

p 691  
MR

#### 4 Data structures and algorithms

2002

Arithmetic encoding compactly represents a string of characters by an enormously precise number in the range  $[0,1)$  represented in binary by a finite sequence of digits following the decimal point. What is remarkable is that this number can be processed efficiently using only fixed point arithmetic on reasonably small integers. As a demonstration, if the original text contained only the characters A, B, C and the end-of-file mark w, such text can be arithmetically encoded using only three bit arithmetic. Illustrate how it can be done by decoding the string 101101000010 on the assumption that the character frequencies are such that the decoding tables of size 8 and 6 are, respectively, wAABBCCC and wABBCC. The first few lines of your working could be as follows:

	0	0	0	0	1	1	1	1				
	0	0	1	1	0	0	1	1				
	0	1	0	1	0	1	0	1				
101 101000010		-w	---	A	---	A	---	B	---	B	-(C)++C+++C+	=> C

Your answer should include a brief description of how the decoding algorithm works.  
[20 marks]

ANSWER:

GO

Bookwork. The answer is page ~~69~~ of the lecture notes.

## 14.4.2 Decoding of 101101000010

```

101 101000010      0  0  0  0  1  1  1  1
                    0  0  1  1  0  0  1  1
                    0  1  0  1  0  1  0  1
101 101000010      |-w---A---A---B---B---(C)++C+++C+|      => C
                    =1===1===1=
                    0  1  1
                    1  0  1
double
011 01000010      0  0  1  1  1  1  1
                    1  1  0  0  1  1
                    0  1  0  1  0  1
011 01000010      |-w---(A)++B---B---C---C-|      => A
                    =0=
                    1
                    1
double
110 1000010      =1===1=
                    1  1
                    0  1
double
101 000010      =1===1===1===1=
                    0  0  1  1
                    0  1  0  1
double
010 00010      0  0  0  0  1  1  1  1
                    0  0  1  1  0  0  1  1
                    0  1  0  1  0  1  0  1
010 00010      |-w---A++(A)++B---B---C---C---C-|      => A
                    =0===0=
                    0  1
                    1  0
double
100 0010      0  0  1  1
                    1  1  0  0
                    0  1  0  1
double#
1#00 010      0  0  0  0  1  1  1  1
                    1###1###1###1###0###0###0###0
                    0  0  1  1  0  0  1  1
                    0  1  0  1  0  1  0  1
1#00 010      |-w---A---A---B++(B)++C---C---C-|      => B
                    0  1
                    1###0
                    1  0
                    1  0
double#
1##00 10      0  0  1  1
                    1###1###0###0
                    1###1###0###0
                    1  1  0  0
double#
1###01 0      0  0  0  0  1  1  1  1
                    1###1###1###1###0###0###0###0
                    1###1###1###1###0###0###0###0
                    1###1###1###1###0###0###0###0
                    0  0  1  1  0  0  1  1
                    0  1  0  1  0  1  0  1
1###01 0      |-w---A---A---B---B---(C)++C+++C+|      => C
                    =1===1===1=
                    =0===0===0=
                    =0===0===0=
                    =0===0===0=
                    0  1  1
                    1  0  1
double
010      0  0  1  1  1  1
                    1  1  0  0  1  1
                    0  1  0  1  0  1
010      |(w)++A---B---B---C---C-|      => w

```