ARITHMETIC CODING

ANHAR
anhar19@gmail.com

Arithmetic Coding

- Message dibangun dari sumber $S = \{x_1, \ldots, x_n\}$; Probabilitas $P = \{p_1, \ldots\ldots, p_n\}$;
- Pada Arithmetic coding message dikodekan sbg bilangan dari interval [0,1) = {y:0≤y<1}
- Bilangan didp dg perluasan sesuai dg prob. kemunculan simbol saat diproses/dikodekan
- Dilakukan dg menggunakan satu set range interval ditentukan dg prob pd P

  IR = $\{[0,p_1),[p_1,p_1+p_2),[p_1+p_2,p_1+p_2+p_3),\ldots[p_1+\ldots+p_{n-1},p_1+\ldots+p_n)\}$

  Atau dalam prob cumulative: $P_i = \sum_{j=1}^{i} p_j$

  IR = $\{[0,P_1),[P_1,P_2),[P_2,P_3), \ldots ,[P_{n-1},1)\}$
- Ini adalah subinterval dari interval [0,1), tetapi dlm arithmetic coding, juga menentukan pembagian proposional dari interval lain [L,R) yg dimuat dlm [0,1) kedlm subinterval:

  $IR_{[L,R]}$ = $\{[L,L+(R-L)P_1),[L+(R-L)P_1,L+(R-L)P_2),[L+(R-L)P_2,L+(R-L)P_3),\ldots[L+(R-L)P_{n-1},L+(R-L))]$

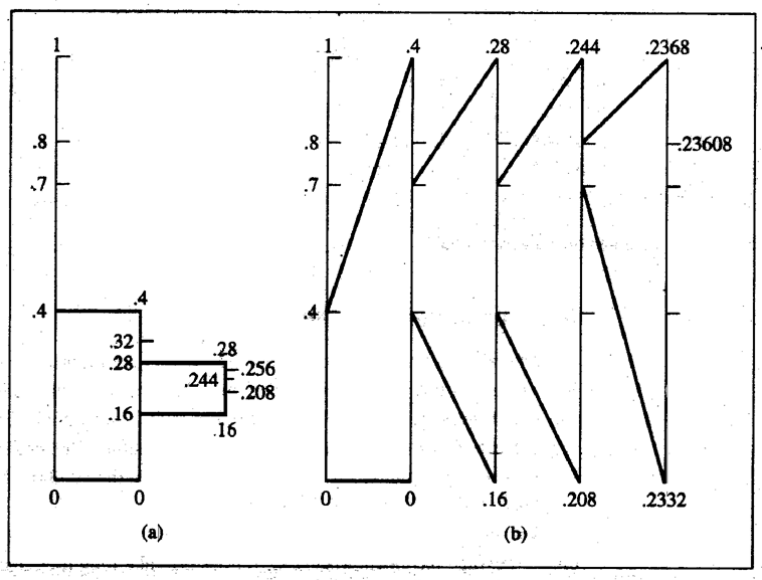
Algoritma Arithmetic Encoding

```
ArithmeticEncoding(message)
    currentInterval = [0,1);
    while the end of message is not reached
        read letter x_i from the message;
        divide currentInterval into subintervals IR_currentInterval;
        currentInterval = subinterval_i in IR_currentInterval;
    output bits uniquely identifying currentInterval;
```

Contoh

- Dg S = {A,B,C,#} dan P = {0.4, 0.3, 0.1, 0.2} kodekan message ABBC#
- A huruf pertama dari message dan huruf pertama dari S
  → subinterval pertama [0,0.4) dipilih dari *CurrentInterval* [0,1)
- Huruf kedua message B, juga kedua dlm S
  → subinterval kedua:
  [0+(0.4-0)*0.4, 0+(0.4-0)*(0.4+0.3)) = [0.16,0.28) dari *CurrentInterval* [0,0.4)
- Huruf ketiga message B, menyebabkan subinterval kedua dari *CurrentInterval* dipilih:
  [0.16+(0.28-0.16)*0.4, 0.16+(0.28-0.16)*(0.4+0.3)) = [0.208, 0.244)
- Setelah memproses huruf C, *CurrentInterval* sama dg:
  [0.208+(0.244-0.208)*(0.4+0.3), 0.208+(0.244-0.208)*(0.4+0.3+0.1)) = [0.2332,0.2368)
- Dan setelah simbol message kelima #, subinterval adalah:
  [0.2332+(0.2368-0.2332)*(0.4+0.3+0.1), 0.2332+(0.2368-0.2332)* (0.4+0.3+0.1+0.2)) = [0.23608,0.2368)
- Codeword adalah sembarang bilangan dalam CurrentInterval
  - Biasanya batas kiri : 0.23608, atau
  - Rata-rata arithmetic : (0.23608+0.2368)/2 = 0.23644

Contoh (cont.)



Contoh (cont.)

current-Interval	length	input letter	subintervals			
[0, 1)	1	*A*	[0, .4)	[.4, .7)	[.7, .8)	[.8, 1)
[0, .4)	.4 = p_1	*B*	[0, .16)	[.16, .28)	[.28, .32)	[.32, .4)
[.16, .28)	.12 = p_1p_2	*B*	[.16, .208)	[.208, .244)	[.244, .256)	[.256, .28)
[.208, .244)	.036 = $p_1p_2p_2$	*C*	[.208, .2224)	[.2224, .2332)	[.2332, .2368)	[.2368, .242)
[.2332, .2368)	.0036 = $p_1p_2p_2p_3$	#	[.2332, .23464)	[.23464, .23572)	[.23572, .23608)	[.23608, .2368)
[.23608, .2368)	.00072 = $p_1p_2p_2p_3p_4$					

Algortima Arithmetic Decoding

```
ArithmeticDecoding(codeword)
    currentInterval = [0,1);
    while (1)
        divide currentInterval into subintervals IR_currentInterval;
        determine the subinterval_i of currentInterval to which codeword belongs;
        output letter x_i corresponding to this subinterval;
        if x_i is the symbol '#'
            return;
        currentInterval = subinterval_i in IR_currentInterval;
```

Contoh Decoding

- Menggunakan contoh sebelumnya, lakukan proses decoding utk codeword 0.23608

current-Interval	subintervals				ouput letter
[0, 1)	[0, .4)	[.4, .7)	[.7, .8)	[.8, 1)	*A*
[0, .4)	[0, .16)	[.16, .28)	[.28, .32)	[.32, .4)	*B*
[.16, .28)	[.16, .208)	[.208, .244)	[.244, .256)	[.256, .28)	*B*
[.208, .244)	[.208, .2224)	[.2224, .2332)	[.2332, .2368)	[.2368, .242)	*C*
[.2332, .2368)	[.2332, .23464)	[.23464, .23572)	[.23572, .23608)	[.23608, .2368)	#

Implementasi Arithmetic Coding

Algoritma Arithmetic coding, tidak bisa langsung diimplementasikan:

- Akhir message harus ditandai (dlm contoh simbol #)
- Makin panjang message range semakin kecil →keterbatasan presisi komputer
- Solusi mengoutputkan digit setelah decimal point jika mempunyai digit yg sama utk lower dan upper bound dari *CurrentInterval* dan men-double-kan panjang interval

Implementasi Arithmetic Coding

Metoda Binary Arithmetic Coding: secara sistematis men-double-kan currentInterval jika panjangnya kurang dari 0.5. Ada 3 kasus:

1. Jika C*urrentInterval* = [$0.0b_1b_2$…,$0.0b_1$'b_2'…), jika berada setengah interval pertama dari interval [0,1), maka kedua batas lebih kecil dari 0.5 = $.1_2$; keduanya memp. 0 setelah floating point. Bit di shift out dan dioutputkan sbg bagian dari codeword. Menggeser (shifting) kekiri sama spt men-double-kan → ukuran interval double
2. Hasil sama didp jika kedua batas lebih besar $.1_2$, berarti kedua batas memp. bit 1 setelah floating point. Bit di shift out, ukuran interval di-double
3. Masalah terjadi jika $.1_2$ ada dlm currentInterval. Bit pertama stlh radix point berbeda pd kedua batas. Ukuran interval di-double-kan hanya jika batas bawah ≥ 0.25 dan batas atas < 0.75 (interval berada pada quarter kedua dan ketiga dari interval [0,1). Keputusan bit yg akan dioutputkan ditunda, hanya proses men-double-kan 'dicatat'. Ada tiga kemungkinan:

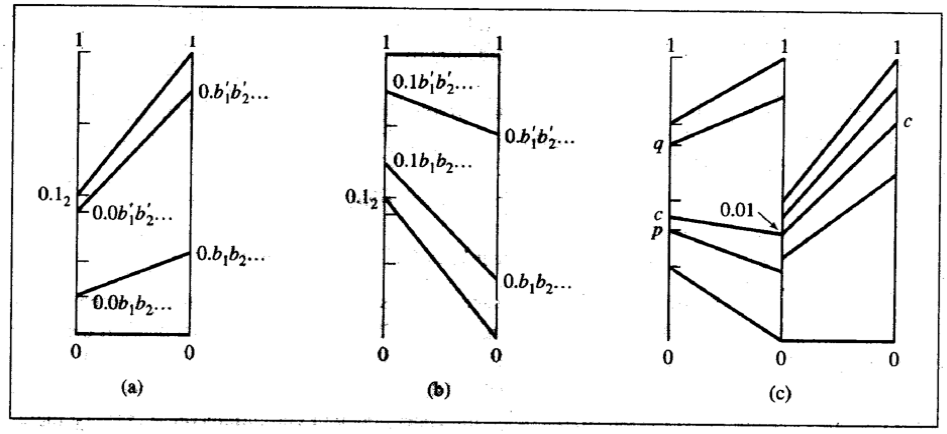
Implementasi Arithmetic Coding

Tiga kemungkinan jika interval [p,q) ada dlm quarter kedua dan ketiga dari interval [0,1)

- Jika *CurrentInterval* = [p,q) dan codeword c yg akan dibangkitkan dibawah titik tengah, bit 0 ditambahkan pd codeword. Setelah itu jika subinterval [p,q) yang memuat c diperluas dan c jatuh di atas titik tengah, maka bit berikutnya yg ditambahkan ke c adalah 1
- Sebaliknya jika c di atas titik tengah, maka bit yang di-outputkan 1, dan jika setelah perluasan subinterval [p,q) yg memuat c, c jatuh di bawah titik tengah, maka bit 0 ditambahkan ke c
- Jika setelah perluasan *CurrentInterval* yg baru tetap pd quarter kedua dan ketiga dari interval [0,1), keputusan kembali ditunda; setelah proses mencapai titik dimana *CurrentInterval* di bawah titik tengah, 0 di-outputkan diikuti dg 1 sebanyak *CurrentInterval* berada pada 0.25 dan 0.75. Jika *CurrentInterval* di atas titik tengah, 1 di-outputkan diikuti dg 0 sebanyak *CurrentInterval* berada pada 0.25 dan 0.75.

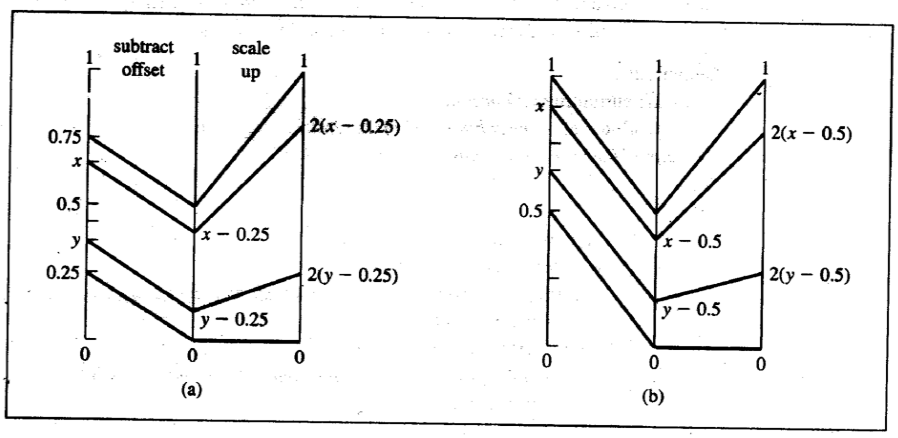
Implementasi Arithmetic Coding

- Scaling currentInterval jika ada pada interval:
  (a) [0, 0.5), (b) [0.5, 1) dan (c) [0.25, 0.75)

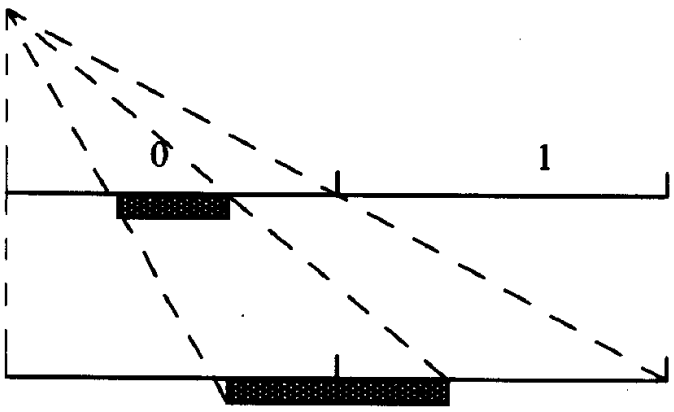


Implementasi Arithmetic Coding

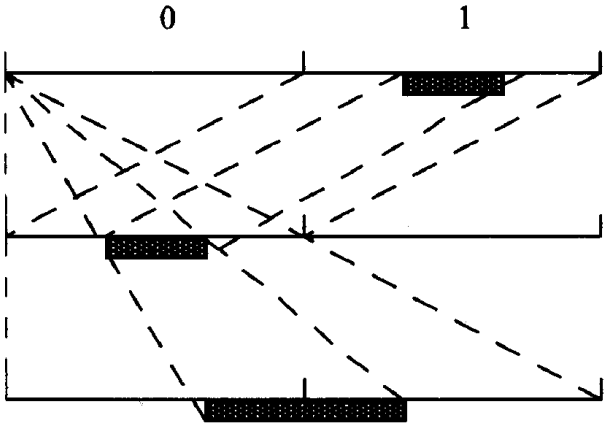
- Scaling currentInterval jika tercakup pada interval:
  (a) [0, 0.5), (b) [0.5, 1)



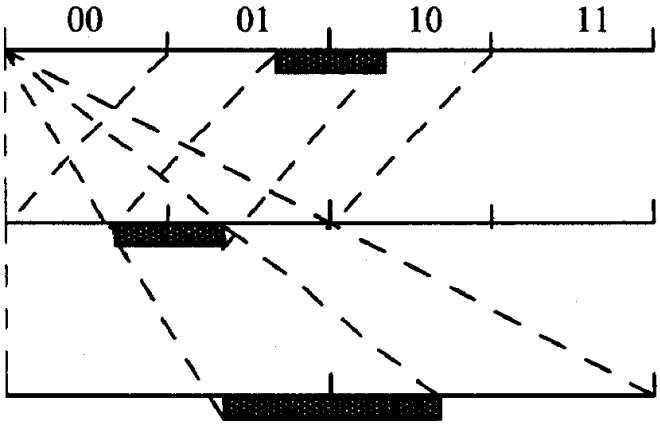
Batas Atas & Bawah pada Setengah Bagian Pertama

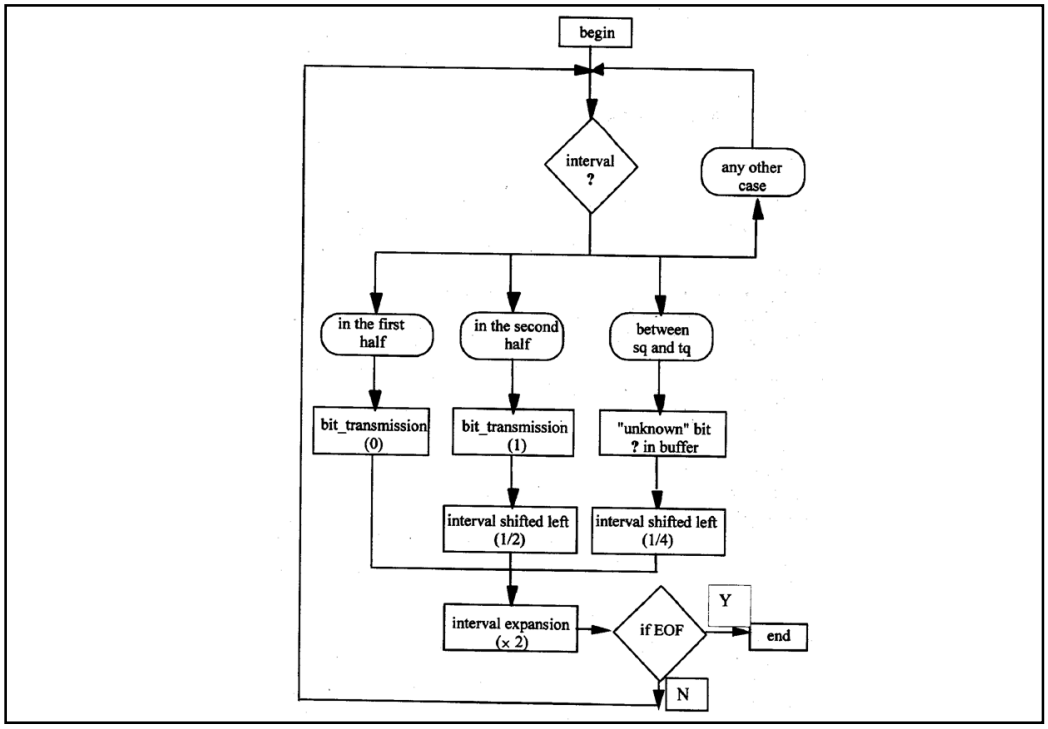


Batas Atas & Bawah pada Setengah Bagian Kedua



Interval pada Quarter Kedua dan Ketiga





Implementasi Arithmetic Coding

- Prosedur di atas diringkaskan:

```
OutputBits()
    while (1)
        if currentInterval ⊂ [0, .5)
            output 0 and bitCount 1s;
            bitCount = 0;

        else if currentInterval ⊂ [.5, 1)
            output 1 and bitCount 0s;
            bitCount = 0;
            subtract .5 from left and right bounds of currentInterval;
        else if currentInterval ⊂ [.25, .75)
            bitCount++;
            subtract .25 from left and right bounds of currentInterval;
        else break;
        double left and right bounds of currentInterval;
```

Finish Arithmetic Encoding

```
FinishArithmeticEncoding()
    bitCount++;
    if lower bound of currentInterval < .25
        output 0 and bitCount 1s;
    else output 1 and bitCount 0s;
```

Algoritma Lengkap Arithmetic Encoding

```
ArithmeticEncoding(message)
    currentInterval = [0, 1);
    bitCount = 0;
    while the end of message is not reached
        read letter x_i from the message;
        divide currentInterval into subintervals IR_currentInterval;
        currentInterval = subinterval_i in IR_currentInterval;
        OutputBits();
    FinishArithmeticEncoding();
```

Contoh

- S = {A,B,C,#}; P={0.4, 0.3, 0.1, 0.2}
- Kodekan ABBC#

current-Interval	input letter	output bit	subintervals			
[0, 1)	*A*		[0, .4)	[.4, .7)	[.7, .8)	[.8, 1)
[0, .4)		0				
[0, .8)	*B*		[0, .32)	[.32, .56)	[.56, .64)	[.64, .8)
[.32, .56)		—				
[.14, .62)	*B*		[.14, .332)	[.332, .476)	[.476, .524)	[.524, .62)
[.332, .476)		01				
[.664, .952)		1				
[.328, .904)	*C*		[.328, .5584)	[.5584, .7312)	[.7312, .7888)	[.7888, .904)
[.7312, .7888)		1				
[.4624, .5776)		—				
[.4248, .6552)		—				
[.3496, .8104)	*#*		[.3496, .53392)	[.53392, .67216)	[.67216, .71824)	[.71824, .8104)
[.71824, .8104)		100				
[.43648, .6208)		—				
[.37296, .7416)		—				
[.24592, .9832)		0111				

- Code: $0.001111000111_2 = 0.236083984375$

In-Class Excersise

- Contoh Soal:

  S = {a, b, c, #}; P = {0.3, 0.5, 0.1, 0.1}

  Kodekan message bbac# menggunakan Arithmetic Coding?

Prosedur Decoding

```
OutputLetter()
    divide currentInterval into subintervals IR_currentInterval;
    currentInterval = the subinterval_i of currentInterval to which bitBuffer belongs;
    output letter x_i corresponding to currentInterval;
    while (1)
        if currentInterval ⊂ [0, .5)
            ; // no action;
        else if currentInterval ⊂ [.5, 1)
            subtract 0.5 from bitBuffer and from left and right bounds of
                currentInterval;
        else if currentInterval ⊂ [.25, .75)
            subtract 0.25 from bitBuffer and from left and right bounds of
                currentInterval;
        else break;
        double left and right bounds of currentInterval;
        shift out one bit from bitBuffer and shift in one bit to bitBuffer;
```

Contoh Decoding

- Decoding codeword 001111000111

current-Interval	bitBuffer	input bits	bitBuffer in decimal		chosen subinterval	output letter
[0, 1)	001111	000111	.234375		[0, .4)	*A*
[0, .4)	001111	000111	.234375			
[0, .8)	011110	00111	.45875		[.32, .56)	*B*
[.32, .56)	011110	00111	.45875			
[.14, .62)	011100	0111	.4375	(–.25)	[.332, .476)	*B*
[.332, .476)	011100	0111	.4375			
[.664, .952)	111000	111	.8750			
[.328, .904)	110001	11	.765625	(–.5)	[.7312, .7888)	*C*
[.7312, .7888)	110001	11	.765625			
[.4624, .5776)	100011	1	.546875	(–.5)		
[.4248, .6552)	100111		.609475	(–.25)		
[.3496, .8104)	101110		.71875	(–.25)	[.71824, .8104)	#
[.71824, .8104)	011100		.21875	(–.5)		
[.43648, .6208)	011000		.375	(–.25)		
[.37296, .7416)	010000		.25	(–.25)		
[.24592, .9832)						