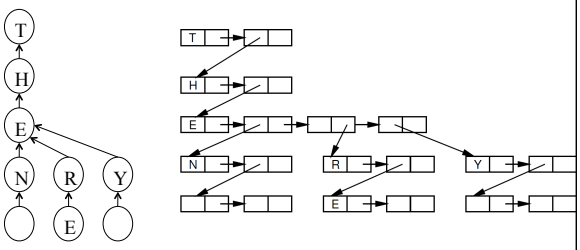
Finite Window

- Clear and start over
- LRU on tree nodes
- Maintain two areas
  - Encode from one (perhaps both)
  - Add to new
  - When new fills,
    - New->old, clear old
- Pick old leaf node to replace

Complexity

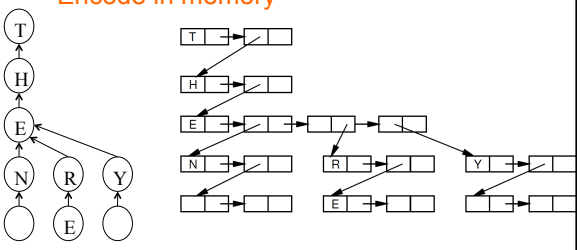
- How much work per character to encode?

Tree Node Representation



Tree Node Representation

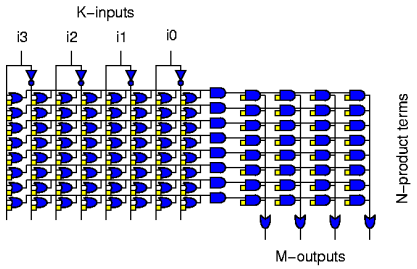
- Encode in memory



Content Addressable Memory

- What's a CAM?

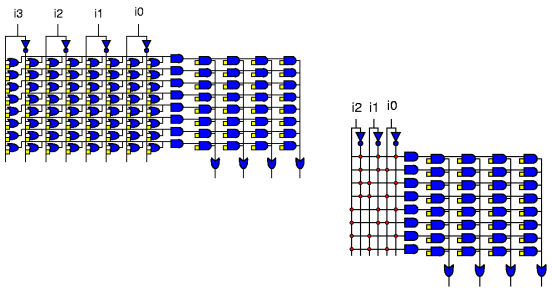
PLA



CAM

- Memory with Programmable Addresses
  - Capacity < $2^{(matchbits)}$
- PLA with both planes writeable

PLA and Memory



Contemplate

- What value do Bunton and Borriello get from using a CAM?

Admin

- Reading for Monday on Web
- FM1 for Monday
  - Implement tree version on processor and estimate energy

Big Ideas [MSB Ideas]

- Can often compress data without loss of information
- Exploit structure in data to encode
- Build dictionary based on data already sent
- Code repeating substrings compactly in terms of data already seen in recent past