The LZW Approach

Key idea: Each ***codeword*** represents multiple symbols.

- Start with ‘trivial’ codeword table where each codeword corresponds to one ASCII symbol.
- Every time a codeword X is used, record a new codeword Y corresponding to X concatenated with the next symbol.

Example begins on next slide for

B=”aababcabcdabcdeabcdefabcdefgabcdefgh”

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂

C(B) =

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- Best prefix match in codeword table is a, so output 0x61.
- Also add aa to table.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂

C(B) = 0x61

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- Best prefix match in codeword table is a, so output 0x61.
- Also add aa to table.

C(B) = 0x61

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- Best prefix match in codeword table is a, so output 0x61.
- Also add ab to table.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa

C(B) = 0x6161

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- Best prefix match in codeword table is a, so output 0x61.
- Also add ab to table.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

C(B) = 0x6161

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- Best prefix match in codeword table is b, so output 0x62.
- Also add ba to table.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

C(B) = 0x616162

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- Best prefix match in codeword table is b, so output 0x62.
- Also add ba to table.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba

C(B) = 0x616162

The LZW Approach: Codewords for Multiple Symbols

Example: B = "aababcabcdabcdeabcdefabcdefgabcdefgh"

- Best prefix match in codeword table is ab, so output 0x81.
- Also add abc to table.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc

C(B) = 0x61616281

Only half as many bits!

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- Best prefix match in codeword table is c, so output 0x63.
- Also add ca to table.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc
0x84	ca

C(B) = 0x6161628163

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- Best prefix match in codeword table is ???, so output ???.
- Also add ??? to table.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc
0x84	ca
???	???

C(B) = 0x6161628163

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- Best prefix match in codeword table is abc, so output 0x83.
- Also add abcd to table.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc
0x84	ca
0x85	abcd

C(B) = 0x616162816383

LZW Prefix Matching

How should we store our codeword table to easily support fast prefix matching?

- B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc
0x84	ca
0x85	abcd

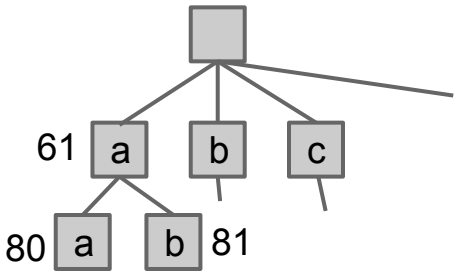
LZW Prefix Matching

How should we store our codeword table to easily support fast prefix matching? A trie mapping strings to codewords.

- B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc
0x84	ca
0x85	abcd



The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- Best prefix match in codeword table is d, so output 0x64.
- Also add da to table.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc
0x84	ca
0x85	abcd
0x86	da

C(B) = 0x61616281638364

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- What will be the codeword for abcdefg (to think about when reviewing this material later)? What will be C(B)?

C(B) = 0x61616281638364

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc
0x84	ca
0x85	abcd
0x86	da

The LZW Approach: Codewords for Multiple Symbols

Example: B = “aababcabcdabcdeabcdefabcdefgabcdefgh”

- abcdefg will be 0x8b.

288 bits.

C(B) =
0x6161628163836485 65
8766896 78b68

120 bits.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc
0x84	ca
0x85	abcd
0x86	da
0x87	abcde
0x88	ea
0x89	abcdef

0x8a	fa
0x8b	abcdefg
0x8c	ga
0x8d	abcdefgh

Decoding

The Decompression Challenge

Suppose we're given a bitstream that we need to decompress.
C(B) = 0x616162816383, but don't have table used for encoding.

- B clearly starts with aab, but what is 0x81?

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂

Decompressing LZW

Key idea: After processing each codeword, add the codeword that would have been added by the *previous* character.

- Start with ‘trivial’ codeword table where each codeword corresponds to one ASCII symbol.
- When codeword X is encountered, return appropriate symbols S(X) given in codeword table.
- Given consecutive codewords X_1 and X_2add codeword corresponding to [$S(X_1)$ + firstCharacterOf($S(X_2)$)]

Example on following slides for : C(B) = 0x61616281638364

LZW Decompression

C(B) = 0x61616281638364

- 0x61 is a in the codeword table.
- No previous codeword so add nothing.

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂

B = “a”

LZW Decompression

C(B) = 0x61616281638364

- 0x61 is a in the codeword table, so output “a”
- S(0x61)=”a” and S(0x61)=”a”, so add “aa”

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa

B = “aa”

LZW Decompression

C(B) = 0x6161<u>62</u>81638364

- 0x<u>62</u> is b in the codeword table, so output “b”.
- S(0x61)=”a” and S(0x62)=”b”, so add “ab”

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

B = “aa<u>b</u>”

LZW Decompression

C(B) = 0x6161681638364

- 0x81 is ab in the codeword table, so output “ab”
- S(0x62)=”b” and S(0x81)=”ab”, so add “ba”

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba

B = “aabab”

LZW Decompression

C(B) = 0x6161628163 8364

- 0x63 is c in the codeword table, so output “c”
- S(0x81)=”ab” and S(0x63)=”c”, so add “abc”

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc

B = “aababc”

LZW Decompression

C(B) = 0x6161628163836 4

- 0x83 is ??? in the codeword table, so output “???”
- S(???)=”???” and S(???)=”???”, so add “???”

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc

B = “aababc???”

LZW Decompression

C(B) = 0x6161628163836̲4

- 0x83 is abc in the codeword table, so output “abc”
- S(0x63)=”c” and S(0x83)=”abc”, so add “ca”

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc
0x84	ca

B = “aababcabc”

LZW Decompression

C(B) = 0x6161628163864

- 0x64 is d in the codeword table, so output “d”
- S(0x83)=”abc” and S(0x64)=”d”, so add “abcd”

0x61	a
0x62	b
0x63	c
...	
0x7e	~
0x7f	⌂
0x80	aa
0x81	ab

0x82	ba
0x83	abc
0x84	ca
0x85	abcd

B = “aababcabcd”

LZW Decompression: Special Case

LZW Compression (exactly like before)

B=”OXOXOXO”

- Best prefix match is O, so output 0x4F.
- Add codeword “OX”

…	…
0x4F	O
…	…
0x58	X
…	…
0x7f	⌂
0x80	OX

C(B) = 0x4F

Compressing this is exactly as we did before, but this one will turn out to be slightly trickier to decompress (as we’ll soon see).

LZW Compression (exactly like before)

B=”OXOXOXO”

- Best prefix match is X, so output 0x58.
- Add codeword “XO”

…	…
0x4F	O
…	…
0x58	X
…	…
0x7f	⌂
0x80	OX
0x81	XO

C(B) = 0x4F58

LZW Compression (exactly like before)

B=”OXOXOXO”

- Best prefix match is OX, so output 0x80.
- Add codeword “OXO”

...	...
0x4F	O
...	...
0x58	X
...	...
0x7f	⌂
0x80	OX
0x81	XO

0x82	OXO

C(B) = 0x4F5880

LZW Compression (exactly like before)

B=”OXOXOXO”

- Best prefix match is OXO, so output 0x82.
- Don’t add anything.

...	...
0x4F	O
...	...
0x58	X
...	...
0x7f	⌂
0x80	OX
0x81	XO

0x82	OXO

C(B) = 0x4F588082

LZW Compression (exactly like before)

B=”OXOXOXO”

- Best prefix match is OXO, so output 0x82.
- Don’t add anything.

...	...
0x4F	O
...	...
0x58	X
...	...
0x7f	⌂
0x80	OX
0x81	XO

0x82	OXO

C(B) = 0x4F588082

LZW Decompression

C(B) = 0x4F588082

- 0x4F is O in the codeword table, so output “O”
- No previous codeword, so output nothing.

...	...
0x4F	O
...	...
0x58	X
...	...
0x7f	⌂

B = “O”

LZW Decompression

C(B) = 0x4F588082

- 0x58 is X in the codeword table, so output “X”
- S(0x4F)=”O” and S(0x58)=”X”, so add “OX”

…	…
0x4F	O
…	…
0x58	X
…	…
0x7f	⌂
0x80	OX

B = “OX”

LZW Decompression

C(B) = 0x4F588082

- 0x80 is OX in the codeword table, so output “OX”
- S(0x58)=”X” and S(0x80)=”OX”, so add “XO”

…	…
0x4F	O
…	…
0x58	X
…	…
0x7f	⌂
0x80	OX
0x81	XO

B = “OXOX”

LZW Decompression

C(B) = 0x4F588082

- 0x82 is … uh oh.

…	…
0x4F	O
…	…
0x58	X
…	…
0x7f	⌂
0x80	OX
0x81	XO

The problem is that the compression algorithm looked ahead one character to created new codewords. When decompressing we look BACK to create new codewords!

Solution: Figure out what 82 is going to be by looking back.

B = “OXOX?????”

LZW Decompression

C(B) = 0x4F588082

- 0x82 is ****.
- S(0x80) is “OX” and S(0x82) is “****”, so add “****”

All the same thing!

B = “OXOX****”

...	...
0x4F	O
...	...
0x58	X
...	...
0x7f	⌂
0x80	OX
0x81	XO

LZW Decompression

C(B) = 0x4F588082

We know that **** must start with OX.

- 0x82 is ****.
- S(0x80) is “OX” and S(0x82) is “OX**”, so add “OX**”

...	...
0x4F	O
...	...
0x58	X
...	...
0x7f	⌂
0x80	OX
0x81	XO

B = “OXOX****”

LZW Decompression

C(B) = 0x4F588082

We know that ** is just the leftmost character of 0x82!

- 0x82 is ****.
- S(0x80) is “OX” and S(0x82) is “OX**”, so add “OX**”

...	...
0x4F	O
...	...
0x58	X
...	...
0x7f	⌂
0x80	OX
0x81	XO

B = “OXOX****”

LZW Decompression

C(B) = 0x4F588082

We also know that ** is just the leftmost character of 0x82!

- 0x82 is ****.
- S(0x80) is “OX” and S(0x82) is “OXO”, so add “OXO”

...	...
0x4F	O
...	...
0x58	X
...	...
0x7f	⌂
0x80	OX
0x81	XO

0x82	OXO

B = “OXOX****”

LZW Decompression

C(B) = 0x4F588082

We also know that ** is just the leftmost character of 0x82!

- 0x82 is OXO.
- S(0x80) is “OX” and S(0x82) is “OXO”, so add “OXO”

...	...
0x4F	O
...	...
0x58	X
...	...
0x7f	⌂
0x80	OX
0x81	XO

0x82	OXO

B = “OXOXOXO”