Jurnal Riset Komputer (JURIKOM), Vol. 3 No. 4, Agustus 2016
ISSN 2407-389X (Media Cetak)
Hal : 5-9

IMPLEMENTASI ALGORITMA RUN LENGTH ENCODING PADA KOMPRESI FILE MP3

Darno Willfrid Midukta Simamora[1], Garuda Ginting[2], Yasir Hasan[3]

[1] Mahasiswa Teknik Informatika STMIK Budi Darma
[2,3] Dosen Tetap STMIK Budi Darma
[1,2,3] Jl. Sisingamangaraja No.338 Simpang Limun Medan

ABSTRAK

Pertukaran informasi saat ini membutuhkan kecepatan dalam pengiriman informasi. Kecepatan pengiriman ini sangat bergantung kepada ukuran dari informasi tersebut. Salah satu solusi untuk masalah di atas adalah dengan melakukan pemamfatan (kompresi). Ada banyak sekali metode kompresi data yang ada saat ini, namun pada skripsi ini akan dibahas prinsip kerja algoritma Run Length Encoding dengan implementasi menggunakan visual basic net 2008. Pengiriman file MP3 seringkali terhambat karena besarnya ukuran file yang akan dikirim. Salah satu cara untuk mengatasi hal tersebut adalah dengan memamfatkan (kompresi) file tersebut sebelum dikirim. Penulis menggunakan algoritma Run Length Encoding yang bersifat lossless dalam pemampatan file MP3. Masukan dalam sistem ini adalah file audio MP3. Pada sistem ini terdapat tahap kompresi dan dekompresi. Tahap kompresi bertujuan untuk memampatkan ukuran file MP3, sedangkan tahap dekompresi bertujuan untuk mengembalikan ukuran file MP3 ke ukuran semula. Algoritma yang dapat digunakan untuk proses kompresi salah satunya adalah RLE (Run Lenght Encoding). RLE menjadi salah satu metode memamfatkan data, dengan menggunakan perhitungan matematika terhadap sebuah bilangan, yang dibuat berdasarkan total hexa yang terdapat pada file MP3 tersebut. Dengan adanya penerapan algoritma RLE dalam penelitian ini, diharapkan dapat membantu untuk memamfatkan data lainnya.

Kata Kunci: *Algoritma, kompresi, file mp3*

I. PENDAHULUAN

Kecepatan pengiriman informasi dalam bentuk perpaduan teks, suara dan gambar menjadi bagian utama dalam pertukaran informasi. Kecepatan pengiriman ini sangatlah bergantung kepada ukuran dari informasi tersebut. Salah satu penyelesaian untuk membantu kendala tersebut adalah dengan melakukan pemampatan (kompresi) data baik itu teks, suara, dan citra sebelum dikirimkan dan kemudian penerima akan merekonstruksinya kembali menjadi data aslinya (dekompresi).Kompresi data merupakan suatu hal yang esensial. Teknik kompresi ini esensial karena ukuran dari data semakin lama semakin besar, tetapi belum optimal karena tidak didukung oleh perkembangan dari teknologi penyimpanan data dan *bandwidth* (untuk kecepatan *download* data dari internet) yang seimbang. Sementara orang orang pun menginginkan data dengan kualitas terbaik dan kuantitas (ukuran) yang minimum.

Melihat masalah-masalah tadi, maka pemecahannya adalah maksimalisasi kompresi, yaitu mengurangi tempat yang digunakan oleh data yang dimamfatkan. Pemamfatan data (*data compression*) merupakan salah satu kajian di dalam ilmu komputer yang bertujuan untuk mengurangi ukuran file sebelum menyimpan atau memindahkan data tersebut ke dalam media penyimpanan (*storage device*). Media penyimpanan seperti *flashdisk, hard disk,* dan CD (*Compact Disc*) mempunyai kapasitas yang terbatas. Jika data yang akan disimpan pada media penyimpanan semakin bertambah dan berukuran besar, maka media penyimpanan tidak dapat menyimpan data tersebut karena melebihi kapasitas. Oleh karena itu, untuk mengatasi masalah ini digunakanlah pemamfatan data.

Seperti yang terdapat di jurnal Aditya rahandi,dkk (FASILKOM-TI USU). Dalam penelitian ini penulis menggunakan metode *Run Length Encoding* (RLE) untuk mengkompresi file *audio*. Metode tersebut digunakan untuk mengefisienkan kapasitas penyimpanan data, selain itu juga kompresi data dapat mencegah kerusakan data. Dalam kompresi data, terdapat 4 (empat) faktor penting yang perlu diperhatikan, yaitu: *Time Process* (waktu yang dibutuhkan dalam menjalankan proses), *Completeness* (kelengkapan data setelah file-file tersebut dikompres), *Ratio Compress* (ukuran data setelah dilakukan kompresi), *Optimaly* (perbandingan apakah ukuran file sebelum dikompres sama atau tidak sama dengan file yang telah dikompres).

II. TEORITIS

A. Teori Dasar Kompresi

Proses kompresi merupakan proses mereduksi ukuran psuatu data untuk menghasilkan representasi digital yang padat atau mamfaat (*compact*) namun tetap dapat mewakili kuantitas informasi yang terkandung pada data tersebut. Pada citra video, dan audio, kompresi mengarah pada minimalisasi jumlah bit *rate* untuk representasi digital. Pada beberapa literatur, istilah kompresi sering disebut juga *source coding*, data *compression*, *bandwidth compression*, dan *signal compression* (Putra Darma, 2010).

B. Kompresi Data

Data dan informasi adalah dua hal yang berbeda, pada data terkandung suatu informasi. Namun tidak semua bagian data terkait dengan informasi tersebut atau pada suatu data terdapat bagian data berulang untuk mewakili informasi yang

sama. Bagian data yang tidak terkait atau bagian data yang berulang tersebut disebut dengan data berlebihan(redudancy data),(Putra Darma, 2010).

C. Lossless Compression

Pada teknik ini tidak ada kehilangan informasi. Jika data dimamfaatkan secara *lossless*, data asli dapat direkontruksi kembali sama persis dari data yang telah dimamfaatkan, dengan kata lain data asli tetap sama sebelum dan sesudah pemamfaatan. secara umum teknik *lossless* digunakan untuk penerapan yang tidak bisa mentoleransi setiap perbedaan antara data asli dan data yang telah direkonstruksi. Data berbentuk tulisan misalnya *file* citra, harus dimamfaatkan menggunakan teknik *lossys*, karena kehilangan sebuah karakter saja dapat mengakibatkan kesalapahaman. *Lossless compression* disebut juga dengan *error free compression* (Putra Darma, 2010).

$$\textbf{Rasio Kompresi} = \left(\frac{\text{U File Asli} - \text{U File Terkompresi}}{\text{U File Asli}}\right) \times 100$$

Misalkan rasio kompresi adalah 25%,artinya 25% data semula telah berhasil dimamfatkan.

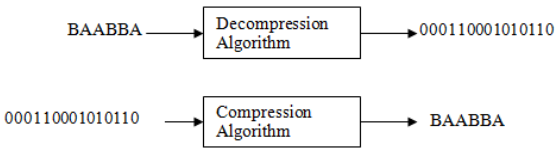
Rasio Kompresi

Rasio kompresi data adalah ukuran persentase data yang telah berhasil dimamfaatkan. Secara matematis rasio pemamfatan data ditulis sebagai berikut:

$$R = 100\% - ((Ko - K1)/Ko) \times 100\ \%$$

Dimana, R = rasio kompresi
Ko = Ukuran file asli
K1 = Ukuran file terkompresi

Sebagai contoh ilustrasi pada gambar 1



Gambar 1 *Lossless Compression*
Sumber : Darma Putra, 2010.

D. File Audio

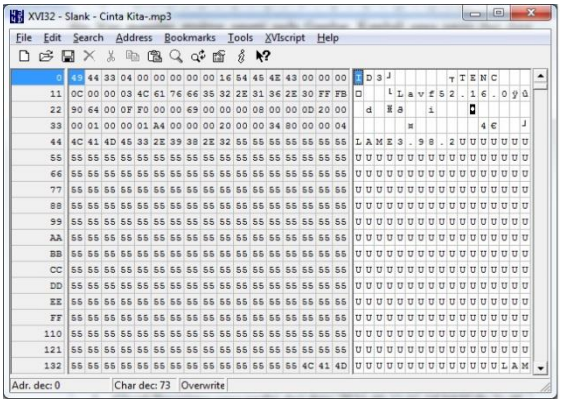
File didalam komputer berwujud file elektronik yang penyimpanannya hanya dapat dilakukan dalam media penyimpanan computer seperti hardisk, disket, CD, atau flashdisk. Berdasarkan bentuknya file dapat berupa teks, gambar, suara, yang masing-masing harus diberi nama secara unik (berbeda)(B.Patmi Istiana dan Y Maryono, 2012).

Audio (sound) adalah fenomena fisik yang dihasilkan oleh getaran suatu benda yang berupa sinyal analog dengan amplitudo yang berubah secara kontinyu terhadap waktu.

Gambar 2. Alur Gelombang Suara
Sumber : Aditya Rahandi, dkk, 2014

E. Struktur *File* Mp3

Mp3 (MPEG Layer-3) merupakan format yang sangat popular untuk pengunduhan dan penyimpanan musik. Dengan mengurangi bagian-bagian dari *file audio* yang tidak terdengar, file. Mp3 dimampatkan secara signifikan sampai 1/10 dari ukuran yang ekuivalen dengan PCM, tetapi dengan tetap mempertahankan kualitas audio yang baik. (Binanto, 2010, hal 60 Bab V). Struktur data pada file *audio* berbeda-beda tergantung format *audio*-nya. Misalnya file Wav memiliki struktur seperti pada Gambar. Kembali sama persis dari data yang telah dikompresi, dengan kata lain data asli tetap sama sebelum dan sesudah kompresi.



Gambar 3. Struktur File Mp3 Dalam Bentuk Hexa

Pada struktur file Mp3 di atas terdiri dari:

1. *Chunk Descriptor* yang terdiri dari data: 4C 41 4D 45 33 2E 39 38 2E 32
2. *Fmt subChunk* yang terdiri data *subChunk1size*, *audioFormat*, *numChannel*, *sampleRate*, *byteRate* dan *BlockAlign* yaitu: 49 44 33 04 00 00 00 00 00 ## 16 54 45 4E 43 00 00 00
3. Data *subChunk* yang terdiri dari data *subChunk2size* serta *sample-sample* 49 44 33 04 # 90 64 00 0F F0 00 00 69 00 00 00 08 00 00 0D 20 00 # 55 55 55 55 55 4C 41 4D.

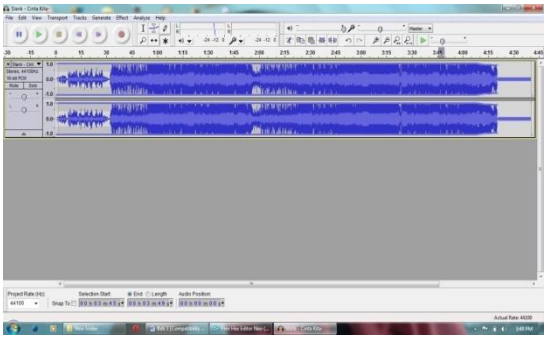
F. Run Length Encoding (RLE)

Kompresi dan dekompresi data menggunakan *run-length encoding* (RLE) ini merupakan suatu bentuk teknik yang digunakan untuk mengkompresi data yang berisi karakter-karakter berulang. *Run-length encoding* (RLE) adalah bentuk yang sangat sederhana dari kompresi data di mana data berjalan (yaitu, urutan di mana nilai data yang sama terjadi pada banyak elemen data yang berturut-turut) disimpan sebagai nilai data tunggal dan dihitung panjangnya. Hal ini berguna pada data yang berisi banyak data berjalan,

misalnya: gambar grafis sederhana seperti ikon, gambar garis, dan animasi. Berbeda dengan teknik-teknik sebelumnya yang bekerja berdasarkan karakter per karakter, teknik run length ini bekerja berdasarkan sederetan karakter yang berurutan. Run Length Encoding adalah suatu algoritma kompresi data yang bersifat lossless. Algoritma ini mungkin merupakan algoritma yang mudah untuk dipahami dan diterapkan untuk kompresi(Aditya Rahandi, dkk, 2014).

III. ANALISA dan PEMBAHASAN

Dalam penelitian ini membahas tentang *file* struktur *file* mp3 yang akan dikompres ke dalam bentuk *file* MP3, teknik yang digunakan dalam menyelesaikan masalah ini adalah teknik *lossless*. Karena *file* asli yang akan kita konversikan dapat dikembalikan kebentuk *file* semula. Tool yang digunakan untuk mengetahui *frekuensi* suara dan tampilan visual pergerakan mp3 tersebut menggunakan *audacity*. *File* mp3 yang diambil dengan cara pilih file mp3 judul Cinta kita berukuran 4,3mb berekstensi mp3 yang di analisa sehingga memudahkan untuk menyelesaikan masalah yang ada. Seperti pada gambar 3.1 dapat dilihat *frekuensi* gelombang Mp3 yang bergerak tidak beraturan,



Gambar 4. Frekuensi File Mp3

Penerapan Metode *Run Length Encoding*

Kompresi *file audio* yang bertipe MP3 dengan menggunakan metode *Run Length Encoding* (RLE). Proses kompresi data didasarkan pada kenyataan bahwa pada hampir semua jenis data selalu terdapat pengulangan pada komponen data yang dimilikinya, misalnya di dalam suatu data *audio* akan terdapat pengulangan penggunaan angka 0 sampai 9 atau huruf a sampai huruf z. Kompresi data melalui proses *encoding* bertujuan untuk menghilangkan unsur pengulangan ini dengan mengubahnya sedemikian rupa sehingga ukuran data menjadi lebih kecil.

Proses pengurangan unsur pengulangan ini dapat dilakukan dengan memakai beberapa teknik kompresi. Misalnya jika suatu komponen muncul berulang kali dalam suatu data, maka komponen tersebut tidak harus di kodekan berulang kali, tapi dapat dikodekan dengan menulis frekuensi muncul komponennya dan dimana komponen tersebut muncul. Teknik kompresi data lainnya, berusaha untuk mencari suatu bentuk kode yang lebih pendek untuk suatu komponen yang sering muncul.

Langkah di dalam proses kompresi dengan algoritma *Run-Length* adalah:

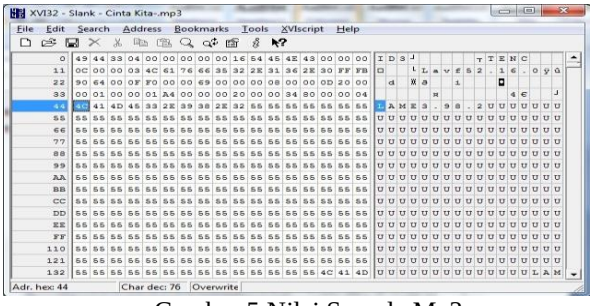
1. Pembacaan sampel *audio*
2. Kompresi data nilai jika ditemukan data nilai sampel yang sama secara berurutan lebih dari dua dengan cara:
   a. Berikan bit penanda # pada data kompresi,
   b. Tambahkan deretan bit untuk menyatakan jumlah nilai data yang sama berurutan.
   c. Tambahkan deretan bit yang menyatakan karakter yang berulang.
   d. Konversikan semua data kompresi ke dalam *hexa*.
3. Simpan file *audio* .
4. Hitung rasio kompresi.

Proses Kompresi *Run Length Encoding*

Secara umum langkah-langkah yang dilakukan untuk kompresi *file* mp3 dengan metode *run length encoding* adalah sebagai berikut:

1. Buka *file* mp3 untuk membaca *header-header* dan sampel mp3.
2. Baca *file audio* untuk mendapatkan data sampel.
3. Ambil nilai sample mp3 ke 1 sampai ke n.
4. Susun nilai sample *audio* pada senarai berantai dengan karakter khusus pembatas antara data sample mp3(#).

Sebagai contoh pada *file* mp3 diperoleh nilai sample mp3 yang akan di-encoding dalam bentuk hexadecimal seperti pada gambar 3.4



Gambar 5.Nilai Sample Mp3

IV. IMPLEMENTASI

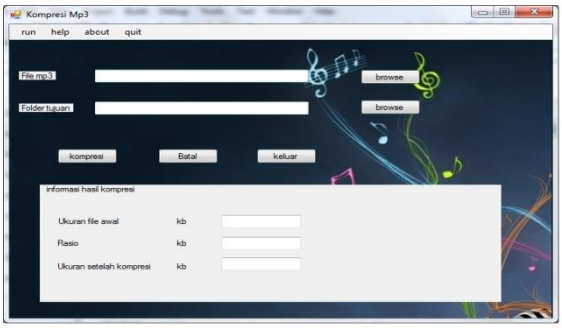
A. Tampilan Menu Utama

Pada tampilan menu utama terdapat tombol Kompresi, tombol Help, tombol About dan tombol Quit. Tampilan Menu Utama dapat dilihat pada Gambar 6.

Gambar 6 Menu Utama

B. Tampilan Kompresi

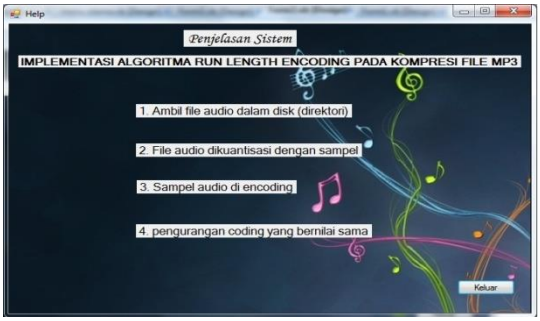
Pada tampilan Kompresi berguna untuk melakukan proses pengurangan bit-bit pada sampel audio serta menyimpan hasilnya. Tampilan Kompresi dapat dilihat pada Gambar 7.



Gambar 7 Tampilan Kompresi

C. Tampilan *Help*

Pada Tampilan Help terdapat tempat untuk menampilkan tampilan bantuan untuk menjalankan aplikasi. Pada Tampilan Help terdapat tombol Keluar untuk menutup tampilan. Tampilan Help dapat dilihat pada Gambar 8



Gambar 8 Tampilan *Help*

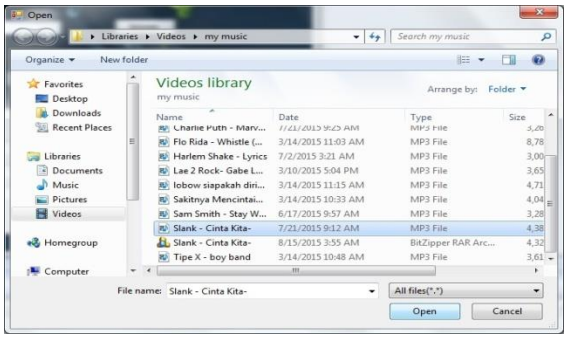
D. Pengujian Sistem

Pengujian sistem adalah dengan melakukan kompresi file audio yang berformat Mp3. Tampilan Pengujian kompresi file Mp3 dapat dilihat pada Gambar 9.

Gambar 9 Tampilan Pengujian Kompresi

E. Pengujian Kompresi File MP3

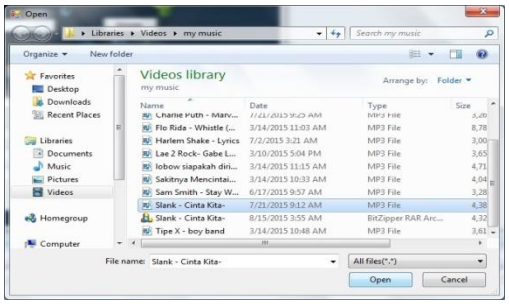
Untuk memanggil file audio pilih tombol browse untuk membuka tampilan kotak dialog seperti pada Gambar 10



Gambar 10 Tampilan Kotak Dialog

F. Pengujian Kompresi File MP3

Untuk memanggil file audio pilih tombol browse untuk membuka tampilan kotak dialog seperti pada Gambar 11

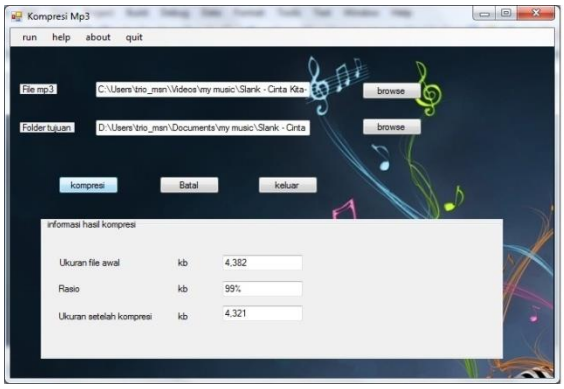


Gambar 11 Tampilan Kotak Dialog

Selanjutnya pilih salah satu file yang ada pada daftar file pada kotak dialog diikuti dengan pemilihan tombol Open dan tampilan proses kompresi dapat dilihat pada Gambar 12.

Gambar 12. Tampilan Proses Kompresi File Mp3

Keterangan:
Tampilan informasi hasil kompresi belum terisi dan tombol <kompresi> dalam keadaan *enabled*. Selajutnya pilih tombol <kompresi> untuk melakukan kompresi terhadap file audio yang telah dipilih pada kotak dialog di atas. Selanjutnya hasil kompresi dapat dilihat pada Gambar 13



Gambar 13 Tampilan Hasil Kompresi Mp3

Keterangan:
Setelah proses kompresi selesai, maka akan muncul pesan konfirmasi bahwa proses kompresi telah selesai dengan hasilnya tertera pada Informasi Hasil Kompresi sebagai berikut:

Ukuran awal : 4,382 kilobytes
Rasio : 99%
Ukuran setelah kompresi : 4,321 kilobytes

G. Pengujian Dekompresi File MP3

Untuk melakukan dekompresi file Mp3 harus secara serentak dengan proses kompresi, karena proses dekompresi memerlukan file asli sebelum kompresi.Proses dekompresi dilakukan dengan memanggil file berformat Mp3 seperti proses kompresi. Pilih tombol *browse* untuk membuka tampilan kotak dialog seperti pada Gambar 14

Gambar 14 Tampilan Hasil kompresi Mp3

Keterangan: Setelah proses dekompresi selesai, maka akan muncul pesan konfirmasi bahwa proses kompresi telah selesai dengan hasilnya tertera pada Informasi Hasil Kompresi sebagai berikut:

Ukuran awal : 4,321 kilobytes
Rasio : 99%
Ukuran setelah kompresi : 4,382 kilobytes

V. KESIMPULAN

Berdasarkan pembahasan, maka dapat ditarik kesimpulan:

1. Proses pengambilan nilai ascii terlabih dahulu buka *file* mp3 kemudian membaca *header*-*header* nilai sampel menggunakan aplikasi hex neo, ambil nilai sampel mp3 ke1 sampai ke n.
2. Penerapan algoritma *Run Length Encoding* yaitu pembacaan *file* mp3, jika adasampel yang sama secara berurutan lebih dari dua di beri tanda #, tambahkan deretan bit menyatakan jumlah nilai data yang sama berurutan tambahkan deretan berulah, lalu simpan *file* mp3, hitung rasio.
3. Kualitas mp3 setelah kompresi hanya mengurangi sedikit kapasitas dan tidak mengubah *file* aslinya.

VI. DAFTAR PUSTAKA

[1] Adi Nugroho, (2010),*"* Rekayasa Perangkat Lunak Berorientasi Objek Dengan Metode USDP" Penerbit Andi, Yogyakarta.
[2] Binanto, (2010), "Multimedia Digital Dasar Teori + Pengembangan", Penerbit Andi, Yogyakarta.
[3] Darma Putra, (2010), "pengolahan citra digital", Penerbit ANDI, Yogyakarta.
[4] Sutabri, (2012), "Analisis Sistem Informasi". Penerbit Andi, Yogyakarta.
[5] B.Patmi Istiana dan Y Maryono Teknologi Informasi Dan Komunikasi 1 kelas VII, 2012 penerbit yudhistira : Yogyakarta.
[6] Edy Irwansyah dan Jurike V. Moniaga, Pengantar Teknologi Informasi 2014 penerbit deepublish : Yogyakarta.
[7] Aditya Rahandi, dkk, (2014) "Analisis dan Implementasi Kompresi File *Audio* Dengan Menggunakan Algoritma *Run Length Encoding* (RLE)", Program Studi S1 Ilmu Komputer, FASILKOM-TI USU, Medan.
[8] Herny Februariyanti dan Setyawan Wibisono, (2010) "Steganografi File Audio Mp3 Menggunakan Mp3Stego", Program Studi Teknik Informatika, Semarang.