Journal of Computer System and Informatics (JoSYC)
Volume 1, No. 3, May 2020, Page 212-217
ISSN 2714-8912 (media online)
ISSN 2714-7150 (media cetak)

Implementasi Algoritma J-Bit Encoding Pada Kompresi Pesan Teks Short Message Service (SMS)

Husnaini

Program Studi Teknik Informatika, STMIK Budi Darma, Medan, Indonesia
Email: [1]husnaini611@gmail.com

Abstrak− Sms kebutuhan manusia akan teknologi informasi sangat meningkat. Sms tersebut bukan hanya berupa teks, Bisa juga berupa suara atau bahkan gambar/video yang membutuhkan banyak memakan banyak biaya dan menyita ruang penyimpanan. Semakin besar pula ukuran dari *file* yang diproses dan dipertukarkan. Pada penelitian inidigunakan algoritma kompresi J-BIT untuk memperkecil ukuran *file* teks,sehingga akan mengurangi ukuran *file* pada media penyimpanan.

Kata Kunci: Kompresi, Sms, J-Bit Encoding

Abstract− Sms human needs for information technology are greatly increasing. The SMS is not only in the form of text, it can also be a sound or even a picture / video that requires a lot of cost and takes up storage space. The larger the size of the file that is processed and exchanged. In this research, the J-BIT compression algorithm is used to reduce the size of text files, so that it will reduce the size of the file on the storage media.

Keywords: Kompresi, SMS, J-Bit Encoding

1. PENDAHULUAN

Pada saat ini kebutuhan manusia akan teknologi informasi dan komunikasi sangat meningkat. Salah satu teknologi alat komunikasi yaitu telepon genggam atau lebih dikenal sebagai HP (*handphone*).Yang memiliki fasilitas *standart* komunikasi suara yaitu *telephone* dan SMS (*Short Message Service)*,fasilitas tersebut digunakan untuk mengirim dan menerima pesan dalam bentuk teks yaitu sms tidak menggunakan kouta tetapi pulsa. namun dalam pengiriman sms sekarang banyak memakan tempat penyimpanan dan banyak memakan pulsa. Supaya dalam mengirim pesan tidak banyak memakan kapasitas ruang penyimpanan yang di lebih besar serta waktu dan biaya pengaksesan menjadi lebih banyak. Solusi dalam mengatasi masalah tersebut harus dikompresi agar penyimpanan pesan teks tersebut semakin menghemat biaya dalam pengiriman pesan teks. Dalam metode pengkompresian memiliki kelebihan dan kekurangan, yaitu data yang diukur dengan ukuran atau kecepatan kapasitas kecepatan kompresi. Dan pesan teks yang telah dikompresi dapat dikirim lebih mudah dan lebih cepat karna kapasitasnya yang lebih kecil.

Kini aplikasi lainnya seperti *Chating* yang sekarang lebih popular dibandingkan sms. SMS (*Short Message Service*) yang sering kita gunakan untuk berkirim pesan teks ke nomor telepon yang ingin dituju dengan menggunakan jalur selular (makan pulsa).sedangkan Line,Wa,Chating dalam mengirim pesan teks mengunakan jalur internet (paket).bahwasanya dari perbedaannya tiga hal tersebut yang paling nenonjol adalah jalur komunikasinya dan jalur jaringannya.

Kompresi data menjadi pilihan solusi utama untuk menangani setidaknya mengimbangi berkembangnya kebutuhan yang tidak terbendung tersebut.Kompresi sangat berguna ketika data terlalu besar tetapi sangat perlu disimpan dalam tempat yang terbatas.pesan yang dikirimkan dari *sender* ke *receiver* umumnya dengan panjang karakter maksimal adalah 160 karakter untuk setiap satu sms,untuk pengiriman yang lebih dari 160 karakter maka biaya yang dikeluarkan pun berlipat sesuai jumlah karakter SMS. Dengan adanya proses kompresi terhadap pesan teks (sms) maka akan terjadi kompresi terhadap teks sms sehingga dapat menghemat biaya pengirim sms (pulsa).selain itu dengan adanya untuk pemuatan karakter sms yang akan dikirim dalam satu halaman,dengan cara mengkompresi isi pesan atau teks sms dengan mengunakan J-Bit *enconding*.

Terdapat beberapa penelitian terkait dengan algoritma J-Bit *Encoding*, seperti yang dilakukan I Made Agus Dwi Suarjaya dengan judul penelitian A New Algorithm for Data Compression Optimization. hasil penelitian ini mengambil bentuk penelitian bagaimana algoritma J-Bit *Encoding* dikombinasikan dan dibandingkan dengan algoritma lain, dan dari hasil pengujian yang dilakukan algoritma J-Bit *Encoding* bisa dikombinasikan dengan baik untuk beberapa algoritma kompresi berbasis teks seperti LZW, LZ77, Run Length Encoding dan BurrowsWheeler.

Agar proses yang dilakukan lebih mudah,ada pun solusi yg ditawarkan dalam mengkompresikan pesan teks (sms) yaitu dengan menggunakan J-Bit *Encoding*. J-Bit *Encoding* merupakan algoritma kompresi data *lossless* yang memanipulasi bit data dalam sms untuk meminimalkan ukuran dengan cara membagi data menjadi dua keluaran, kemudian dikombinasikan kembali menjadi satu keluaran[2]. Kelebihan dari algoritma J-Bit *Encoding* adalah dapat menghasilkan rasio kompresi yang tinggi. Sebaliknya apabila persentase *byte* nol dari data masukkan rendah, maka algiritma J-*Bit Encoding* akan menghasilkan pesan teks kompresi yang dapat mendekati nilai satu, bahkan bisa melebihi satu (data terekspansi).

2. METODOLOGI PENELITIAN

Submitted: **09/05/2020**; Accepted: **18/05/2020**; Published: **26/05/2020**

Journal of Computer System and Informatics (JoSYC)
Volume 1, No. 3, May 2020, Page 212-217
ISSN 2714-8912 (media online)
ISSN 2714-7150 (media cetak)

2.1 Kompresi

Kompresi merupakan pengurangan ukuran data menjadi ukuran yang lebih kecil dari aslinya. Pengkompresian data ini sangat menguntungkan manakala terdapat suatu data yang berukuran besar dan data di dalamnya mengandung banyak pengulangan karakter. Adapun teknik dari kompresi ini adalah dengan mengganti karakter yang berulangulang tersebut dengan suatu pola tertentu sehingga data tersebut dapat meminimalisasi ukurannya. Masalah kompresi melibatkan proses bagaimana mengoptimalkan algoritma untuk menghapus berbagai redudansi dari suatu objek data. Terdapat beberapa permasalahan umum seperti jumlah bit total yang dikompresi, bagaimana mengembalikan ke bentuk string atau bit asli dan berapa banyak redundasi di ekstraksi dari data asli, pada penelitian skripsi ini penulis menggunakan algoritma J-Bit Encoding (JBE) untuk melakukan kompresi terhadap file teks.

2.2 Dictionary Based Compression Algorithm

Sebelum *dictionary based compression algorithm* ditemukan, algoritma-algoritma kompresi melakukan encode ke setiap simbol menjadi bit string yang menggunakan bit lebih sedikit. Metode di atas kurang optimal bila data yang dikompresi memiliki tingkat redundansi tinggi, seperti pada buku yang memiliki banyak pengulangan kata. *Dictionary based compression algorithm* bekerja dengan mengganti sekelompok simbol dalam data menjadi kode-kode (token) dengan panjang tertentu, dengan asumsi kode-kode tersebut secara umum lebih pendek dari kelompok simbol yang digantikan (Santoso, 2001). Token umumnya berupa campuran nomor halaman dan nomor baris simbol dari frasa yang diganti pada kamus yang berisi frasa yang diganti oleh token tersebut. Bila panjang token lebih pendek dari frasa, maka frasa tersebut telah berhasil dikompresi. Kamus yang digunakan disesuaikan dengan data yang akan dikompresi. Contohnya, bila data kompresi berisi kata-kata dengan bahasa Inggris, maka dapat menggunakan *Oxford English Dictionary* untuk melakukan encode ke token.

2.3 Algoritma J-BIT

J-Bit encoding (JBE) merupakan algorima yang mengoptimalkan masukan untuk algoritma kompresi lainnya seperti algoritma LZW, Run Length dan Half Byte. Konsep utama dari algoritma J-Bit Encoding adalah membagi input yang ada kedalam 2 (dua) buah bentuk data dimana bentuk pertama memiliki keseluruhan zero byte, dan bentuk kedua memiliki keseluruhan informasi nonzero byte.

Berikut adalah beberapa langkah untuk melakukan kompresi file teks dengan menggunakan algoritma J-Bit Encoding.

1. Baca byte per byte dari file
2. Tentukan nilai dari byte yang dengan nonzero atau zero byte
3. Tuliskan nonzero byte dari data I dan berikan bit ‘1’ kedalam temporary byte atau berukan bit ‘0’ into kedalam temporary byte data untuk nilai zero
4. Lakukan proses 1-3 sampai semua temporary byte berisi 8 bit data
5. Jika temporary byte data sudah terisi sebanyak 8 bit, lakukan penulisan temporary data untuk byte kedalam data II
6. Bersihkan temporary byte data
7. Lakukan proses 1-6 secara berulang hingga akhir byte dari file
8. Lakukan proses penulisan dengan menggabungkan data I dan data II
   a. Tuliskan original byte dari file asli
   b. Tuliskan data I.
   c. Tuliskan data II.

3. HASIL DAN PEMBAHASAN

3.1 Analisa Masalah

Dalam hal ini pengirim mengetik pesan sms yang akan di kompresi dengan memasukan nomor tujuan dan selanjutnya pengirim mengkompres pesan tersebut atau pengirim juga bisa mendekompresi pesan tersebut. Penerima pesan melihat isi pesan yang terkompresi dan mengetahui nomor tujuan sipengirim. Dengan teknik kompresi yang digunakan agar penyimpanan pesan teks tersebut semakin menghemat biaya dalam pengiriman pesan teks. Dengan menerapkan algoritma J-Bit *Encoding* merupakan kompresi data *lossless* yang memanipulasi bit data dalam sms untuk meminimalkan ukuran dengan cara membagi data menjadi dua keluaran, apabila persentase byte nol dari data masukan rendah, maka algoritma J-Bit *Encoding* akan menghasilkan pesan teks kompresi yang dapat mendekati nilai satu, bahkan bisa melebihi satu data terekspansi, sehingga dalam pengiriman pesan tidak banyak memakan kapasitas ruang penyimpanan yang di lebih besar serta waktu dan biaya pengaksesan menjadi lebih banyak.

Penelitian ini membangun aplikasi kompresi pesan teks dengan menggunakan bahasa pemograman eclipse. Proses yang dilakukan adalah file teks dengan extensi.*.txt (encoding) terlebih dahulu akan di kompresi, sehingga akan mengurangi ukuran file teks berdasarkan algoritma J-Bit *Encoding*, kemudian hasil encoding kompresi di

Submitted: **09/05/2020**; Accepted: **18/05/2020**; Published: **26/05/2020**

Journal of Computer System and Informatics (JoSYC)
Volume 1, No. 3, May 2020, Page 212-217
ISSN 2714-8912 (media online)
ISSN 2714-7150 (media cetak)

dekompresi kembali berdasarkan J-Bit *Encoding* sehingga menghasilkan decoding hingga dihasilkan ukuran file teks awal.

Dalam masalah kapasitas ruang penyimpanan pesan sms bahkan bisa melebihi satu data terekspansi, sehingga dalam pengiriman pesan tidak banyak memakan kapasitas ruang penyimpanan yang di lebih besar serta waktu dan biaya pengaksesan menjadi lebih banyak. Solusi dalam mengatasi masalah tersebut dapat digunakan teknik kompresi agar penyimpanan pesan teks tersebut semakin menghemat biaya dalam pengiriman pesan teks. Dalam teknik kompresi membutuhkan metode pengkompresian memiliki kelebihan dan kekurangan, yaitu data yang diukur dengan ukuran atau kecepatan kapasitas kecepatan kompresi. Dan pesan teks yang telah dikompresi dapat dikirim lebih mudah dan lebih cepat karna kapasitasnya yang lebih kecil.

3.2 Penerapan Algoritma J-Bit Encoding

Tahapan kompresi J-Bit *Encoding* pada file teks adalah membagi input yang ada kedalam 2 (dua) buah bentuk data dimana bentuk pertama memiliki keseluruhan *zero byte*, dan bentuk kedua memiliki keseluruhan informasi *non zero byte*. Berikut contoh penjabaran kompresi dan dekompresi dengan menggunakan algoritma J-Bit *Encoding*. Contoh berikut ini dimisalkan pesan sms "SEKARANG" hanya akan dikompres atupun didekompresi terhadap beberapa karakter. Berikut penjabaran dengan mengkodekan menurut kode ASCII sebagai berikut :
P = SEKARANG

Kalimat ***P*** kemudian dirubah kedalam bentuk bilangan biner terlebih dahulu sebelum melakukan proses kompres, berikut adalah biner dari setiap karakter dari kalimat ***P***.

Tabel 1. Biner P

No	Karakter	ASCII	Biner
1	S	083	01010011
2	E	069	01000101
3	K	075	01001011
4	A	065	01000001
5	R	082	01010010
6	A	065	01000001
7	N	078	01001110
8	G	071	01000111

Proses berikutnya menentukan *zero byte* dan *non zero byte* dari setiap karakter dari kalimat ***P***, berikut adalah proses kompresi untuk ***P***.

Tabel 2. Proses *Byte* I

Original	Data I
01010011	00000000
01000101	11111111
01001011	00000000
01000001	10001000
01010010	00100010
01000001	01000011
01001110	10101011
01000111	11110101
64 bit	

Langkah berikutnya adalah menentukan nilai bit 0 atau 1 yang dibaca untuk dimasukan kedalam *temporary byte* data, dan pada kasus ini peneliti memilih *byte* 1, maka proses berikutnya adalah menulis semua bit 1 kedalam *non zero byte* dan hasilnya sebagai berikut :
Data I : 00000000 11111111 00000000 10001000
00100010 01000011 10101011 11110101 = **64 bit**
Temporary Data *Byte* : 11111111 10001000 00100010 01000011 10101011 11110101 = **48 bit**
Temporary Data *Byte* : 111111111000100000100010010000111010101 11110101

Kemudian Temporary Byte Data dibagi kedalam kelompok 8 bit untuk dimasukan dan diproses kedalam Data II, dan hasilnya sebagai berikut :

Tabel 3. Bit Data II

***Temporary Byte* Data**	**Data II**
1111111110001000	11111111
0010001001000011	10001000
1010101111110101	00100010
48 bit	01000011

Submitted: **09/05/2020**; Accepted: **18/05/2020**; Published: **26/05/2020**

Journal of Computer System and Informatics (JoSYC)
Volume 1, No. 3, May 2020, Page 212-217
ISSN 2714-8912 (media online)
ISSN 2714-7150 (media cetak)

10101011
11110101

Data II merupakan hasil akhir dari proses dari kompresi dari kalimat ***P*** = SEKARANG, hasil biner tersebut dirubah kedalam bentuk ASCII akan menghasilkan encoding teks " ỹ Ě " C « ŏ " sebagai berikut :
11111111 = 255 : ỹ
10001000 = 136 : Ě
00100010 = 34 : "
01000011 = 67 : C
10101011 = 171 : «
11110101 = 245 : ŏ
Hasil kompresi menjadi 6 *byte* dan hasil kompresi lebih kecil dari file original 8 *byte*
Decoding :

Proses decoding adalah mengembalikan proses kedalam bentuk asli, langkah awal bentuk biner sebagai berikut :
111111111000100000100010010000111010101111110101

Kemudian berikutnya adalah merubah biner yang ada kedalam bentuk 8 bit dimulai dari sebelah kanan dan hasilnya sebagai berikut :
11111111 10001000 00100010 01000011 10101011 11110101

Setelah di konversi dengan melakukan proses data *byte* I dan data *byte* II seperti pada proses kompresi, maka hasil biner ditambahkan *zero byte* dan hasilnya sebagai berikut :

Tabel 4. Proses *Byte* I

Original	Data I
1 1 1 1 1 1 1 1	110011
1 0 0 0 1 0 0 0	100101
0 0 1 0 0 0 1 0	101011
0 1 0 0 0 0 1 1	100010
1 0 1 0 1 0 1 1	110010
1 1 1 1 0 1 0 1	100001
48 bit	101110
	100111

Langkah berikutnya adalah menentukan nilai bit 0 atau 1 yang dibaca untuk dimasukan kedalam *temporary byte* data, dan pada dekompresi memilih *byte* 0, maka proses berikutnya adalah menulis semua bit 0 kedalam *zero byte* dan hasilnya sebagai berikut :
Data I : 110011 100101 101011 100010
110010 100001 101110 100111 = **48 bit**
Temporary Data *Byte* : 01010011 01000101 01001011 01000001 01010010 01000001 01001110 01000111 = **64 bit**
Temporary Data *Byte* : : 01010011 01000101 01001011 01000001 01010010 01000001 01001110 01000111

Kemudian *Temporary Byte* Data dibagi kedalam kelompok 8 bit untuk dimasukan dan diproses kedalam Data II, dan hasilnya sebagai berikut :

Tabel 5. Bit Data II

***Temporary Byte* Data**	**Data II**
	01010011
0101001101000101	01000101
0100101101000001	01001011
0101001001000001	01000001
0100111001000111	01010010
	01000001
64 bit	01001110
	01000111

Data II merupakan hasil akhir dari proses dari dekompresi, hasil biner tersebut dirubah kedalam bentuk ASCII akan menghasilkan dekoding teks" SEKARANG" Hasil dekompresi kembali menjadi 8 *byte* dan hasil dekompresi lebih besar dari file kompresi original 6 *byte*

3.3 Pengujian

Tampilan dalam pengujian program merupakan tampilan halaman yang muncul pertama sekali pada saat sistem dijalankan. Yang terdiri dari Halaman utama menu memiliki 3 menu, yaitu menu tulis pesan, menu baca pesan dan about. Adapun Tampilan Halaman menu utama dapat dilihat pada gambar 1.

Submitted: **09/05/2020**; Accepted: **18/05/2020**; Published: **26/05/2020**

Journal of Computer System and Informatics (JoSYC)
Volume 1, No. 3, May 2020, Page 212-217
ISSN 2714-8912 (media online)
ISSN 2714-7150 (media cetak)

Gambar 1. Tampilan Menu Utama

Form ini digunakan untuk memproses kompresi pesan yang akan dikirim, setelah dikompresi jumlah karater pesan 25 bit setelah dikompresi menjadi 24 bit. Pada proses dekompresi mengembalikan kompresi ke awal. Adapun *form* tersebut dapat dilihat pada gambar 2. dapat dilihat dibawah ini.

Gambar 2. *Form* tulis pesan

Form ini digunakan untuk melihat kotak masuk pesan dari pengirim. Adapun *form* dekripsi tersebut dapat dilihat pada gambar 3. dapat dilihat dibawah ini.

Gambar 3. *Form* Baca pesan

4. KESIMPULAN

Berdasarkan uraian dari bab-bab sebelumnya, maka penulis dapat memberikan kesimpulan sebagai berikut:

1. Proses J-Bit *Encoding* dalam proses sms untuk memanipulasi setiap *bit* dan memperkecil setiap ukuran karakter.
2. Proses kompresi pada sms dapat dilakukan dengan J-Bit encoding sehinggapesan yang dikirim tidak banyak memakan tempat penyimpanan dan menghemat biaya.
3. Perancangan aplikasi dengan menggunakan *Eclipse* yang telah selesai dirancang dengan desain minimalis diharapkan dapat berguna dalam mengirim sms.

Submitted: **09/05/2020**; Accepted: **18/05/2020**; Published: **26/05/2020**

REFERENCES

[1] Utami, Metode Predictive,2013.
[2] Santoso,Dictionary Based Compression Algorithm.2001.
[3] I. Made and A.D . Suarjaya." A new Algoritma for Data Compression Optimization," Int. J. Adv .Conput .sei. Appl, Vol 3, no. 8, PP 14-17.
[4] A.Rachmat , ALGORITMA DAN PEMROGRAMAN DENGAN BAHASA C: KONSEP,TEORI,DAN IMPLEMENTASI .2016.
[5] Mizwar, T. *et al*. (2017) 'IMPLEMENTASI ALGORITMA J-BIT ENCODING PADA KOMPRESI FILE TEKS', *KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer)*, 1(1), pp. 232–236.

Submitted: **09/05/2020**; Accepted: **18/05/2020**; Published: **26/05/2020**