Error Correction & Detection dengan Hamming Code

Andhika Tantyo Anugrah - 13522094[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13522094@mahasiswa.itb.ac.id

***Abstrak*—Kode koreksi kesalahan adalah teknik enkode yang digunakan untuk mengendalikan kesalahan dalam transmisi data agar data dapat dikoreksi jika keadaan beberapa bit dari data tersebut berubah secara tidak sengaja. Koreksi ini dilakukan saat penerima data melakukan dekode pada data yang telah diterima. Kode koreksi kesalahan digunakan dalam berbagai kasus transmisi pesan melalui kanal komunikasi yang tidak bisa diandalkan atau kotor karena interferensi. Makalah ini akan menggunakan kode Hamming dan format data yang berbentuk matriks dengan tujuan untuk membahas kode koreksi kesalahan.**

***Keywords*—Enkode, dekode, kode Hamming, kode koreksi kesalahan, transmisi data.**

I. PENDAHULUAN

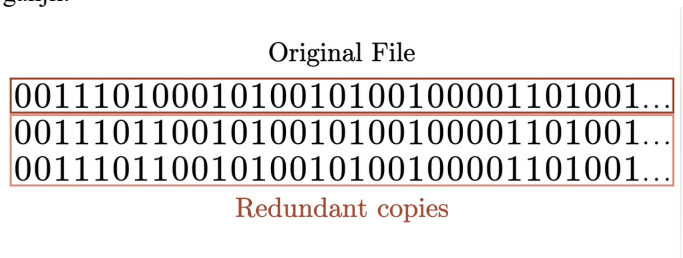
Kode koreksi kesalahan adalah sebuah teknik enkode yang sangat penting di zaman informasi sekarang ini. Tanpa kode koreksi kesalahan, integritas data yang disimpan sementara, disimpan jangka panjang, ataupun ditransmisi menjadi rendah karena adanya gangguan dari partikel berenergi seperti partikel alpha dan partikel neutron. Kode koreksi kesalahan sering diimplementasikan ke *memory* atau biasa disebut *Random Access Memory* (RAM) untuk kasus-kasus yang tidak bisa menoleransi kerusakan data seperti komputer di pesawat. [1]

Kebutuhan akan kode koreksi keesalahan menjadi sebuah keharusan karena situasi-situasi ekstrem yang dulunya jarang terjadi, lama-kelamaan menjadi sebuah kebiasaan yang selalu terjadi. [2] Beberapa dari situasi-situasi tersebut:

1. operasi yang tidak diawasi dengan jangka waktu yang panjang dengan beban kerja yang konstan.
2. kompleksitas sistem yang semakin kompleks menyebabkan sebuah kegagalan dalam sistem bisa menimbulkan *avalance effect* seperti A Byzantine failure in the real world (cloudflare.com).
3. interferensi sinyal yang tidak bisa atau tidak ekonomis untuk mengurangi efeknya pada sinyal.

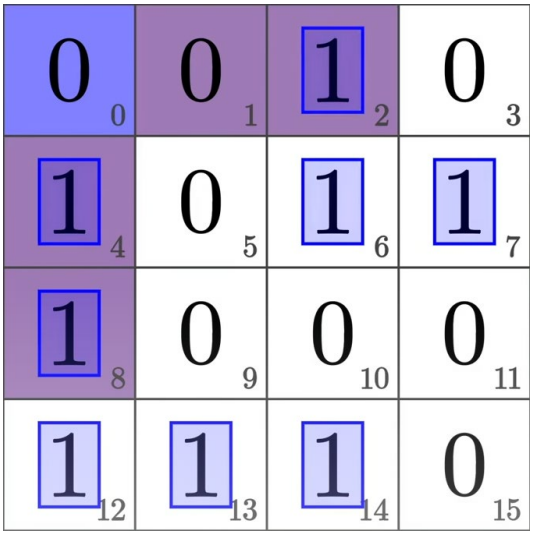
Kita asumsikan alat transmisi menerima informasi dalam bentuk sekuens binary dari 0 dan 1. Asumsi ini dibuat karena bentuk alami untuk merepresentasikan sirkuit flip-flop dan data digital adalah sistem angka binary. Oleh karena itu, setiap kode data akan direpresentasikan oleh sebuah sekuens 0 dan 1. [2]

Secara sederhana, mendeteksi dan mengkoreksi kesalahan dalam transmisi data dapat dilakukan dengan strategi mengirimkan salinan data yang ingin dikirim sebanyak tiga kali. Namun, cara ini tidaklah efisien, karena kapasistas yang digunakan untuk menyimpan salinan data itu menjadi tiga kali lipat. Pendekatan yang lebih baik bisa dilakukan dengan pengecekan paritas. Dengan menetapkan suatu bit, bernilai 1 jika jumlah angka 1 pada sekuens data bernilai genap, dan 0 jika ganjil.



Gambar 1.1 Ilustrasi deteksi dan koreksi sebuah file menggunakan strategi mengirim 3 buah salinan sebuah file. [3]

Kode Hamming versi *Single Error Correction, Double Error Detection* (SECDED) memanfaatkan pengecekan paritas ini untuk mendeteksi dan mengkoreksi kesalahan 1-bit dan mendeteksi kesalahan 2-bit.



Gambar 1.2 Contoh matriks dari suatu sekuens data, dengan sel ke-0, 1, 2, 4, dan 8 sebagai penanda paritas yang dihasilkan dari kode Hamming. [3]

II. LANDASAN TEORI

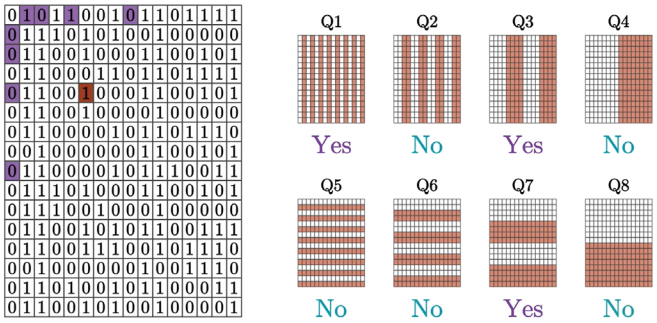
Kode Hamming adalah sebuah kelompok kode koreksi kesalahan linear. Kode Hamming ditemukan oleh Richard Wesley Hamming pada 1950 sebagai cara untuk memperbaiki

kesalahan yang dihasilkan oleh pembaca *punch cards*. [2] Dalam koreksi kesalahan 1-bit dan deteksi kesalahan 2-bit, dimanfaatkan aljabar Boolean untuk membentuk sebuah matriks yang berisi bit paritas dan bit data.

Bit paritas ini diletakkan di tempat-tempat yang strategis, seperti bit ke-0, ke-2, ke-4, ke-8, sampai seterusnya sesuai dengan aturan 2^{n-1}, dengan n adalah ukuran matriks bujur sangkar. Pemilihan matriks bujur sangkar di sini bertujuan untuk memudahkan perhitungan. Lalu pada sel matriks yang memuat nilai 1, indeksnya dikumpulkan dan dilakukan operasi XOR pada semua indeks tersebut, dan dihasilkanlah indeks yang memiliki kesalahan.

Kode Hamming ini efisien, karena hanya menggunakan n/2 + 1 bit atau sel matriks dari ukuran matriks bujur sangkar dengan besaran n. Hal ini terjadi, karena kode Hamming ini menggunakan indeks yang berbentuk binary sebagai pembeda antara kolom atau baris tertentu.

$$256 = 2^8 \text{ bits}$$



Gambar 2.1 Contoh dari matriks yang memiliki kesalahan di baris ke-5, kolom ke-6 [3]

Aljabar Boolean adalah sebuah sebuah cara formal untuk mendeskripsikan operasi logika. Aljabar Boolean dikenalkan oleh George Boole Jnr dalam bukunya yang berjudul *The Mathematical Analysis of Logic* (1847) dan dikemukakan lebih lengkap dalam *An Investigation of the Laws of Thought* (1854). [4][5]

Aljabar Boolean dapat dimisalkan dengan tupel <**B**, +, ., ‘, **0**, **1**>, dengan B sebagai himpunan yang didefinisikan pada dua operan biner, + dan ., dan sebuah operan uner, ‘. Oleh karena itu, untuk setiap a, b, c ∈ **B**, berlaku aksioma:

1. Identitas
   (i) a + 0 = a
   (ii) a . 1 = 1

2. Komutatif
   (i) a + b = b + a
   (ii) a . b = b . a

3. Distributif
   (i) a . (b + c) = (a . b) + (a . c)
   (ii) a + (b . c) = (a + b) . (a + c)

4. Komplemen
   Untuk setiap a ∈ **B** terdapat elemen unik a‘∈ **B** sehingga
   (i) a + a’ = 1
   (ii) a . a’ = 0

Aljabar himpunan dan aljabar proposisi adalah himpunan bagian (*subset*) dari aljabar Boolean karena memenuhi keempat aksioma di atas. Aljabar Boolean 2-nilai merupakan aljabar Boolean yang paling populer karena aplikasinya yang luas. Pada aljabar Boolean 2-nilai **B** adalah {**0**, **1**}, operan biner, + dan **.**, dan sebuah operan uner, ‘ dengan hasil operasi:

a	*b*	*a* · *b*
0	0	0
0	1	0
1	0	0
1	1	1

a	*b*	*a* + *b*
0	0	0
0	1	1
1	0	1
1	1	1

a	*a*’
0	1
1	0

Gambar 2.2 hasil dari operasi biner dan uner dari 2 nilai **B**. [6]

Hukum-hukum aljabar Boolean meliputi:

1. Hukum Identitas
(i) a + 0 = a
(ii) a . 1 = a

2. Hukum Idempoten
(i) a + a = a
(ii) a . a = 0

3. Hukum Komplemen
(i) a + a’ = 1
(ii) aa’ = 0

4. Hukum Dominansi
(i) a . 0 = 0
(ii) a + 1 = 1

5. Hukum Involusi
(i) (a’)’ = a

6. Hukum Penyerapan
(i) a + ab = a
(ii) a(a + b) = a

7. Hukum Komutatif
(i) a + b = b + a
(ii) ab = ba

8. Hukum Asosiatif
(i) a + (b + c) = (a + b) + c
(ii) a(bc) = (ab)c

9. Hukum Distributif
(i) a + (bc) = (a + b) (a + c)
(ii) a(b + c) = ab + ac

10. Hukum De Morgan
(i) (a + b)’ = a’b’
(ii) (ab)’ = a’ + b’

11. Hukum 0/1
(i) 0’ = 1
(ii) 1’ = 0

Fungsi boolean adalah fungsi yang dibentuk dari literal

boolean, contohnya adalah:

$$f(x, y) = x'y + xy' + y'$$

Fungsi boolean dapat dituliskan dalam bentuk kanonik yang terdiri dari dua jenis, penjumlahan dari fungsi kali dan perkalian dari hasil jumlah. Bentuk kanonik adalah ekspresi Boolean yang dinyatakan sebagai penjumlahan dari satu atau lebih *minterm* atau perkalian dari satu atau lebih *maxterm*:

1. Penjumlahan dari Fungsi Kali (*Sum of Product* (SOP))
   Cara membentuk *minterm* adalah setiap peubah yang bernilai 0 dinyatakan dalam bentuk komplemen, sedangkan peubah yang bernilai 1 dinyatakan tanpa komplemen. Untuk membentuk SOP, dapat mengambil *minterm* dari setiap nilai fungsi yang bernilai 1 pada tabel kebenaran. Contoh:

   $$f(x, y, z) = x'y'z + xy'z' + xyz$$

2. Perkalian dari Hasil Jumlah (*Product of Sum* (POS))
   Cara membentuk *maxterm* adalah setiap peubah yang bernilai 0 dinyatakan tanpa komplemen, sedangkan peubah yang bernilai 1 dinyatakan dalam bentuk komplemen. Untuk membentuk POS, kita dapat mengambil *maxterm* dari setiap nilai fungsi yang bernilai 0 pada tabel kebenaran. Contoh:

   $$g(x, y, z) = (x + y + z)(x + y' + z)(x + y' + z')(x' + y + z')(x' + y' + z)$$

		Minterm		*Maxterm*	
x	*y*	Suku	Lambang	Suku	Lambang
0	0	$x'y'$	m_0	$x + y$	M_0
0	1	$x'y$	m_1	$x + y'$	M_1
1	0	xy'	m_2	$x' + y$	M_2
1	1	$x\,y$	m_3	$x' + y'$	M_3

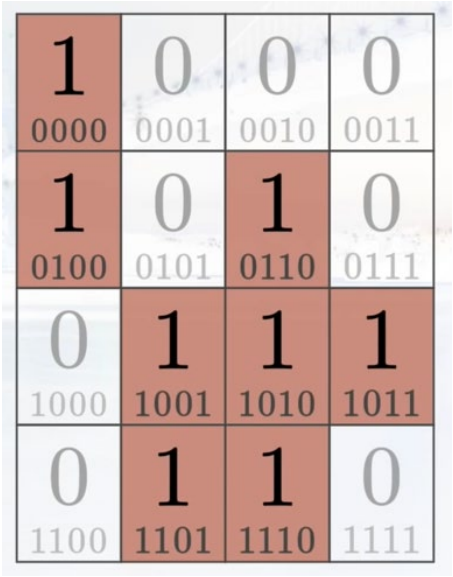
Gambar 2.3 Tabel *minterm* dan *maxterm* untuk dua peubah [6]

III. Implementasi Program

Penulis menggunakan bahasa python untuk membuat sebuah *array* dengan panjang 16 dengan elemen [1,0,0,0,1,0,1,0,0,1,1,1,0,1,1,0] untuk mensimulasikan sebuah data yang ingin ditransmisikan.

```
>>> import numpy as np
>>> array = np.array([1,0,0,0,1,0,1,0,0,1,1,1,0,1,1,0])
>>> list(enumerate(array))
[(0, 1), (1, 0), (2, 0), (3, 0), (4, 1), (5, 0), (6, 1), (7, 0), (8, 0), (9, 1), (10, 1), (11, 1), (12, 0), (13, 1), (14
, 1), (15, 0)]
>>> [i for i, bit in enumerate(array)]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
>>> [i for i, bit in enumerate(array) if bit]
[0, 4, 6, 9, 10, 11, 13, 14]
>>> 0 ^ 4 ^ 6 ^ 9 ^ 10 ^ 11 ^ 13 ^ 14
9
>>>
```

Gambar 3.1 Array yang errornya diketahui



Gambar 3.2 Matriks dengan elemen sesuai dengan array di atas dan dengan indeks biner

Buatlah index dari tiap elemen pada matriks dalam biner, ini dilakukan untuk mempermudah perhitungan lokasi nanti. Cara termudah untuk mendapatkan lokasi data bit yang tidak sesuai adalah dengan cara melakukan operasi XOR pada setiap indeks sel yang memiliki angka 1. Setelah itu, didapatkanlah lokasi bit yang memiliki kesalahan, yaitu pada bit ke 9.

V. Kesimpulan

Telah diimplementasikan kode Hamming untuk mengkoreksi dan mendeteksi *Single Error Correction, Double Error Detection* (SECDED). Aljabar Boolean dapat digunakan untuk menemukan lokasi dari kesalahan dengan cepat menggunakan operan XOR.

Hasil yang diperoleh melalui penelitian ini menunjukkan kegunaan aljabar Boolean yang bisa diterapkan pada sistem yang memerlukan sebuah metode untuk memeriksa kesalahan pada transmisi data. Aljabar Boolean sangat berguna dalam kehidupan kita, seperti implementasinya pada kode Hamming ini.

VI. Lampiran

Lampiran Kode beserta outputnya

```
>>> import numpy as np
>>> array = np.array([1,0,0,0,1,0,1,0,0,1,1,1,0,1,1,0])
>>> [i for i, bit in enumerate(array)]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
>>> [i for i, bit in enumerate(array) if bit]
[0, 4, 6, 9, 10, 11, 13, 14]
>>> 0^4^6^9^10^11^13^14
9
```

VII. Ucapan Terima Kasih

Puji syukur kita panjatkan kepada Tuhan Yang Maha Esa karena makalah sebagai tugas akhir mata kuliah IF2120 Matematika Diskrit berjudul “Error Correction & Detection dengan Hamming Code” bisa diselesaikan tepat waktu. Makalah

ini tidak mungkin terselesaikan dengan baik tanpa adanya dukungan dari berbagai pihak. Maka di kesempatan ini, izinkan kami mengucapkan banyak terima kasih kepada

1) Dr. Fariska Zakhralativa Ruskanda sebagai dosen pengajar mata kuliah IF2120 Matematika Diskrit yang sudah mengajar dengan baik.
2) Dr. Rinaldi Munir sebagai pembuat materi yang banyak membantu penulis dalam menyelesaikan makalah ini.

Penulis harap makalah yang sudah disusun bisa bermanfaat bagi banyak orang.

REFERENCES

[1] E. Normand, "Single event upset at ground level," in IEEE Transactions on Nuclear Science, vol. 43, no. 6, pp. 2742-2750, Dec. 1996, doi: 10.1109/23.556861.

[2] R. W. Hamming, "Error detecting and error correcting codes," in The Bell System Technical Journal, vol. 29, no. 2, pp. 147-160, April 1950, doi: 10.1002/j.1538-7305.1950.tb00463.x.

[3] But what are Hamming codes? The origin of error correction (youtube.com), diakses pada 04 Desember 2023 pukul 21.05 UTC+0700

[4] Who is George Boole: the mathematician behind the Google doodle (smh.com.au), diakses pada 11 Desember 2023 pukul 21.39 UTC+0700.

[5] The Bicentennial of George Boole, the Man Who Laid the Foundations of the Digital Age - Scientific American Blog Network, diakses pada 11 Desember 2023 pukul 23.42 UTC+0700

[6] https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)-bagian1.pdf, diakses pada 11 Desember 2023 pukul 21.53 UTC+0700

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 10 Desember 2023

Andhika Tantyo Anugrah 13522094