CO3096 e-lecture 3

Lossless compression (conclusion)

Outline

- Applications of LZW/LZ77
- Overall comparison of lossless algorithms
  - Performance
  - Asymmetric/Symmetric
  - Adaptive/Nonadaptive
  - Input/Output: F2V/V2F/V2V…

LZ77 applications: QIC-122

- Standard for compressed 1/4-inch tapes.
  - Raw character is coded as 0<ASCII code> (1 byte)
  - Pointer is coded as 1<Offset><Length>, where Offset and Length may each be chosen from a pre-arranged set of values. E.g. Length = 4 bits and Offset = 11 bits.

LZ77: the ZIP family

- Highly optimised variation of the LZ77 algorithm.
- Uses hash table to locate matches in HB based on 3-character prefixes.
  - Restrict length of linked list to keep search costs down.
- Uses semi-adaptive Huffman coding to code pointers for each block of 64KB.
  - Huffman codes for the pointers in this block of 64KB are pre-pended to the block.

LZW apps: GIF

"Lossless" compression of small images.

- Normally alphabet size large (2^{24} ~ 16 million).
  - Not much hope for dictionary compression!
- 24-bit colour image is converted to 8-bit indexed colour image (in general lossy).
- Apply LZW to indexed colour image (lossless).

LZW: UNIX utility compress

- This is a widely-available LZW variant (.Z file extension). The key features are:
  - The number of bits used for representing tokens is gradually increased during encoding, so that only just enough bits are used to encode the tokens.
  - It moves from using 9-bit codes to 10-bit codes when inserting token 511 into dictionary.
- Limit size of dictionary & monitor performance. Rebuild from scratch if need be.
- The dictionary is represented using a *trie* data structure which makes coding and decoding very fast. [Compressing the trie: R. Geary & RR]

LZW apps: V.42 bis

- ITU-T standard for telephone-line modems.
  - Each modem has a pair of dictionaries, one for incoming data and one for outgoing data.
  - Maximum dictionary size is negotiated between the calling and called mode as the connection is made. (min 512 tokens with 6 chars/token.)
- Infrequently used tokens may be deleted from the dictionary.
  - Modem may also request that the called modem discard dictionary (e.g. a new file may be transmitted).
- Modem may switch to transmitting uncompressed data.

Conclusion (Part 1)

- We now summarise what we have said about lossless coding. We have done the following algorithms:
  1. Huffmann; Arithmetic [memoryless model]
  2. Run-Length Encoding [simple Markov model]
  3. LZ77, LZW, Burrows-Wheeler [Markov model]
- Real-life applications place constraints (and offer opportunities) to compression algorithms.

Comparison of Lossless Algos

- Difference between compression and decompression speeds (asymmetric/symmetric)
- A new classification of symbolic algorithms.
- Performance
  - Compression ratio
  - Overall speed

Asymmetric/Symmetric

- Asymmetric if compression slower than decompression (or vice-versa, in theory).
  - Asymmetric algorithms unsuitable for real-time bidirectional compression.

Asymmetric

- Huffman
  - Decoding reads a bit at a time, coding a character at a time. Not too significant.
- LZ77
  - Searching for a match in HB (coding) is slower than reading off the match (decoding).
- BZIP
  - Coding: O(n log n), for n symbols.
  - Decoding: O(n)
- Symmetric: Arithmetic, RLEs, LZW.

Adaptive or Not?

- Non-adaptive
  - Huffman, RLE
- Adaptive
  - Adaptive Huffman, arithmetic coding, LZW, LZ77.
- Semi-Adaptive
  - BZIP.

Input/Output

- Classification for "on-line" algorithms… (i.e. not semi-adaptive).
- F2V (Fixed to Variable)
  - Each "coding step" reads in a fixed amount of raw data, but outputs a variable amount of compressed data.
  - Huffman coding, adaptive HC.

Input/Output

- V2F (Variable-to-fixed)
  - Each coding step reads a variable amount of raw data but outputs a fixed-size compressed data
- LZ77, LZ78, RLE (in basic form, also e.g. IBM HDC compression)
  - Fax compression: V2F front-end (RLE) + F2V back-end (Huffman) = V2V.

Input/Output

- V2V (Variable to Variable)
  - Each coding step reads a variable amount of raw data and outputs a variable-sized compressed data.
- Arithmetic coding (real arithmetic coders).
  - Fax compression.
- Classification matters when mixing compression and communication. Matters even more for streaming diffuse data.

Performance Comparison (RR)

- The input file: mailbox of size 5.53MB in size. Run time on Sun UltraSPARC-II server.

	compressed file size	compression ratio	compression time	decompression time
bzip	1.10MB	5.53/1.10 = 5.05	20.72s	7.37s
gzip -9	1.55MB	5.53/1.55 = 3.55	15.31s	1.40s
gzip -6	1.56MB	5.53/1.56 = 3.54	9.52s	1.39s
compress	2.25MB	5.53/2.25 = 2.46	1.52s	0.95s
pack	3.57MB	5.53/3.67 = 1.50	0.85s	1.81s

Conclusions

Draw your own!