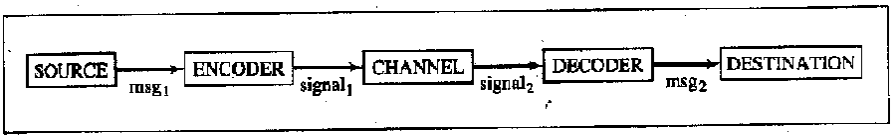
Informasi dan Coding

Anhar
anhar19@gmail.com

Elemen Dasar Proses Komunikasi

- Proses komunikasi



- Isue (kuliah)
  - Kompresi
  - Informasi

Informasi dan Entropy

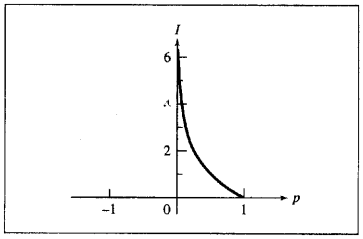
- Apakah informasi dan bagaimana mengukurnya?
- Mana yang memuat 'lebih banyak' informasi?
  - Besok matahari akan terbit
  - Harga BBM di Indonesia turun
- 'nilai' informasi ~ *surprise, unexpectedness, uncertainty*
- Jumlah kombinasi nilai informasi dari kejadian (*event*) yg tidak berelasi ~ jumlah nilai informasi masing-masing kejadian (mempunyai harga yang lebih kecil jika kejadian-kejadian berelasi)
  - Hari ini hujan + Saya tidak perlu menyiram taman
  - Hari ini hujan + Ada halaman yang hilang dari textbook saya
- Intuisi di atas menjadi basis dari teori informasi yang diusulkan Claude E Shannon (1948)

Informasi dan Entropy

- Set event: $S = \{x_1, \ldots, x_n\}$
- S disebut alphabet jika x_i sebuah simbol (huruf) digunakan utk membangun pesan (message)
- Probabilitas kemunculan masing-masing event, $p(x_i) = p_i$
- $P = \{p_1, \ldots, p_n\}$, dimana $p_i \geq 0$, $\sum_{i=1}^{n} p_i = 1$
- Untuk sumber memoryless:
  - Nilai *self-information* yg berhub. dg event x_i digunakan definisi
    $I(x_i) = -\log_k p_i$
  - Fungsi di atas adalah ukuran informasi (*surprise* atau *unexpectedness*) dari kemunculan event x_i

Informasi dan Entropy

- Fungsi self-information *I*



- Unit informasi (*uncertainty*) disebut bit jika digunakan algoritma dengan basis 2 ($\lg x = \log_2 x$)

Informasi dan Entropy

- Untuk sumber biner
  - $S = \{x_1, x_2\}$, $P = \{½, ½\}$
    $I(x_1) = -\lg ½ = 1$ bit $\quad I(x_2) = -\lg ½ = 1$ bit
  - $S = \{x_1, x_2\}$, $P = \{¼, 3/4\}$,
    $I(x_1) = -\lg ¼ = 2$ bit $\quad I(x_2) = -\lg 3/4 = 0{,}415$ bit
- Fungsi *I* hanya fokus pada satu event
  - pada kebanyakan situasi (kompresi data) lebih relevan mengukur content informasi pada keseluruhan set
    → Konsep Shannon: entropy → ukuran *uncertainty* satu set event

Entropy

$S = \{x_1, \ldots, x_n\}$, satu set event independen
$P = \{p_1, \ldots, p_n\}$, probabilitas kemunculan

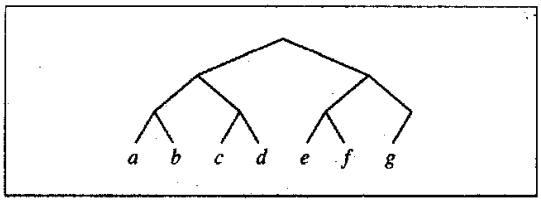
Entropy:

$$H(p_1, \ldots, p_n) = H(S) = -\sum_{i=1}^{n} p_i \lg p_i$$

Entropy = rata-rata self-information kemunculan event x_i

Entropy

- Entropy dapat juga diinterpretasikan jumlah rata-rata minimum dari jumlah pertanyaan ya/tidak untuk menentukan harga spesifik dari variabel *X*



- Dalam konteks coding message, entropy merepresentasikan batas bawah (*lower bound*) dari jumlah rata-rata bit per satu nilai input – yaitu rata-rata panjang code word digunakan untuk mengkodekan input

Entropy

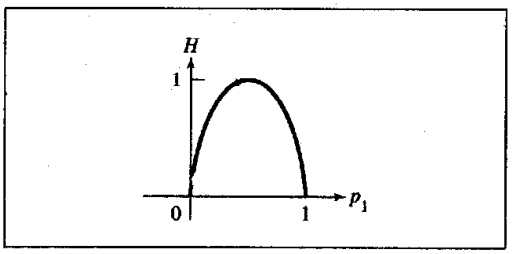
Contoh

- Untuk sumber biner, set probabilitas

$P = \{p_1, p_2\} = \{p_1, 1\text{-}p_1\}$

$H(p_1,p_2) = -p_1 \lg p_1 - p_2 \lg p_2$

$= -p_1 \lg p_1 - (1 - p_1)\lg(1 - p_1) = H(p_1)$



Entropy

Contoh

- Jika $S = \{x_1, x_2, x_3\}$, $P = \{½,¼,¼\}$, maka:

$H(p_1,p_2,p_3) = - ½\lg½ - ¼\lg¼ - ¼\lg¼$

$= ½ + ½ + ½ = 1{,}5$ bit/event

Karakteristik Entropy

Karakteristik-karakteristik fungsi Entropy *H*:

- Symmetry dari *H* : urutan dari argumen *H* tidak berpengaruh
- Fungsi *H* mempunyai batas atas dan bawah:
  $0 = H(1,0,\ldots,0) \leq H(p_1,\ldots,p_n) \leq H(1/n,\ldots,1/n) = \lg n$
- Null set property. Menambahkan event dg prob 0 pada set event tidak mengubah entropy
  $H(p_1, \ldots, p_n, 0) = H(p_1, \ldots, p_n)$
- Fungsi $f(n) = H(1/n, \ldots, 1/n)$ tumbuh monoticall:
  $f(n) < f(n+i)$ utk $i > 0$ dan $n > 0$

Karakteristik Entropy

- Grouping axiom. Jika set $S = \{x_1, \ldots, x_n\}$, nilai $x_1, \ldots, x_i$ disatukan bersama membentuk satu grup S_i:

$$H(p_1,\ldots,p_i.p_{i+1},\ldots,p_n) =$$
$$H(p_1+\ldots+p_i, p_{i+1},\ldots,p_n) +$$
$$(p_1+\ldots+p_i)H\left(\frac{p_1}{p_1+\ldots+p_i},\ldots,\frac{p_i}{p_1+\ldots+p_i}\right)$$

Noiseless & Memoryless Coding

- Sumber tidak punya memory (simbol ditransmisikan secara independen)
- $S = \{x_1, \ldots, x_n\}$, $P = \{p_1, \ldots, p_n\}$
- Codewords $C = \{c_1, \ldots, c_n\}$
- Code disebut binary code jika komposisi codeword adalah 0 dan 1
- Rata-rata panjang codeword

$$L_{avg} = \sum_{i=1}^{n} p_i l_i$$

dimana l_i adalah panjang codeword c_i yang mengkodekan simbol x_i

- Panjang codeword Shannon : $l_i = \lceil -\lg p_i \rceil$
- *Dalam kompresi data → meminimumkan cost (L_{avg})*

Noiseless & Memoryless Coding Definisi

- Suatu code adalah uniquely decodable jika hanya ada satu cara memecah deretan codeword $c_{i1}c_{i2}...c_{ik}$ ke dalam codeword terpisah. Yaitu jika $c_{i1}c_{i2}\ldots c_{ik} = c_{j1}c_{j2}\ldots c_{jk}$, maka untuk tiap s,
  $i_s = j_s$ (yaitu $c_{is} = c_{js}$)
- Suatu code mempunyai prefix (atau irreducibility atau self-punctuating) property jika tidak ada code word didapat dari codeword lain dengan menambahkan 0 atau 1, atau tidak ada codeword merupakan prefix dari codeword lain
  - scanning deretan codeword, tidak memerlukan melihat kedepan (*look ahead*) untuk menghindari ambiguitas
  - Tidak diperlukan tanda khusus untuk memisahkan dua codeword dalam message
- Optimal code adalah yang menghasilkan harga L_{avg} terendah yang mungkin

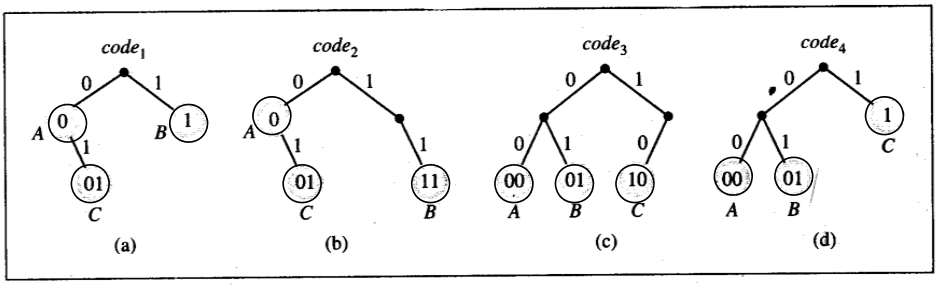
Contoh

Huruf	code1	code2	code3	code4
A	0	0	00	00
B	1	11	01	01
C	01	01	10	1

Mana yang mempunyai karakteristik

- *uniquely decodable*
- *prefix property*
- *optimal code*

Contoh



The Kraft Inequality

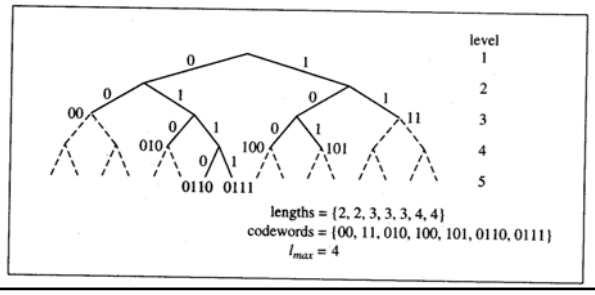
Kraft's Theorem

- Terdapat prefix binary code dengan codeword {c_1, .., c_n} dengan panjang {l_1, ….., l_n} jika dan hanya jika

$$\sum_{i=1}^{n} 2^{-l_i} \leq 1$$

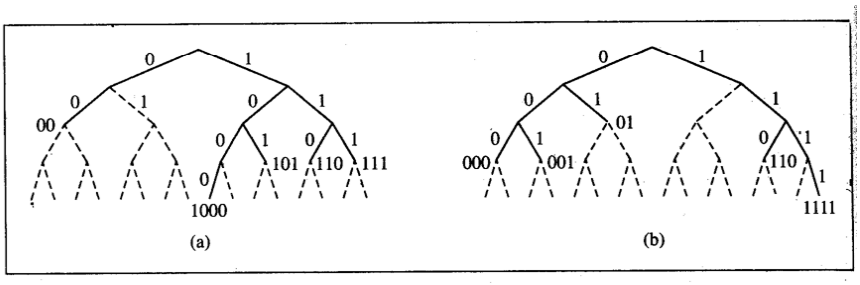
- Bukti

$\sum_{i=1}^{n} 2^{l_{\max}-l_i} \leq 2^{l_{\max}}$ *atau* $\sum_{i=1}^{n} 2^{-l_i} \leq 1$



The Kraft Inequality

- Theorem menyatakan untuk satu set panjang codeword, prefix code dapat dibentuk dan bukan bagaimana membentuknya
- Memungkinkan banyak prefix code dapat dibuat dengan tetap memenuhi kondisi teorema
- Teorema menjamin mendapatkan prefix code tapi tidak menjamin optimal



The Kraft Inequality

- The Kraft inequality menjadi equality jika codeword tidak dp diperpendek lagi, perhatikan utk set panjang codeword {2,3,3,4} dan {1,3,3,3)

$$\frac{1}{2^2}+\frac{1}{2^3}+\frac{1}{2^3}+\frac{1}{2^3}+\frac{1}{2^4}=\frac{11}{16}<1$$

$$\frac{1}{2^1}+\frac{1}{2^3}+\frac{1}{2^3}+\frac{1}{2^3}+\frac{1}{2^3}=\frac{8}{8}=1$$

- Utk sejumlah panjang codeword prefix code dp dicari, tetapi ada kemungkinan utk set panjang yg sama, dp dibangun non-prefix code
- Teorema hanya berbicara prefix code, tetapi prefix code → uniquely decodable code

Teorema Fundamental Discrete Coding

- Ukuran kinerja (dari sudut pandang kompresi data):

$$\textit{Compression Ratio} = \frac{\textit{Panjang (output)}}{\textit{Panjang (input)}}\ \textit{x 100\%}$$

$$\textit{Compression Rate} = \textit{1 - Compression Ratio}$$

- Mis. Kompresi file dg compression ratio 75% berarti file hasil kompresi ¾ file sblm kompresi
- Compression rate 25% berarti file sblm kompresi dikurangi ¼ nya (persentasi space yg dp dihemat)

Teorema Fundamental Discrete Coding

- Untuk suatu rasio kompresi yang didapat, bisakah ditingkatkan lagi?
- Konsep Entropy menunjukan batas kompresi yang dapat dicapai
- Panjang codeword rata-rata > source entropy

Teorema Fundamental Discrete Coding

- Teorema
  - Untuk suatu prefix binary code dengan panjang rata-rata codeword $L_{avg} = \sum p_i l_i$ maka

    $L_{avg} \geq H(S)$
  - Terdapat prefix binary code dimana

    $L_{avg} < H(S) + 1$
- Shannon's Fundamental Theorem of Discrete Noiseless Coding
  - Untuk sumber *S* dengan entropy *H(S),* dimungkinkan mengalokasikan codeword deretan *k* simbol dengan kondisi prefix dipenuhi, dan panjang rata-rata codeword L_k

$$H(S) \leq \frac{L_k}{k} < H(S) + \frac{1}{k}$$

Teorema Fundamental Discrete Coding

- Contoh
  $S = \{a,b,c\}$, $P = \{0.1, 0.45, 0.45\}$
  $H(S) = 1,369$
  Panjang codeword: $p = 0,1 \rightarrow l = \lceil -\lg 0,1 \rceil = 4$
  $p = 0,45 \rightarrow l = \lceil -\lg 0,45 \rceil = 2$
  $L_{avg} = 2,2$ bit/karakter

  Ambil set panjang codeword = $\{2,2,1\}$ → memenuhi Kraft inequality
  $L_{avg} = 1,55$ bit/karakter

- $1,55$ bit/kar lebih baik drpd $2,2$ bit/kar → masih ada ruang perbaikan (ingat entropy sistem = $1,369$)

Teorema Fundamental Discrete Coding

- L_{avg} dp diperbaiki dg mengambil blok/deretan karakter drpd single karakter (dg bayaran kompleksitas yg meningkat)
- Contoh: $S = \{a,b,c\}$, $P = \{0.1, 0.45, 0.45\}$
  Bentuk sumber baru $S_2 = \{aa,ab,ac,ba,bb,bc,ca,cb,cc\}$
  → $P = \{0.01, 0.045, 0.045, 0.045, 0.2025, 0.2025, 0.045, 0.2025, 0.2025\}$
  $H(S_2) = 2H(S) = 2,738$ → *buktikan!*
  Panjang codeword (Shannon)
  $\lceil -lg\ 0,01 \rceil = 7$; $\lceil -lg\ 0,45 \rceil = 5$; $\lceil -lg\ 0,2025 \rceil = 3$
  Panjang rata-rata per sub-simbol:
  $L_2 = 0,01 \cdot 7 + 4 \cdot 0,045 \cdot 5 + 4 \cdot 0.2025 \cdot 3 = 3,4$
  ∴ Panjang rata-rata per karakter $= L_2/2 = 1,7$ bit/karakter

Shannon-Fano Coding

Suboptimal code
- Shannon code
- Shannon-Fano code

Optimal code
- Huffman code
- Arithmetic coding

Efisiensi macam-macam code diukur dengan:

$$effisiensi = \frac{H(S)}{L_{avg}}.100\%$$

Shannon Coding

- $S = \{x_1, \ldots, x_n\}$
- $P = \{p_1, \ldots, p_n\}$
- $p_i = p(x_i)$ dari semua simbol sumber x_i diurut dari yang paling besar: $p_1 \geq p_2 \geq \ldots \geq p_n$
- Cumulative prob didefinisikan: $P_i = p_1 + \ldots + p_{i-1}$
- Codeword utk simbol x_i didp dg mengambil $l_i = |-\lg p_i|$ digit pertama dari ekspansi biner P_i

  $P_i = 0.b_1b_2b_3b_4 \ldots = b_1/2^1 + b_2/2^2 + b_3/2^3 + \ldots$

Shannon Coding

- Contoh:

  $S = \{A, B, C, D, E\}$

  $P = \{0.35, 0.17, 0.17, 0.16, 0.15\}$

x_i	p_i	l_i		P_i	codeword
A	.35	2	.0	.0000000 ...	00
B	.17	3	.35	.0101100 ...	010
C	.17	3	.52	.1000010 ...	100
D	.16	3	.69	.1011000 ...	101
E	.15	3	.85	.1101100 ...	110

Shannon-Fano Coding

order the source letters into a sequence s *according to the probability of occurrence;*
ShannonFano (*sequence* s)
    if s *has two letters*
        attach 0 *to the codeword of one letter and* 1 *to the codeword of another;*
    else if s *has more than one letter*
        divide s *into two subsequences* s_1 *and* s_2, *with the minimal difference between probabilities of each subsequence;*
        extend the codeword for each letter in s_1 *by attaching* 0, *and attaching* 1 *to each codeword for letters in* s_2;
        ShannonFano(s_1);
        ShannonFano(s_2);

Shannon-Fano Coding

- Contoh

  $S = \{A, B, C, D, E\}$

  $P = \{0.35, 0.17, 0.17, 0.16, 0.15\}$

- Pengkodean Shannon-Fano:
  - Bagi S kedalam s_1 dan s_2 (pilih yang memberikan perbedaan $p(s_1)$ dan $p(s_2)$ terkecil
  - s_1 = (A,B) → $p(s_1) = p(A) + p(B)$ = 0,52
  - s_2 = (C,D,E) → $p(s_2) = p(C) + p(D) + p(E)$ = 0,48
  - Panggil ShannonFano()

Shannon-Fano Coding

x_i	p_i	codeword
A	.35	00
B	.17	01
C	.17	10
D	.16	110
E	.15	111

- Panjang code rata-rata:

  L_{sh} = 0,35*2 + 0,17*2 + 0,17*2 + 0,16*3+0,15*3 = 2,31
- Efisiensi = (2,23284/2,31)*100 = 96,66 %

Kompresi Text

- Shannon-Fano coding salah satu yg digunakan utk kompresi text

Scheme	Description
Shannon-Fano Coding	Uses variable length code words i.e symbols with higher probability of occurrence are represented by smaller codes-words.
Huffman Coding	Same as Shannon-Fano Coding
LZW	Replaces strings of characters with single codes. It does not do any analysis of the incoming text. Instead it just adds every new string of characters it sees to a table of strings. Compression occurs when a single code is output instead of string of characters.
Unix Compress	Use LZW with growing dictionary. Initially 512 entries, and is subsequently doubled till it reaches the maximum value set by the user.

PR (Tugas-1):

kumpulkan minggu depan (waktu kuliah)

1. Untuk sumber alphabet S={a,b,c} dan probabilitas P={0.1, 0.2, 0.7}:
   a. cari panjang codeword Shannon rata-rata
   b. Spt soal (a) utk semua pasangan yg mungkin dari 3 huruf di atas.
2. Suatu sumber S={a,b,c,d,e,f} dg probabilitas P={0.15, 0,16, 0,13, 0,20, 0,25, 0,11}. Cari codeword utk masing-masing simbol dg metoda Shannon.
3. Dengan menggunakan metoda Shannon-Fano Coding, tentukan codeword utk suatu sumber dg probabilitas berikut
   a. (1/2, 1/4, 1/8, 1/16, 1/16)
   b. (0.4, 0.3, 0.2, 0.1)