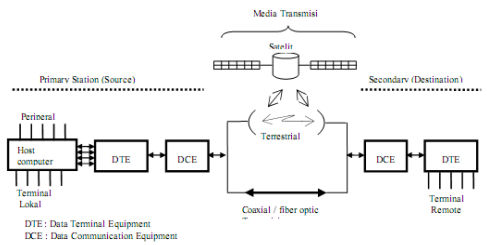
APLIKASI SIMULASI DETEKSI DAN KOREKSI KESALAHAN MENGGUNAKAN METODE *HAMMING CODE* PADA PROSES PEGIRIMAN DATA BERBASIS ANDROID

Helfy Susilawati[1], Dede Sunardi[2]
Prodi Teknik Elektro, Prodi D3 Teknik Telekomunikasi
Universitas Garut

Abstrak
Eclipse merupakan salah satu software untuk membuat aflikasi dengan basis android. Semakin maraknya penggunaan *handphone* berbasis android membuat aplikasi dengan basis ini akan lebih mudah untuk dikembangkan dan juga mudah untuk digunakan. Metode deteksi dan koreksi kesalahan *Hamming code* merupakah salah satu metode koreksi kesalahan dalam proses pengiriman data. Metode ini disimulasikan dengan cara membuat aflikasi yang bias secara langsung melakukan deteksi dan melakukan koreksi data yang dimasukkan. Perancangan aplikasi simulasi deteksi dan koreksi kesalahan dengan menggunakan metode *Hamming code* ini bias menjadi salah satu percobaan semu dalam proses komunikasi data.
Kata kunci : Simulasi, Eclipse, *Hamming code*.

Pendahuluan

Proses pengiriman data merupakan proses yang sangat penting pada system komunikasi. Salah satu ukuran keberhasilan dalam system komunikasi adalah tidak terjadinya kesalahan dalam proses pengiriman data dari pengirim ke penerima. Adapun jika terdapat keslahan dalam proses komunikasi (khusunya pada proses pengiriman data) antara pengirim dan penerima, diperlukan suatu cara untuk memperbaiki kesalahan pengiriman data tersebut. Prosesd pada jaringan komunikasi dapat dilihat pada gambar dibawah ini.



Gambar 1 Blok diagram jaringan komunikasi data sederhana

Kesalahan dalam proses pengiriman data dapat terjadi karena media transmisi yang digunakan, sinyal informasi pada proses pengiriman, dan adanya gangguan/noise pada saat proses pengiriman dta. Salah satu cara untuk memperbaiki hal tersebut adalah dengan menggunkan metode yang bias digunakan untuk melakukan deteksi dan koreksi kesalahan oleh penerima dan pengirim yakni metode *Hamming Code*. *Metode Hamming code* Merupkan salah satu metode koreksi kesalahan dalam proses pengecekan pengiriman data. Metode Hamming Code adalah metode metode yang diciptakan Richard Haming di Bell Lab pada tahun 1950. Pada proses perancangan simulasi ini penulis menggunakan metode Hamming Code dikarenkan metode ini memiliki mekanisme koreksi kesalahn yang paling sederhana. Pada proses pembuatan aplikasi ini, penulis juga menggunakan software eclipse sebagai alat untuk membuat aplikasi. Software eclipse merupakan software yang sering digunakan oleh para developer untuk membuat aplikasi dengan basis android.

Data yang dikirim oleh pengirim ke penerima akan melalui beberapa tahap proses salah satunya adalah proses koreksi kesalahan data. Salah satu metode dalam koreksi kesalahan adalah metode deteksi dan koreksi kesalahan dengan Hamming Code. Metode Haming Code dapat dapat digunakan untuk membuat sandi baru dan juga untuk mengkoreksi data yang dikirim. Dalam perancangan aplikasi ini, penulis membuat aplikasi yang dapat membuat sandi baru dan juga untuk koreksi data.

Ladasan Teori

Algoritma Hamming Code dikemukakan oleh Richard W. Hamming pada tahun 1940-an. Metode koreksi kesalahan dengan menggunakan Hamming Code merupakan suatu metode pendeteksian kesalahan dengan menambahkan data dengan suatu kode, biasanya disebut paritas. Algoritma Hamming Code merupakan salah satu algoritma pendeteksi error (error detection) yang mampu untuk mendeteksi beberapa error, namun hanya mampu mengoreksi satu error (single error correction).[3] Data yang dikirimkan berupa data asli dan data yang mengandung parity.

Jumlah parity tergantung dari panjangnya data yang akan dikirimkan. Jika suatu data memiliki nilai dari perpangkatan 2 (misal 2^y), maka untuk menentukan jumlah parity nya dapat dilakukan dengan cara:

parity = (y +1)

Jika data yang dimiliki adalah 4, maka jumlah parity ada 3, dan jika data ada 16, maka jumlah parity ada 5. Dengan kata lain, jika pengirim mengirimkan 4 bit, maka data yang akan dikirim menjadi 7 bit, dengan 3 bit merupakan bit parity, begitu juga jika pengirim akan mengirimkan data sebanyak 16 bit, maka banyak data yang dikirim sebenarnya adalah 21 dengan 5 bit tambahan merupakan bit parity. Untuk menentukan letak parity, maka dapat ditemukan dengan menggunakan menggunakan persamaan

$P_i = 2^{i-1}$

Sehingga posisi parity dapat dilihat pada tabel dibawah.

Tabel 1 Posisi *Parity*

Parity	Posisi
P_1	1
P_2	2
P_3	4
P_4	8
P_5	16
P_6	32

Proses deteksi kesalahan dengan menggunakan *Hamming Code*, yaitu:

a. Menentukan jumlah data yang akan dideteksi/dikirim
b. Menentukan letak parity (di posisi 1,2,4, dst)
c. Menentukan nilai dari setiap parity
d. Menggabungkan nilai bit parity dengan bit data yang ada

Proses koreksi error dengan menggunakan *Hamming Code*, yaitu:

1. Hitung jumlah bit data yang dikirimkan
2. Tandai posisi setiap parity pada data bit tersebut
3. Lakukan operasi *XOR* pada setiap parity
4. Jika nilai output dari parity tidak sama dengan input, maka terjadi error
5. Mengkonversikan hasil output parity ke bilangan desimal sehingga akan diketemukan letak data yang salah

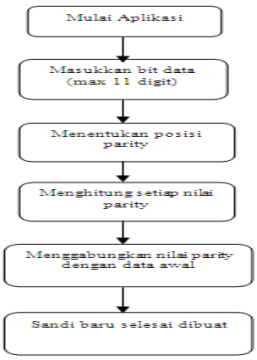
Perancangan Simulasi

Perancangan aplikasi simulasi deteksi dan koreksi menggunakan metode *Hamming Code* terbagi dalam dua kategori. Kategori pertama adalah pembuatan sandi baru pada saat data

akan dikirimkan. Pembuatan sandi baru dapat dilakukan dengan beberapa tahapan, yaitu:

a. Memasukkan bit data maksimal 11 digit
b. Menentukan posisi parity 1, parity 2, parity 3, dan parity 4 pada data yang telah dimasukkan
c. Menentukan nilai dari setiap parity
d. Menggabungkan data parity dengan data awal
e. Pembuatan sandi baru selesai

Untuk lebih jelasnya, perancangan simulasi untuk pembuatan sandi baru dapat dilihat pada diagram dibawah.



Gambar 2. Diagram pembuatan sandi baru

Untuk proses deteksi dan koreksi kesalahan dengan menggunakan *Hamming Code* dilakukan dengan beberapa tahapan, yaitu:

a. Memasukkan bit data maksimal 15 digit
b. Menentukan posisi parity 1, parity 2, parity 3, dan parity 4 pada data yang telah dimasukkan
c. Menentukan nilai dari setiap parity
d. Menentukan data yang benar dan yang salah dari data yang telah diinputkan
e. Menentukan letak data yang salah
f. Melakukan koreksi data berdasarkan data yang diinputkan

Untuk lebih jelasnya, perancangan simulasi untuk deteksi dan koreksi kesalahan dapat dilihat pada diagram dibawah.

Berikut adalah potongan code untuk proses pembuatan sandi baru dan proses deteksi dan koreksi kesalahan menggunakan *Hamming Code*.

```
public class Cek extends ActionBarActivity {

    Button button;
    EditText data1;
    EditText data2;

    TextView txt;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.cek_satu);

        final EditText ed=(EditText)findViewById(R.id.ed);
        final EditText ed1=(EditText)findViewById(R.id.ed2);
        final EditText ed2=(EditText)findViewById(R.id.ed3);
        final EditText ed3=(EditText)findViewById(R.id.ed4);
        final EditText ed4=(EditText)findViewById(R.id.ed5);
        final EditText ed5=(EditText)findViewById(R.id.ed6);
        final EditText ed6=(EditText)findViewById(R.id.ed7);
        final EditText ed7=(EditText)findViewById(R.id.ed8);
        final EditText ed8=(EditText)findViewById(R.id.ed9);
        final EditText ed9=(EditText)findViewById(R.id.ed10);
        final EditText ed10=(EditText)findViewById(R.id.ed11);
```

Gambar 3 Potongan source code untuk membuat sandi baru

Hasil Dan Pembahasan

Pembuatan sandi baru

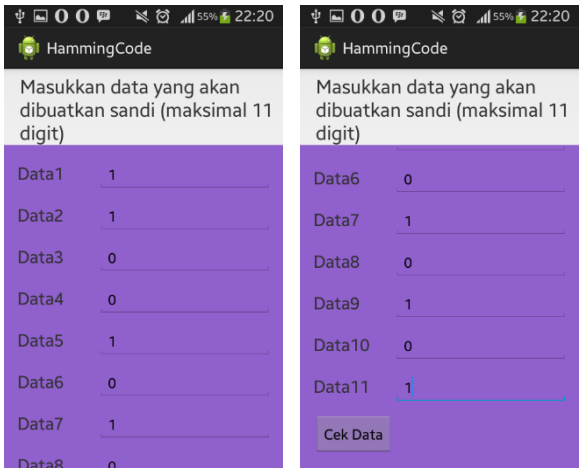
- Masuk ke aplikasi

Gambar 4 Halaman awal aplikasi

- Pilih menu Buat Sandi Baru

Gambar 5
Halaman pemilihan menu

- Masukkan maksimal 11 data bit untuk pembuatan sandi baru



Gambar 6 Halaman pembuatan sandi baru dengan memasukkan data maksimal 11 bit

- Penentuan posisi parity sebagai data tambahan

Gambar 7 Penentuan parity

- Penentuan sandi baru untuk menjadi data

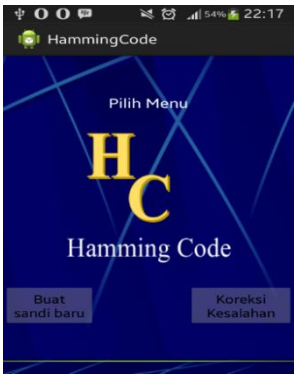
Gambar 8 Pembuatan sandi baru

Deteksi dan koreksi kesalahan

1. Masuk ke aplikasi

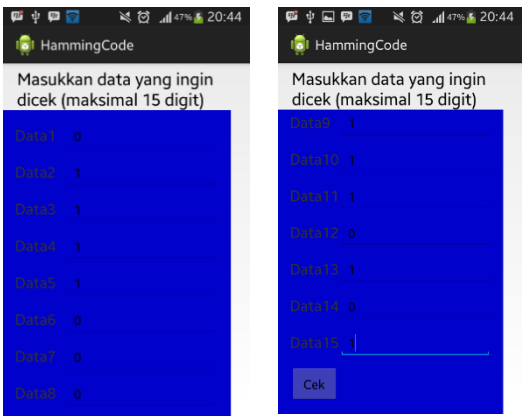
Gambar 9 Halaman awal aplikasi

2. Pilih menu Koreksi Kesalahan



Gambar 10 Halaman pemilihan menu

3. Masukkan maksimal 15 data bit untuk dicek data



Gambar 11 Halaman koreksi kesalahan dengan memasukkan maksimal 15 bit data

4. Penentuan parity pada setiap data yang telah dimasukkan

Gambar 12 Halaman penentuan *parity* dari data awal

5. Penentuan jumlah *parity* dan posisi kesalahan pada data

Gambar 13 Halaman hasil deteksi dan koreksi kesalahan

Kesimpulan

1. Data yang dikirim oleh pengirim akan berbeda dengan data yang diterima oleh penerima dikarenakan pada penerima sudah mendapatkan sandi baru.
2. Pembuatan sandi baru menambah panjang bit data dari pengirim, sehingga semakin banyak data yang dikirim, maka penambahan data bit untuk sandi barupun akan semakin banyak.
3. Pada deteksi dan koreksi kesalahan, bisa mendeteksi berdasarkan posisi data yang salah dan melakukan perbaikan pada data tersebut.

Daftar Pustaka

1. Fiedler James, *"Hamming Code.pdf"*
2. Forouzan, Behrouz, *"Data Coomunication and Networking 4th edition.pdf"*.
3. Hutauruk, Sindak, "Perancangan simulasi koreksi kesalahan data dengan metode FEC pada komputer berbasis Visual Basic.pdf".
4. Purnomo, Galih, *"Metode Hamming"*.
5. Tanenbaum, *"Computer_network_4th_edition.pdf"*.
6. Xie, Yao, "Lecture 15: *Hamming codes and Viterbi algorithm.pdf"*.