

**ANALISIS KINERJA KOMPRESI *FILE AUDIO*
MENGUNAKAN ALGORITMA *ARITHMETIC*
CODING DENGAN METODE BILANGAN *INTEGER***

SKRIPSI

Diajukan untuk Memenuhi Salah Satu Syarat Mencapai Gelar Strata Satu

Jurusan Informatika



Disusun Oleh:

YAHYA FATHONI AMRI

NIM. M0508076

**JURUSAN INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS SEBELAS MARET**

SKRIPSI
ANALISIS KINERJA KOMPRESI FILE AUDIO MENGGUNAKAN
ALGORITMA ARITHMETIC CODING DENGAN METODE BILANGAN
INTEGER

Disusun oleh :

YAHYA FATHONI AMRI

M0508076

telah dipertahankan di hadapan Dewan Penguji
Pada tanggal : 17 Juli 2012


Susunan Dewan Penguji

Pembimbing I


Esti Suryani, S.Si., M.Kom.

NIP. 19761129 200812 2 001

Pembimbing II


Umi Salamah, S.Si., M.Kom.

NIP. 19700217 199702 2 001

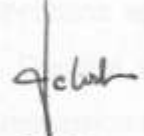
Anggota Dewan Penguji Lain :

1. Dewi Wisnu Wardani, S.Kom., M.S.

NIP. 19781026 200501 2 002

2. Abdul Aziz, S.Kom., M.Cs.


NIP. 19810413 200501 1 001

()

()


Disahkan oleh

Dekan FMIPA UNS


Prof. Ir. Ari Handono Ramelan, M.Sc. (Hons), Ph.D

NIP. 19610223 198601 1 001

Ketua Jurusan Informatika


Umi Salamah, S.Si., M.Kom.

NIP. 19700217 199702 2 001

ABSTRACT

The compression techniques usually process the data transmission and data storage. The advantage of compressed data such as: reducing the bottleneck in the process I / O and data transmission, more efficient in data storage space, complicates the reading of data by unauthorized parties, and facilitate the distribution of data by electronic media or removable media such as flash disks, CD, DVD , and others.

Based on advantages such as the above, this thesis applies the algorithm of data compression on the audio file. The algorithm used is a lossless compression algorithm, namely Arithmetic Coding. The method used in the algorithm is a method with integer. Testing is to determine the ratio of compression, the compression and decompression time. There are four types of audio files tested, and tested 75 times. Tested format audio files are .wav, .aif, .au, and .mid.

The research has produced an audio file compression application using Arithmetic Coding algorithm with integer numbers. The test results obtained average compression ratio is 28.648 %, the average compression time is 5.8 s and the average decompression time is 6.84 s. Meanwhile, the average compression ratio of the largest and the fastest compression and decompression time in a row is the file with format .wav, .mid, .au, and .aif.

Keyword : *Arithmetic Coding, Arithmetic Coding with integer numbers, file audio*

ABSTRAK

Teknik kompresi biasanya digunakan untuk proses transmisi data (*data transmission*) dan penyimpanan data (*storage*). Keuntungan data yang terkompresi antara lain: mengurangi *bottleneck* pada proses *I/O* dan transmisi data, penyimpanan data lebih hemat ruang, mempersulit pembacaan data oleh pihak yang tidak berkepentingan, dan memudahkan distribusi data dengan media elektronik maupun media *removable* seperti *flash disk*, *CD*, *DVD*, dan lain-lain.

Berdasarkan keuntungan seperti di atas, tugas akhir ini menerapkan algoritma kompresi data pada *file audio*. Algoritma yang dipakai adalah algoritma *lossless compression*, yaitu *Arithmetic Coding*. Metode yang dipakai pada algoritma adalah metode dengan bilangan *integer*. Pengujian dilakukan untuk mengetahui besar rasio kompresi, waktu kompresi dan waktu dekompresi. Ada 4 jenis *file audio* yang diujikan, dan dilakukan pengujian sebanyak 75 kali. *File audio* yang diujikan berformat *.wav*, *.aif*, *.au*, dan *.mid*.

Penelitian menghasilkan sebuah aplikasi kompresi *file audio* menggunakan algoritma *Arithmetic Coding* dengan bilangan *integer*. Hasil pengujian diperoleh rata-rata rasio kompresi sebesar 28.648 %, rata-rata waktu kompresi sebesar 5.8 s dan rata-rata waktu dekompresi sebesar 6.84 s. Rata-rata rasio kompresi paling besar serta waktu kompresi dan dekompresi paling cepat berturut-turut adalah pada file yang berformat *.wav*, *.mid*, *.au*, dan *.aif*.

Keyword : *Arithmetic Coding, Arithmetic Coding dengan bilangan integer, file audio*

MOTTO

Sesungguhnya sesudah kesulitan itu ada kemudahan.

-Q.S Al Insyirah: 6-

Allah tidak membebani seseorang melainkan sesuai dengan kesanggupannya. Ia mendapat pahala yang diusahakannya dan ia mendapat siksa yang dikerjakannya.

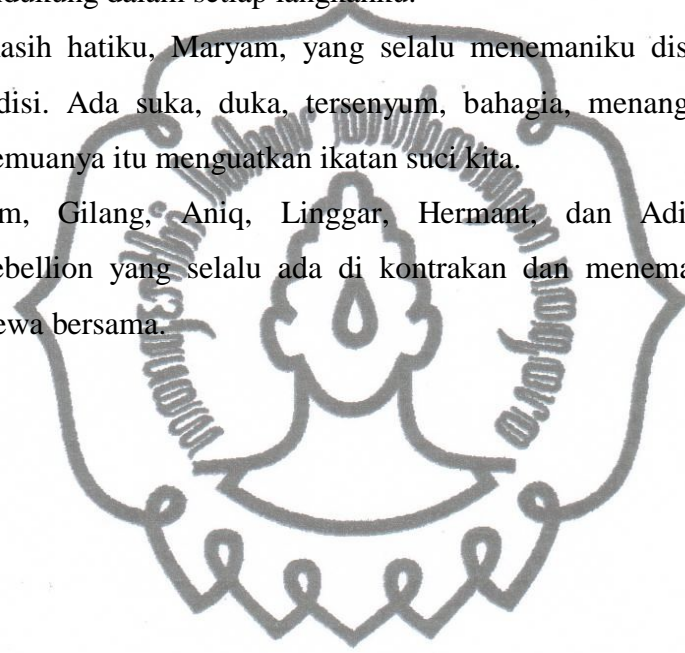
: "Ya Tuhan kami, janganlah Engkau hukum kami jika kami lupa atau kami tersalah. Ya Tuhan kami, janganlah Engkau bebankan kepada kami beban yang berat sebagaimana Engkau bebankan kepada orang-orang sebelum kami. Ya Tuhan kami, janganlah Engkau pikulkan kepada kami apa yang tak sanggup kami memikulnya. Beri ma'aflah kami; ampunilah kami; dan rahmatilah kami. Engkaulah Penolong kami, maka tolonglah kami terhadap kaum yang kafir."

-Q.S Al Baqarah : 286-

PERSEMBAHAN

Skripsi ini saya persembahkan untuk :

- Allah Azza wa Jalla, atas segala nikmat yang tak bisa terhitung jumlahnya.
- Bapak, Ibu dan adik-adiku yang telah membimbing, mengarahi, dan mendukung dalam setiap langkahku.
- Kekasih hatiku, Maryam, yang selalu menemaniku disetiap situasi dan kondisi. Ada suka, duka, tersenyum, bahagia, menangis, tersakiti, dan kesemuanya itu menguatkan ikatan suci kita.
- Imam, Gilang, Aniq, Linggar, Hermant, dan Adit. Teman-teman [R]ebellion yang selalu ada di kontrakan dan menemani. Tertawa dan kecewa bersama.



KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT yang telah memberikan kesempatan kepada penulis untuk menyelesaikan skripsi yang berjudul “Analisis Kinerja Kompresi *File Audio* Menggunakan Algoritma *Arithmetic Coding* dengan Metode Bilangan *Integer*”.

Penulis menyadari akan keterbatasan yang penulis miliki dalam penyusunan skripsi ini, sehingga begitu banyak bantuan dari berbagai pihak yang diberikan kepada penulis dan semoga Allah SWT membalas segala kebaikan mereka. Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. Ibu Umi Salamah, S.Si., M.Kom. selaku Ketua Jurusan S1 Informatika.
2. Bapak Wiharto, S.T., M.Kom. selaku Pembimbing Akademik.
3. Ibu Esti Suryani, S.Si., M.Kom. selaku Dosen Pembimbing I yang telah memberikan pengarahan selama proses penyusunan skripsi ini.
4. Ibu Umi Salamah, S.Si., M.Kom. selaku Dosen Pembimbing II yang telah memberikan pengarahan selama proses penyusunan skripsi ini.
5. Bapak dan Ibu tercinta atas do’a, nasihat, kasih sayang, dan pengorbanan yang tulus.
6. Teman-teman seperjuangan di Jurusan Informatika UNS yang telah memberikan dukungan dan bersedia berbagi ilmu.
7. Semua pihak yang tidak dapat penulis sebutkan satu persatu.

Semoga laporan skripsi ini bermanfaat bagi pembaca dan semua pihak yang berkepentingan.

Surakarta, 22 Juli 2012

Penulis

commit to user

DAFTAR ISI

HALAMAN JUDUL	i
HALAMAN PENGESAHAN	ii
ABSTRACT	iii
ABSTRAK	iv
MOTTO	v
PERSEMBAHAN	vi
KATA PENGANTAR	vii
DAFTAR ISI	viii
DAFTAR GAMBAR	x
DAFTAR TABEL	xii
DAFTAR LAMPIRAN	xv
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah	2
1.4 Tujuan Penelitian	2
1.5 Manfaat Penelitian	3
1.5 Sistematika Penulisan	3
BAB II TINJAUAN PUSTAKA	4
2.1 Dasar Teori	4
2.1.1 Kompresi Data	4
2.1.2 <i>File Audio</i>	9
2.1.3 Struktur Data pada <i>File Audio</i> dan Pembacaanya	10
2.1.4 Algoritma <i>Arithmetic Coding</i>	11
2.2 Penelitian Terkait	16
2.3 Rencana Penelitian	19
BAB III METODOLOGI PENELITIAN	21
3.1 Studi Literatur	21

commit to user

3.2 Identifikasi Masalah	21
3.3 Tahapan Implementasi	21
3.3.1 Tahap Kompresi	22
3.3.2 Tahap Dekompresi	24
3.4 Tahap Pengujian	26
BAB IV HASIL DAN PEMBAHASAN	28
4.1 Spesifikasi Perangkat	28
4.2 Hasil Implementasi	28
4.3 Hasil Pengujian dan Analisa.....	31
4.3.1 Hasil Pengujian dan Analisa Rasio dan Waktu Kompresi.....	31
4.3.2 Hasil Pengujian dan Analisa Waktu Dekompresi.....	45
4.3.3 Hasil Pengujian Kualitas Hasil	59
BAB V KESIMPULAN DAN SARAN	60
5.1 Kesimpulan.....	60
5.2 Saran.....	60
DAFTAR PUSTAKA	61
LAMPIRAN.....	63

DAFTAR GAMBAR

Gambar 2.1. Klasifikasi dari Beberapa Teknik Kompresi Data	6
Gambar 2.2. <i>Lossless Compression</i>	8
Gambar 2.3. <i>Lossy Compression</i>	9
Gambar 2.4. Struktur <i>File</i> Berformat <i>wav</i> Dalam Bentuk <i>Hexadecimal</i>	10
Gambar 2.5. Frekuensi <i>File Audio</i>	11
Gambar 2.6. <i>Encoding Sample Audio</i>	11
Gambar 3.1. Proses Kompresi	21
Gambar 3.2. Proses Dekompresi	22
Gambar 3.3. <i>Flowchart</i> Proses Kompresi Menggunakan <i>Arithmetic Coding</i> dengan Bilangan <i>Integer</i>	23
Gambar 3.4. <i>Flowchart</i> Proses Dekompresi Menggunakan <i>Arithmetic Coding</i> dengan Bilangan <i>Integer</i>	25
Gambar 4.1. Tampilan Menu Kompresi/Dekompresi	28
Gambar 4.2. Grafik Kompresi <i>File</i> Berformat <i>.wav</i> yang Berukuran 0 – 1 MB Berdasarkan Waktu Kompresi Terhadap Ukuran <i>File</i>	31
Gambar 4.3. Grafik Kompresi <i>File</i> Berformat <i>.wav</i> yang Berukuran 1 – 5 MB Berdasarkan Waktu Kompresi Terhadap Ukuran <i>File</i>	32
Gambar 4.4. Grafik Kompresi <i>File</i> Berformat <i>.wav</i> yang Berukuran 5 – 10 MB Berdasarkan Waktu Kompresi Terhadap Ukuran <i>File</i>	33
Gambar 4.5. Grafik Kompresi <i>File</i> Berformat <i>.wav</i> yang Memiliki Satu Jenis Suara Berdasarkan Waktu Kompresi Terhadap Ukuran <i>File</i>	34
Gambar 4.6. Grafik Kompresi <i>File</i> Berformat <i>.wav</i> yang Memiliki Dua Jenis Suara Berdasarkan Waktu Kompresi Terhadap Ukuran <i>File</i>	35
Gambar 4.7. Grafik Kompresi <i>File</i> Berformat <i>.wav</i> yang Memiliki Beragam Jenis Suara Berdasarkan Waktu Kompresi Terhadap Ukuran <i>File</i>	36
Gambar 4.8. Grafik Kompresi <i>File</i> Berformat <i>.aif</i> Berdasarkan Waktu Kompresi Terhadap Ukuran <i>File</i>	38
Gambar 4.9. Grafik Kompresi <i>File</i> Berformat <i>.au</i> yang Berukuran 0 – 1 MB Berdasarkan Waktu Kompresi Terhadap Ukuran <i>File</i>	39

Gambar 4.10. Grafik Kompresi <i>File</i> Berformat <i>.au</i> yang Berukuran 1 – 5 MB Berdasarkan Waktu Kompresi Terhadap Ukuran <i>File</i>	40
Gambar 4.11. Grafik Kompresi <i>File</i> Berformat <i>.au</i> yang Berukuran 5 – 10 MB Berdasarkan Waktu Kompresi Ukuran <i>File</i>	41
Gambar 4.12. Grafik Kompresi <i>File</i> Berformat <i>.mid</i> Berdasarkan Waktu Kompresi Terhadap Ukuran <i>File</i>	43
Gambar 4.13. Grafik Dekompresi <i>File</i> Berformat <i>.wav</i> yang Ukuran Awalnya 0–1 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran <i>File</i>	46
Gambar 4.14. Grafik Dekompresi <i>File</i> Berformat <i>.wav</i> yang Ukuran Awalnya 1–5 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran <i>File</i>	47
Gambar 4.15. Grafik Dekompresi <i>File</i> Berformat <i>.wav</i> yang Ukuran Awalnya 5 – 10 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran <i>File</i>	48
Gambar 4.16. Grafik Dekompresi <i>File</i> Berformat <i>.wav</i> yang Memiliki Satu Jenis Suara Berdasarkan Waktu Dekompresi Terhadap Ukuran <i>File</i>	49
Gambar 4.17. Grafik Dekompresi <i>File</i> Berformat <i>.wav</i> yang Memiliki Dua Jenis Suara Berdasarkan Waktu Dekompresi Terhadap Ukuran <i>File</i>	50
Gambar 4.18. Grafik Dekompresi <i>File</i> Berformat <i>.wav</i> yang Memiliki Beragam Jenis Suara Berdasarkan Waktu Dekompresi Terhadap Ukuran <i>File</i>	51
Gambar 4.19. Grafik Dekompresi <i>File</i> Berformat <i>.aif</i> Berdasarkan Waktu Dekompresi Terhadap Ukuran <i>File</i>	52
Gambar 4.20. Grafik Dekompresi <i>File</i> Berformat <i>.au</i> yang Ukuran Awalnya 0 – 1 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran <i>File</i>	53
Gambar 4.21. Grafik Dekompresi <i>File</i> Berformat <i>.au</i> yang Ukuran Awalnya 1 – 5 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran <i>File</i>	54
Gambar 4.22. Grafik Dekompresi <i>File</i> Berformat <i>.au</i> yang Ukuran Awalnya 5 – 10 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran <i>File</i>	55
Gambar 4.23. Grafik Dekompresi <i>File</i> Berformat <i>.mid</i> Berdasarkan Waktu Dekompresi Terhadap Ukuran <i>File</i>	57

DAFTAR TABEL

Tabel 2.1. Tabel Probabilitas	13
Tabel 2.2. <i>Range</i> Probabilitas	13
Tabel 4.1. Rasio Kompresi untuk <i>File</i> Berformat <i>.wav</i> yang Berukuran 0 – 1 MB	31
Tabel 4.2. Rasio Kompresi untuk <i>File</i> Berformat <i>.wav</i> yang Berukuran 1 – 5 MB	32
Tabel 4.3. Rasio Kompresi untuk <i>File</i> Berformat <i>.wav</i> yang Berukuran 5 – 10 MB	33
Tabel 4.4. Rasio Kompresi untuk <i>File</i> Berformat <i>.wav</i> yang Memiliki Satu Jenis Suara	34
Tabel 4.5. Rasio Kompresi untuk <i>File</i> Berformat <i>.wav</i> yang Memiliki Dua Jenis Suara	35
Tabel 4.6. Rasio Kompresi untuk <i>File</i> Berformat <i>.wav</i> yang Memiliki Beragam Jenis Suara	36
Tabel 4.7. Rata-rata Rasio Kompresi untuk Keseluruhan <i>File</i> yang Berformat <i>.wav</i>	37
Tabel 4.8. Rasio Kompresi untuk <i>File</i> Berformat <i>.aif</i>	37
Tabel 4.9. Rasio Kompresi untuk <i>File</i> Berformat <i>.au</i> yang Berukuran 0 – 1 MB	39
Tabel 4.10 Rasio Kompresi untuk <i>File</i> Berformat <i>.au</i> yang Berukuran 1 – 5 MB	40
Tabel 4.11 Rasio Kompresi untuk <i>File</i> Berformat <i>.au</i> yang Berukuran 5 – 10 MB	41
Tabel 4.12 Rata-rata Rasio Kompresi untuk Keseluruhan <i>File</i> yang Berformat <i>.au</i>	42
Tabel 4.13 Rasio Kompresi untuk <i>File</i> Berformat <i>.mid</i>	42
Tabel 4.14. Rata-rata Rasio Kompresi	43
Tabel 4.15. Rata-rata Waktu Kompresi/Byte untuk <i>File</i> yang Berukuran 0 – 1 MB	44

Tabel 4.16. Rata-rata Waktu Kompresi/Byte untuk File yang Berukuran 1 – 5 MB	44
Tabel 4.17. Rata-rata Waktu Kompresi/Byte untuk File yang Berukuran 5 – 10 MB	44
Tabel 4.18. Waktu Dekompresi <i>File</i> Berformat <i>.wav</i> yang Ukuran Awalnya 0 – 1 MB	45
Tabel 4.19. Waktu Dekompresi <i>File</i> Berformat <i>.wav</i> yang Ukuran Awalnya 1 – 5 MB	46
Tabel 4.20. Waktu Dekompresi <i>File</i> Berformat <i>.wav</i> yang Berukuran Awal 5 – 10 MB	47
Tabel 4.21. Waktu Dekompresi <i>File</i> Berformat <i>.wav</i> yang Memiliki Satu Jenis Suara	48
Tabel 4.22. Waktu Dekompresi <i>File</i> Berformat <i>.wav</i> yang Memiliki Dua Jenis Suara	49
Tabel 4.23. Waktu Dekompresi <i>File</i> Berformat <i>.wav</i> yang Memiliki Beragam Jenis Suara	50
Tabel 4.24. Waktu Dekompresi untuk <i>File</i> Berformat <i>.aif</i>	51
Tabel 4.25. Waktu Dekompresi <i>File</i> Berformat <i>.au</i> yang Ukuran Awalnya 0 – 1 MB	53
Tabel 4.26. Waktu Dekompresi <i>File</i> Berformat <i>.au</i> yang Ukuran Awalnya 1 – 5 MB	54
Tabel 4.27. Waktu Dekompresi <i>File</i> Berformat <i>.au</i> yang Ukuran Awalnya 5 – 10 MB	55
Tabel 4.28. Waktu Dekompresi untuk <i>File</i> Berformat <i>.mid</i>	56
Tabel 4.29. Rata-rata Waktu Dekompresi/Byte untuk File yang Berukuran 0 – 1 MB	57
Tabel 4.30. Rata-rata Waktu Dekompresi/Byte untuk File yang Berukuran 1 – 5 MB	58
Tabel 4.31. Rata-rata Waktu Kompresi/Byte untuk File yang Berukuran 5 – 10 MB	58

commit to user

Tabel 4.32. Kesamaan Kualitas File Sebelum Dikompresi dan Sesudah Didekompresi	59
--	----



DAFTAR LAMPIRAN

LAMPIRAN I	59
LAMPIRAN II	69
LAMPIRAN III	76



BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada dasarnya teknik kompresi digunakan untuk proses transmisi data (*data transmission*) dan penyimpanan data (*storage*). Bila ditinjau dari sisi penggunaannya, kompresi data bisa bersifat umum untuk segala keperluan atau bersifat khusus untuk keperluan tertentu. Keuntungan data yang terkompresi antara lain: mengurangi *bottleneck* pada proses *I/O* dan transmisi data, penyimpanan data lebih hemat ruang, mempersulit pembacaan data oleh pihak yang tidak berkepentingan, dan memudahkan distribusi data dengan media elektronik maupun media *removable* seperti *flash disk*, *CD*, *DVD*, dan lain-lain.

Arithmetic Coding adalah salah satu algoritma *lossless data compression*, yang terkenal mempunyai kemungkinan rasio kompresi terbaik daripada algoritma *lossless data compression* lainnya. Hal ini dikarenakan *Arithmetic Coding* mempunyai efisiensi yang tinggi dan dapat dengan mudah diimplementasikan pada *hardware*. *Decoder* pada *Arithmetic Coding* menggunakan kode sumber yang hampir sama dengan *encoder* yang juga membuat implementasinya mudah (Boden, Clasen, dan Kneis, 2007). Menurut Gozali dan Mervyn, 2004, *Arithmetic Coding* dengan metode bilangan *integer* juga dipakai karena jumlah *bit* kodenya lebih sedikit dan mempercepat proses kompresi dan dekompresi data karena perhitungan *integer* jauh lebih cepat dari perhitungan *floating point* serta dapat diimplementasikan dalam program, tetapi masih terbatas pada *file* teks.

File audio adalah salah satu *file* yang paling sering digunakan untuk tranmisi data maupun penyimpanan. *File* ini mempunyai ukuran yang termasuk besar. Dengan ukurannya ini, akan membuat kesulitan dalam tranmisi maupun penyimpanan data. Oleh karena itu, perlu dilakukan kompresi data untuk memperkecil ukurannya sehingga proses tranmisi dan penyimpanan akan lebih mudah. Dengan hipotesa, rata-rata rasio kompresi, waktu kompresi, dan waktu dekompresi yang akan dihasilkan nantinya secara berturut-turut adalah sebesar 15 %, 2 detik, dan 4 detik dan waktu kompresi dan waktu dekompresi akan naik

secara linier sesuai dengan ukuran *file*. Hipotesa ini didasarkan pada hasil di penelitian algoritma *Arithmetic Coding* pada *file* teks sebelumnya.

1.2 Rumusan Masalah

Rumusan masalah dalam penulisan penelitian ini adalah :

1. Bagaimana menerapkan cara kerja algoritma *Arithmetic Coding* dengan metode bilangan *integer*.
2. Seberapa baik rasio kompresi, waktu kompresi dan dekompresi yang dihasilkan oleh algoritma *Arithmetic Coding* dengan bilangan *integer* dalam mengkompresi *file audio*.

1.3 Batasan Masalah

File audio yang dikompresi dalam penelitian ini dibatasi hanya untuk *file audio* yang tidak terkompresi khususnya yang berformat *wav*, *aif*, *au*, dan *mid*.

1.4 Tujuan Penelitian

Tujuan yang ingin dicapai dalam penulisan tugas akhir ini adalah :

1. Mempelajari dan memahami penerapan cara kerja algoritma *Arithmetic Coding* dengan metode bilangan *integer*.
2. Mengetahui rasio kompresi, waktu kompresi dan waktu dekompresi serta kualitas hasil kompresi yang dapat dihasilkan dari penerapan metode *Arithmetic Coding* dengan bilangan *integer* ini dalam mengkompresi *file audio*.

1.5 Manfaat Penelitian

Manfaat yang ingin dicapai dalam penelitian ini adalah :

- a. Mengkompresi *file audio* untuk memperkecil ukuran *file* sehingga mudah ditranmisikan dan menghemat penyimpanan.
- b. Mengetahui keefektifan penerapan algoritma *Arithmetic Coding* dengan bilangan *integer* pada beberapa jenis *file audio*.

commit to user

1.6 Sistematika Penulisan

Penelitian ini akan disusun dengan sistematika penulisan sebagai berikut :

BAB I PENDAHULUAN

Bab I merupakan bab pendahuluan yang menguraikan latar belakang masalah, rumusan masalah, pembatasan masalah, tujuan, dan sistematika penulisan.

BAB II DASAR TEORI

Bab II berisi tentang teori-teori yang digunakan sebagai landasan dalam penelitian.

BAB III METODE PENELITIAN

Bab III membahas langkah dari proses perancangan dan pembangunan aplikasi kompresi *file audio* menggunakan algoritma *Arithmetic Coding* dengan bilangan *integer*.

BAB IV HASIL DAN PEMBAHASAN

Bab IV berisi tentang eksperimen pengujian dari aplikasi yang telah dibuat dengan menggunakan *sample* data yang telah disiapkan untuk kemudian dilakukan analisa terhadap hasil dari eksperimen yang telah dilakukan.

BAB V PENUTUP

Bab V menguraikan kesimpulan tugas akhir dan saran-saran sebagai bahan pertimbangan untuk pengembangan penelitian selanjutnya.

BAB II

TINJAUAN PUSTAKA

2.1 Dasar Teori

2.1.1 Kompresi Data

Kompresi data adalah proses mengubah sebuah input *stream* data (sumber *stream* atau data mentah asli) ke aliran data lain (output, *bitstream*, atau aliran terkompresi) yang memiliki ukuran lebih kecil. Sebuah *stream* dapat berupa *file*, *buffer* di memori, atau *bit* individun yang dikirim pada sebuah saluran komunikasi (Salomon dan Motta, 2010).

Proses kompresi data didasarkan pada kenyataan bahwa pada hampir semua jenis data selalu terdapat pengulangan pada komponen data yang dimilikinya, misalnya didalam suatu teks kalimat akan terdapat pengulangan penggunaan huruf alphabet dari huruf a sampai dengan huruf z. Kompresi data melalui proses *encoding* berusaha untuk menghilangkan unsur pengulangan ini dengan mengubahnya sedemikian rupa sehingga ukuran data menjadi lebih kecil.

Berdasarkan tipe peta kode yang digunakan untuk mengubah pesan awal (isi *file* input) menjadi sekumpulan *codeword*, metode kompresi terbagi menjadi dua kelompok (Panggabean dan Linawati, 2004) :

- a. Metode statik : menggunakan peta kode yang selalu sama. Metode ini membutuhkan dua fase (*two-pass*): fase pertama untuk menghitung probabilitas kemunculan tiap simbol/karakter dan menentukan peta kodenya, dan fase kedua untuk mengubah pesan menjadi kumpulan kode yang akan ditransmisikan. Contoh: algoritma *Huffman* statik.
- b. Metode dinamik (adaptif) : menggunakan peta kode yang dapat berubah dari waktu ke waktu. Metode ini disebut adaptif karena peta kode mampu beradaptasi terhadap perubahan karakteristik isi *file* selama proses kompresi berlangsung. Metode ini bersifat *one pass*, karena hanya diperlukan satu kali pembacaan terhadap isi *file*. Contoh: algoritma LZW (*Lempel–Ziv–Welch*) yaitu algoritma yang melakukan kompresi dengan

menggunakan *dictionary*, di mana fragmen-fragmen teks digantikan dengan indeks yang diperoleh dari sebuah “kamus” dan algoritma DMC (*Dynamic Markov Compression*) yaitu algoritma yang membuat probabilitas dari karakter biner berikutnya dengan menggunakan pemodelan *finite-state*, dengan model awal berupa mesin FSA dengan transisi 0/1 dan 1/1.

Berdasarkan teknik pengkodean/pengubahan simbol yang digunakan, metode kompresi dapat dibagi ke dalam tiga kategori (Panggabean dan Linawati, 2004), yaitu :

- a. Metode *symbolwise* : menghitung peluang kemunculan dari tiap simbol dalam *file* input, lalu mengkodekan satu simbol dalam satu waktu, dimana simbol yang lebih sering muncul diberi kode lebih pendek dibandingkan simbol yang lebih jarang muncul. Contoh: algoritma *Huffman*.
- b. Metode *dictionary* : menggantikan karakter/fragmen dalam *file* input dengan indeks lokasi dari karakter/fragmen tersebut dalam sebuah kamus (*dictionary*). Contoh : algoritma LZW (*Lempel–Ziv–Welch*).
- c. Metode *predictive* : menggunakan model *finite-context* atau *finite-state* untuk memprediksi distribusi probabilitas dari simbol-simbol selanjutnya. Contoh : algoritma DMC (*Dynamic Markov Compression*).

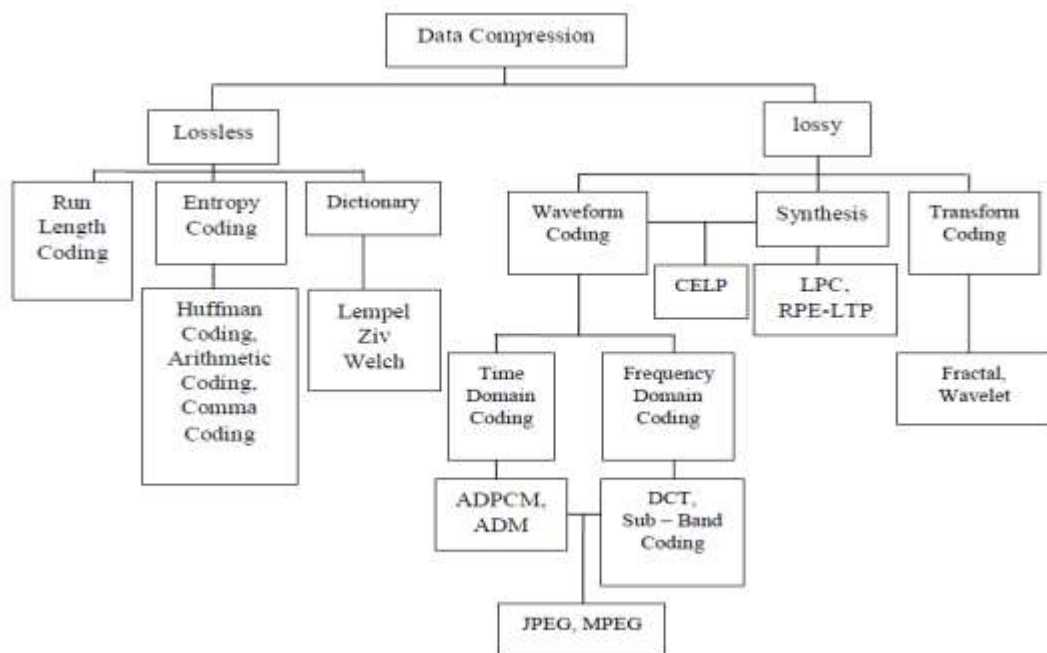
Kompresi data sangat populer sekarang ini karena dua alasan yaitu (Salomon dan Motta, 2010):

1. Orang – orang lebih suka mengumpulkan data. Tidak peduli seberapa besar media penyimpanan yang dimilikinya. Akan tetapi cepat atau lambat akan terjadi *overflow*.
2. Orang – orang benci menunggu waktu yang lama untuk memindahkan data. Misalnya ketika duduk di depan komputer untuk menunggu halaman *Web* terbuka atau men-*download* sebuah *file*.

Ada banyak metode yang dikenal untuk kompresi data. Semuanya didasarkan pada ide yang berbeda, yang cocok untuk berbagai jenis data, dan menghasilkan hasil yang berbeda, tapi semuanya didasarkan pada prinsip

yang sama, yaitu mengkompresi data dengan menghilangkan redundansi dari data asli dalam file sumber (Salomon dan Motta, 2010).

Secara garis besar klasifikasi metode kompresi data dapat dilihat pada Gambar 2.1.



Gambar 2.1. Klasifikasi dari Beberapa Teknik Kompresi Data [Fauzi, 2009]

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

$$\text{Rasio Kompresi} = \frac{(\text{ukuran file asli} - \text{ukuran file terkompresi})}{\text{ukuran file asli}} \times 100\% \quad \dots 2.1$$

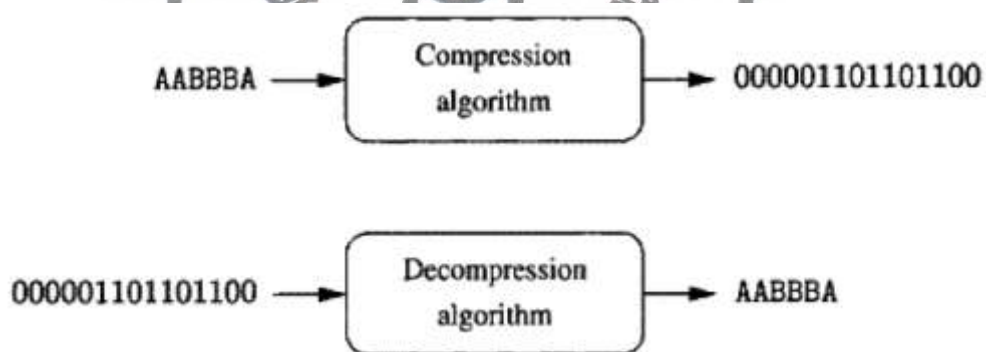
Metode pemampatan data atau kompresi data dapat dikelompokkan dalam dua kelompok besar yaitu :

1. *Lossless Compression*

Pada teknik ini tidak ada kehilangan informasi. Jika data dimampatkan secara *lossless*, data asli dapat direkonstruksi kembali sama persis dari data yang telah dimampatkan, dengan kata lain data asli tetap sama sebelum dan sesudah pemampatan. 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* teks,

harus dimampatkan menggunakan teknik *lossless*, karena kehilangan sebuah karakter saja dapat mengakibatkan kesalahpahaman. *Lossless compression* disebut juga dengan *reversible compression* karena data asli bisa dikembalikan dengan sempurna. Akan tetapi rasio kompresinya sangat rendah, misalnya pada data teks, gambar seperti GIF dan PNG. Contoh metode ini adalah *Shannon-Fano Coding*, *Huffman Coding*, *Arithmetic Coding* dan lain sebagainya.

Kebanyakan program kompresi *lossless* menggunakan dua jenis algoritma yang berbeda: yang satu menghasilkan model statistik untuk input data, dan yang lainnya memetakan data input ke rangkaian bit menggunakan model ini dengan cara bahwa “*probable*” data akan menghasilkan *output* yang lebih pendek dari “*improbable*” data. algoritma *encoding* yang utama yang dipakai untuk menghasilkan rangkaian bit adalah *Huffman Coding* dan *Aritmetic Coding*.



Gambar 2.2. *Lossless Compression* (Pu, 2006)

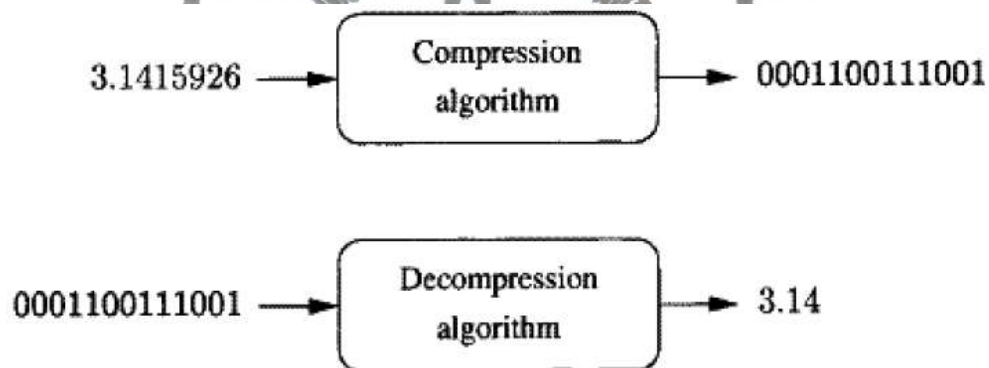
2. *Lossy Compression*

Pada teknik ini akan terjadi kehilangan sebagian informasi. Data yang telah dimampatkan dengan teknik ini secara umum tidak bisa direkonstruksi sama persis dari data aslinya. Di dalam banyak penerapan, rekonstruksi yang tepat bukan suatu masalah. Sebagai contoh, ketika sebuah *sample* suara ditransmisikan, nilai eksak dari setiap *sample* suara belum tentu diperlukan. Tergantung pada yang memerlukan kualitas suara yang direkonstruksi,

commit to user

sehingga banyaknya jumlah informasi yang hilang di sekitar nilai dari setiap *sample* dapat ditoleransi.

Biasanya teknik ini membuang bagian-bagian data yang sebenarnya tidak begitu berguna, tidak begitu dirasakan, tidak begitu dilihat sehingga manusia masih beranggapan bahwa data tersebut masih bisa digunakan walaupun sudah dikompresi. Misalnya pada gambar dan MP3. Contoh metode ini adalah *Transform Coding*, *Wavelet*, dan lain-lain. *Lossy compression* disebut juga *irreversible compression* karena data asli mustahil untuk dikembalikan seperti semula. Kelebihan teknik ini adalah rasio kompresi yang tinggi dibanding metode *lossless*.



Gambar 2.3. *Lossy Compression* (Pu, 2006)

2.1.2 File Audio

Suatu format *file audio* adalah format *file* untuk menyimpan data *audio* digital pada sistem komputer. Data ini dapat disimpan tanpa dikompresi, atau dikompresi untuk mengurangi ukuran *file*. Ini bisa menjadi sebuah *raw bitstream*, tetapi biasanya merupakan sebuah *container format* atau format data *audio* dengan lapisan penyimpanan yang telah ditetapkan.

Penting untuk membedakan antara format *file* dan codec *audio*. Codec melakukan encoding dan decoding data *audio* mentah sementara data itu sendiri disimpan dalam *file* dengan format *file audio* tertentu. Meskipun sebagian besar format *file audio* hanya mendukung satu jenis data *audio* (dibuat dengan coder *audio*), multimedia container format (seperti Matroska

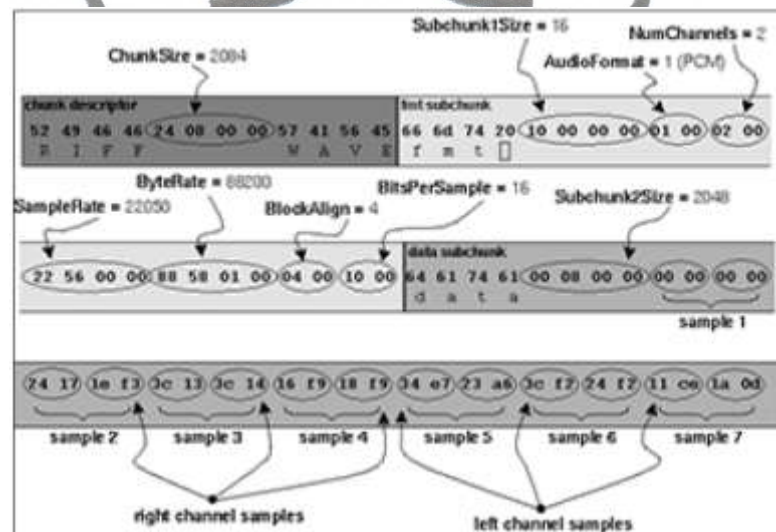
commit to user

atau AVI) dapat mendukung beberapa jenis data *audio* dan video. Ada tiga kelompok utama format *file audio*.

- Format *audio* yang tidak terkompresi, seperti WAV, AIFF, AU atau raw header-less PCM;
- Format *audio* dengan kompresi lossless, seperti FLAC, Monkey's Audio (yang berekstensi APE), WavPack (yang berekstensi WV), TTA, ATRAC Advance Lossless, Apple Lossless yang berekstensi m4a), MPEG-4 SLS, MPEG-4 ALS, MPEG -4 DST, Windows Media Audio Lossless (WMA Lossless), dan Shorten (SHN);
- Format *audio* dengan kompresi lossy, seperti MP3, Vorbis, Musepack, AAC, ATRAC dan Windows Media Audio lossy (WMA lossy).

2.1.3 Struktur Data pada File Audio dan Pembacaannya

Struktur data pada *file audio* berbeda-beda tergantung *format audio*-nya. Misalnya *file wav* memiliki struktur seperti pada Gambar 2.4.



Gambar 2.4. Struktur File Berformat wav Dalam Bentuk Hexadecimal

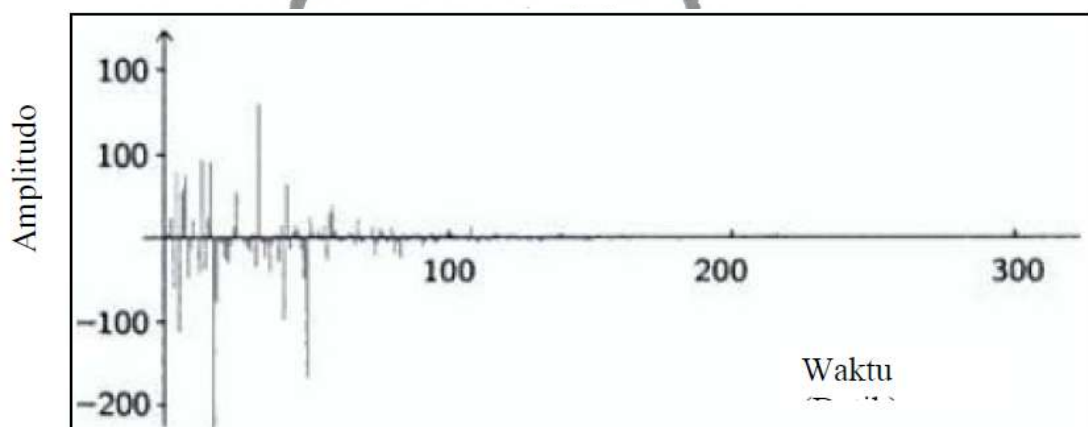
Pada struktur *file wav* di atas terdiri dari:

- a. *Chunk Descriptor* yang terdiri dari data: 52 49 46 46 28 08 00 00 57 41 56 45.
- b. *Fmt subChunk* yang terdiri data *subChunk1size*, *audioFormat*, *numChannel*, *sampleRate*, *byteRate* dan *BlockAlign* yaitu:

66 6d 74 20 10 00 00 00 01 00 02 00 22 56 00 00 ## 50 01 00 04 00 10 00

c. Data *subChunk* yang terdiri dari data *subChunk2size* serta *sample-sample* yaitu: 64 61 74 61 00 0##1 00 00 00 00#2 24 17 1e f3#3 3c 13 3c 14 #4 16 f9 18 f9 34 e7 23 a6 3c f2 24 f2 24 f2 11 ce 1a 0d

Data *audio* yang di-*encoding* terdiri dari 576 baris frekuensi tiap *channel* dan bagian kecil yang disimpan sebagai 16 bit *signed integer*(Redmond, 1993). Sebagai contoh diberi sebuah spektrum frekuensi pada *file audio* berformat *wav* yang dapat dilihat pada Gambar 2.5.



Gambar 2.5. Frekuensi *File Audio* (Redmond, 1993)

Pada *file* suara yang terkuantisasi dilakukan *encoding* yang menghasilkan nilai integer yang merupakan nilai frekuensi dari sampel *audio* (Redmond, 1993). Sampel *audio* yang di-*encoding* seperti Gambar 2.6.

```
00 3e 1f 00 9a 00 1f 9a 00 3e 00 3f 00 3e 1f 00 5h 7d 00 -33 2f 5c 00
3e 1f 00 9a 00 1f 9f 00 3e 00 1f 00 3e 1f 00 5h 7d 02 -33 2f 5c 00 4d
2e 1f 00 9a 00 1f 9a 00 3e 00 1f 00 3e 1f 00 5h 7d 00 -33 2f 5c 00 3e
1f 00 9a 00 1f 9a 00 3e 00 1f 00 3e 1f 00 5h 7d 00 -23 2f 5c 00 3e 1f
12 9a 00 1f 9a 00 3e 00 1f 00 3e 1f 00 5h 7d 00 -33 2f 5c 00 3e 1f 00
7b 00 1b 9a 00 3e 00 1f 00 3e 1f 00 5h 7d 00 -33 2f 5c 00 3e 1f 00
9a 00 1f 9a 00 00 00 1f -12 3d 1f 00 5h 7d 00 -56 2f -01
```

Gambar 2.6. *Encoding Sample Audio* (Redmond, 1993)

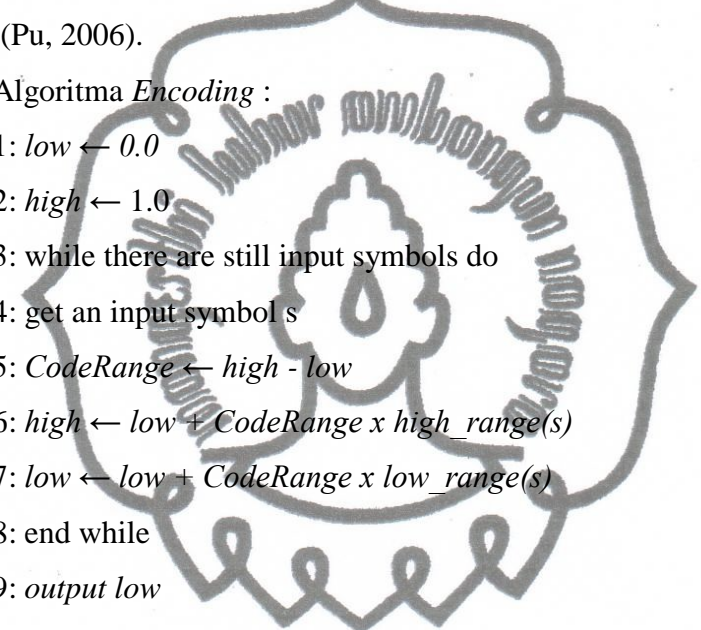
2.1.4 Algoritma *Arithmetic Coding*

Arithmetic coding adalah suatu bagian dari *entropy encoding* yang mengkonversi suatu data ke dalam bentuk data yang lain dengan lebih sering

menggunakan sedikit bit dan jarang menggunakan lebih banyak bit karakter. Teknik pengkodean ini memisahkan pesan masukan ke dalam simbol dan menukar masing – masing simbol dengan suatu *floating-point*. *Arithmetic Coding* mengkodekan seluruh pesan ke dalam suatu bilangan pecahan n di mana ($0.0 = n < 1.0$)

Berikut ini algoritma *encoding* dan algoritma *decoding*. Akan digunakan dua variabel *low* dan *high* untuk mendefenisikan interval [*low*, *high*). (Pu, 2006).

Algoritma *Encoding* :



```

1:  $low \leftarrow 0.0$ 
2:  $high \leftarrow 1.0$ 
3: while there are still input symbols do
4: get an input symbol  $s$ 
5:  $CodeRange \leftarrow high - low$ 
6:  $high \leftarrow low + CodeRange \times high\_range(s)$ 
7:  $low \leftarrow low + CodeRange \times low\_range(s)$ 
8: end while
9: output  $low$ 

```

Algoritma *Decoding* :

```

1: get encoded number
2: repeat
3: find symbol whose range covers the encoded number
4: output the symbol
5:  $CodeRange \leftarrow symbol\_high\_value - symbol\_low\_value$ 
6: subtract  $symbol\_low\_value$  from  $encoded\_number$ 
7: divide  $encoded\_number$  by range
8: until no more symbols

```

Algoritma *Arithmetic Coding* melakukan penggantian satu deretan simbol *input* dengan sebuah bilangan *floating point*. Semakin panjang dan semakin kompleks pesan yang dikodekan, akan semakin banyak bit yang diperlukan untuk keperluan tersebut. *Output* dari pengkodean *Arithmetic*

Coding adalah satu angka yang lebih kecil dari angka 1 dan lebih besar atau sama dengan 0. Angka ini secara unik dapat dapat didecoding sehingga menghasilkan deretan simbol yang dipakai untuk menghasilkan angka tersebut. Untuk menghasilkan angka *output* tersebut, tiap simbol yang akan di-*encode* diberi satu set nilai probabilitas. Contohnya andaikan pada sampel *audio* “00 3e 1f 00 9a 00 1f 9a 00 3e 00 1f 00 3e 1f” akan di-*encode*, maka akan diperoleh tabel probabilitas seperti ini :

Tabel 2.1. Tabel Probabilitas

Nilai	Frekuensi	Probabilitas
00	6	$6/15 = 0,4$
1f	4	$4/15 = 0,3$
9a	2	$2/15 = 0,1$
3e	3	$3/15 = 0,2$

Setelah probabilitas tiap karakter diketahui, tiap simbol/karakter akan diberikan *range* tertentu yang nilainya berkisar antara 0 dan 1, sesuai dengan probabilitas yang ada. Tidak ada ketentuan urutan penentuan segmen, asalkan antara *encoder* dan *decoder* melakukan hal yang sama. Selanjutnya akan diperoleh tabel *Range Probabilitas* seperti ini:

Tabel 2.2. *Range Probabilitas*

Nilai	Frekuensi	Probabilitas	Range
00	6	$6/15 = 0,4$	$0,0 \leq 00 < 0,4$
1f	4	$4/15 = 0,3$	$0,4 \leq 1f < 0,7$
9a	2	$2/15 = 0,1$	$0,7 \leq 9a < 0,8$
3e	3	$3/15 = 0,2$	$0,8 \leq 3e < 1,0$

Keterangan :

- $0,0 \leq 00 < 0,4$: Nilai “00” memiliki *range* dari 0,0 sampai dengan 0,4
- $0,4 \leq 1f < 0,7$: Nilai “1f” memiliki *range* dari 0,4 sampai dengan 0,7
- $0,7 \leq 9a < 0,8$: Nilai “9a” memiliki *range* dari 0,7 sampai dengan 0,8
- $0,8 \leq 3e < 1,0$: Nilai “3e” memiliki *range* dari 0,8 sampai dengan 1,0

commit to user

Satu hal yang perlu dicatat dari tabel ini adalah tiap karakter melingkupi *range* yang disebutkan, kecuali bilangan yang tinggi. Jadi, simbol “3e” sesungguhnya mempunyai *range* mulai dari 0,8 sampai dengan 0,9999... Selanjutnya, untuk melakukan proses *encoding*, algoritma berikut dipakai :

1. Set *low* = 0,0 (kondisi awal)
2. Set *high* = 1.0 (kondisi awal)
3. While (simbol input masih ada) do
4. Ambil simbol input.
5. $CR = high - low$.
6. $High = low + CR * high_range(simbol)$
7. $Low = low + CR * low_range(simbol)$
8. End while
9. Cetak *low*

Nilai *low*, *high* dan *CR* berturut-turut menyatakan batas bawah, batas atas dan jangkauan interval dari setiap simbol (*character*). Tahap awal *low* dan *high* di inisialisasi dengan nilai 0 dan 1 kemudian pada proses selanjutnya kedua nilai tersebut diperbaharui dengan rumus :

1. $High = low + CR * high_range(simbol)$
2. $Low = low + CR * low_range(simbol)$

Implementasi *Arithmetic Coding* harus memperhatikan kemampuan encoder dan decoder, yang umumnya mempunyai keterbatasan jumlah *mantissa*. Hal ini dapat menyebabkan kesalahan atau *error* apabila suatu *Arithmetic Coding* mempunyai kode dengan *floating point* yang sangat panjang (Amir, 2004). Sehingga diberikan solusi berupa modifikasi algoritma *Arithmetic Coding* dengan menggunakan bilangan *integer*.

Modifikasi ini mampu mengatasi keterbatasan pengolahan *floating point* dalam melakukan kompresi dan dekompresi data. Modifikasi dengan bilangan *integer* juga dipakai karena jumlah bit kodenya lebih sedikit dan mempercepat proses kompresi dan dekompresi data karena perhitungan *integer* jauh lebih cepat dari perhitungan *floating point* serta dapat diimplementasikan dalam program (Gozali dan Mervyn, 2004).

Dalam modifikasi ini, akan terjadi penukaran interval bilangan yang sebelumnya dalam 0 - 1 menjadi 0000h - FFFFh yang sebenarnya adalah sama. hal dapat dijelaskan pada contoh berikut. Misalkan diambil sebuah bilangan dan membaginya dengan FFFFh maka dapat dilihat sebagai berikut.

1. 0000h: $0 / 65535 = 0,0$
2. 4000h: $16384 / 65535 = 0,25$
3. 8000h: $32768 / 65535 = 0,5$
4. C000h: $49152 / 65535 = 0,75$
5. FFFFh: $65535 / 65535 = 1,0$

Kemudian dilakukan penyesuaian sehingga bit – bit yang diperlukan untuk pengolahan tidak di atas 16 bit (sebagai contoh 16 bit) dan sekarang telah diperoleh suatu interval yang baru. Untuk implementasi tersebut, langkah-langkahnya adalah sebagai berikut:

Algoritma *encoding* metode *Arithmetic Coding* dengan bilangan *integer* (Santoso, 2001) :

- Pertama ditetapkan nilai *high* dan *low*, sesuai dengan panjang register.
Untuk 16 bit *high* = \$FFFF dan *low* = \$0000
- Tetapkan nilai *code range* (CR). $CR = high - low + 1$
- While (simbol input masih ada) do
 - Ambil simbol input
 - $High = low + CR * high_range (simbol) - 1$
 - $Low = low + CR * low_range (simbol)$
 - $CR = high - low + 1$
 - Jika digit pertama sama maka SHIFT OUT digit pertama
- End While
- SHIFT OUT digit pertama jika masih dibutuhkan
- Cetak output

Algoritma *decoding* metode *Arithmetic Coding* dengan bilangan *integer* (Salomon dan Motta, 2010) :

- Pertama ambil *Encoded Symbol* (ES.)
- Tetapkan nilai *high* dan *low*, sesuai dengan nilai pada proses encoding.

- Tetapkan nilai code range (CR). $CR = high - low + 1$
- While (belum EOF dari simbol input(ES)) do
 - $Index \leftarrow ((ES - Low + 1) * Jml\ Frek - 1) / CR$
 - Cari range simbol yang melingkupi ES
 - $High = low + CR * high_range\ (simbol) - 1$
 - $Low = low + CR * low_range\ (simbol)$
 - Jika digit pertama sama maka SHIFT OUT digit pertama
 - $CR = high - low + 1$
- End While
- SHIFT OUT digit pertama jika masih dibutuhkan
- Cetak output

2.2 Penelitian Terkait

Berikut adalah uraian-uraian dari penelitian yang pernah ada mengenai metode kompresi *Arithmetic Coding*:

2.2.1 Analisis Perbandingan Kompresi Data dengan Teknik *Arithmetic Coding* dan *Run Length Encoding*

Penelitian ini menyimpulkan bahwa proses kompresi *file* dokumen menggunakan teknik *Arithmetic Coding* dihasilkan persentase rasio kompresi yang lebih baik dibandingkan dengan teknik *Run Length Encoding* yaitu sebesar 1,14 sampai 2,61 kali. Sedangkan perhitungan waktu untuk proses kompresi *file* dokumen menggunakan teknik *Arithmetic Coding* dihasilkan waktu yang lebih lama dibandingkan dengan teknik *Run length Encoding* yaitu sebesar 1,114 sampai 1,564 kali. Dan untuk waktu dekomposisi *file* dokumen menggunakan teknik *Arithmetic Coding* dihasilkan waktu yang lebih lama dibandingkan dengan teknik *Run Length Encoding* yaitu sebesar 2,85 sampai 5,635 kali.

Hasil analisis mengatakan bahwa untuk setiap teknik kompresi dan dekomposisi *file* dokumen menggunakan teknik *Arithmetic Coding* memiliki kelebihan dalam hal rasio kompresi tetapi membutuhkan waktu yang lebih lama. Sedangkan dengan menggunakan teknik *RunLength Encoding* memiliki rasio kompresi yang kurang tetapi membutuhkan waktu yang lebih cepat untuk proses

kompresi dan dekompresi dibandingkan dengan teknik *Arithmetic Coding* (Gozali dan Mervyn, 2004).

2.2.2 Aplikasi *Java Mobile* untuk Kompresi Layanan Pesan Singkat

Penelitian ini membahas pembuatan aplikasi kompresi teks untuk meningkatkan *jumlah* karakter pada setiap pengiriman pesan singkat (SMS) dengan mengimplementasikan algoritma *Arithmetic Coding*. Aplikasi ini tidak terintegrasi dengan aplikasi SMS bawaan pada *mobile phone* dan diimplementasikan pada *mobile phone* berbasis *Java* yang mendukung MIDP 2.0 dan CLDC 1.1. Tingkat keberhasilan proses kompresi pesan sangat dipengaruhi oleh penggunaan jenis huruf sesuai dengan tingkat probabilitas yang telah ditetapkan. Pengujian memperlihatkan peningkatan sebesar 16 karakter untuk pengiriman 1 SMS, 21 karakter pada pengiriman 2 SMS dan 37 karakter pada pengiriman 3 SMS.

Proses kompresi akan mengubah sebuah pesan menjadi sebuah kode tunggal. Kode tunggal ini merupakan deretan angka yang didapat dari hasil perhitungan berdasarkan probabilitas kemunculan setiap huruf yang sudah ditentukan sebelumnya. Pesan yang diketik pada *TextBox* akan diambil dan disimpan dalam sebuah *String* pesan dan akan diolah untuk mendapatkan kode tunggal sebelum dikirim ke penerima. Proses *encoding* atau kompresi ini menggunakan tabel statik yang sudah ditentukan. Dalam tabel tersebut terdapat 90 karakter yang mungkin muncul dalam sebuah pesan beserta dengan probabilitasnya. Karakter-karakter tersebut adalah karakter A-Z, a-z, 0-9, dan 28 karakter tambahan seperti @, &, * dan lain sebagainya. *String* pesan yang diketik pada *textbox* akan dihitung mulai dari karakter pertama sampai karakter terakhir dengan menggunakan nilai probabilitasnya. Nilai probabilitasnya berkisar antara angka 0 sampai 1 dengan range yang berbeda untuk tiap karakter. Pada perhitungan sampai pada karakter terakhir maka akan muncul sebuah kode tunggal. *Code Number* dalam tipe data *double* inilah yang akan dikirim ke penerima sebagai *String Message*.

commit to user

Proses dekompresi dilakukan dengan cara mengambil *String Message* yang dikirimkan dan memisahkan kode pertama yang menunjukkan banyaknya karakter dalam pesan dan kode kedua yang merupakan kode yang akan menunjukkan karakter-karakter pesan yang dikirim (Putro, Santoso, dan Basoeki, 2010).

2.2.3 Image Compression with Adaptive Arithmetic Coding

Penelitian ini membahas sebuah metode kompresi citra menggunakan transformasi *wavelet*, *zero tree coding* dan *adaptive arithmetic coding*. *Order 0* dari *adaptive arithmetic coding* dieksplorasi secara mendalam untuk meningkatkan rasio kompresi. Metode yang diusulkan menguraikan gambar ke dalam bentuk *subband* beberapa gambar menggunakan *discrete wavelet transform*, *decorrelated coefficients quantized* dengan menanamkan *zerotree wavelet algorithm* oleh Shapiro dan dikodekan menggunakan *adaptive arithmetic coding order 0*. Metode yang diusulkan menghasilkan rasio kompresi yang lebih baik sekaligus mengurangi waktu coding dibandingkan dengan *context based dynamic*. Hasil yang diperoleh sebanding dengan yang diperoleh menggunakan pendekatan *context modeling* (Sulthana dan Chandra, 2010).

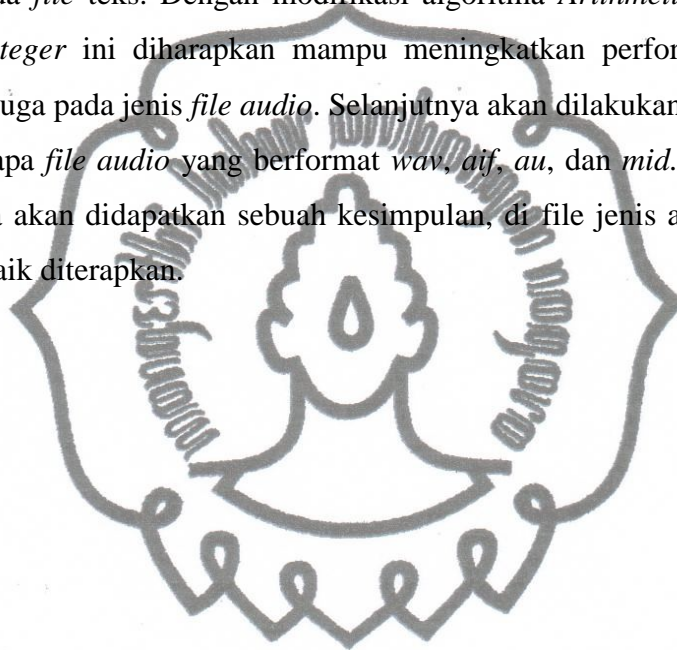
2.2.4 An Improved Bit-level Arithmetic Coding Algorithm

Penelitian ini menyajikan sebuah algoritma baru untuk lebih meningkatkan efisiensi implementasi *arithmetic coding*, yaitu *bit-level arithmetic coding* dengan menggunakan penambahan *integer* dan *shift*. Algoritma ini memiliki kompleksitas komputasi yang rendah dan lebih fleksibel, dan dengan demikian sangat cocok untuk desain hardware dan software. Algoritma baru ini memiliki kompleksitas yang minimal dan kecepatan tertinggi dalam algoritma *Zhao*, algoritma *RM*, algoritma *LR* dan algoritma *arithmetic coding* dasar. Kadang-kadang algoritma baru memiliki tingkat kompresi yang lebih tinggi dibandingkan dengan algoritma *arithmetic coding* dasar. Oleh karena itu, algoritma ini menyediakan sebuah pilihan yang sangat baik antara kinerja yang baik dan kompleksitas rendah (Zhang dan Ni, 2010).

2.3 Rencana Penelitian

commit to user

Penelitian yang akan dilakukan dalam tugas akhir ini adalah menerapkan cara kerja algoritma *Arithmetic Coding* dengan bilangan *integer* pada kompresi *file audio*. *File audio* yang akan dikompresi adalah *file audio* yang tidak terkompresi yang berformat *wav*, *aif*, *au*, dan *mid*. Kompresi dengan menggunakan *Arithmetic Coding* ini seperti pada penelitian yang dilakukan oleh Santoso (2001) pada teks dan oleh Gozali dan Mervyn (2004) dengan perbaikan algoritma, tetapi masih terbatas pada *file* teks. Dengan modifikasi algoritma *Arithmetic Coding* dengan bilangan *integer* ini diharapkan mampu meningkatkan performansi dan dapat diterapkan juga pada jenis *file audio*. Selanjutnya akan dilakukan proses pengujian pada beberapa *file audio* yang berformat *wav*, *aif*, *au*, dan *mid*. Tahap pengujian ini nantinya akan didapatkan sebuah kesimpulan, di file jenis apakah modifikasi ini paling baik diterapkan.



BAB III

METODOLOGI PENELITIAN

3.1 Studi Literatur

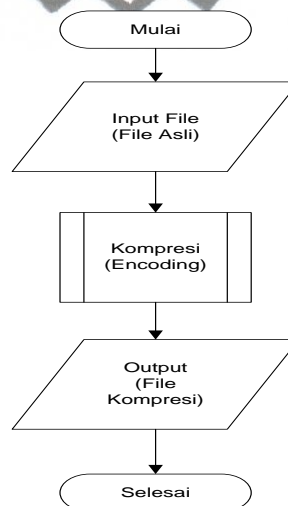
Metode ini dilaksanakan dengan melakukan studi kepustakaan yang relevan serta buku-buku maupun artikel-artikel yang bersumber dari *internet*.

3.2 Identifikasi Masalah

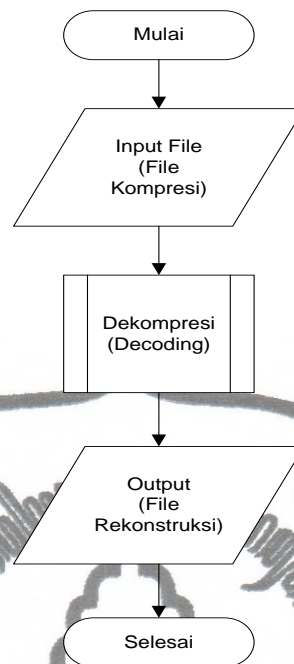
Identifikasi masalah yaitu tahap penentuan hal-hal penting seperti pengumpulan *file-file audio* dan pembacaan *header* dari setiap *file audio* tersebut sebagai dasar dari permasalahan yang dianalisis. Tahap ini merupakan tahap untuk mengkaji dan membatasi masalah yang akan diimplementasikan dalam sistem.

3.3 Tahapan Implementasi

Ada dua tahap proses utama yang akan dilakukan dalam implementasi penelitian ini. Yaitu tahap kompresi dan tahap dekompresi. Secara umum semua proses dalam kedua tahap itu sesuai dengan algoritma dari metode *Arithmetic Coding*. Gambar 3.1 adalah *flowchart* dari kedua tahapan tersebut.



Gambar 3.1. Proses Kompresi



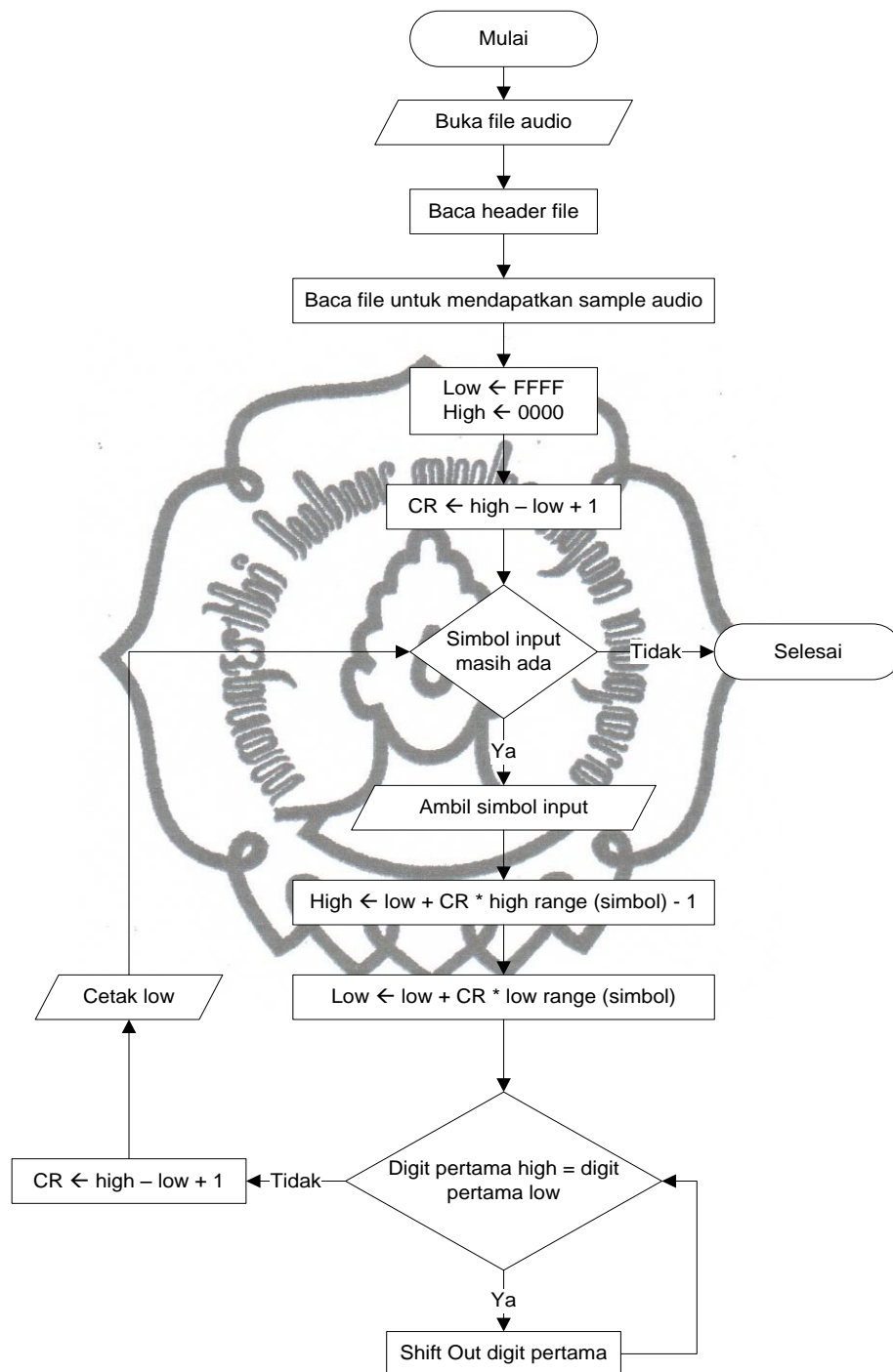
Gambar 3.2. Proses Dekompresi

3.3.1 Tahap Kompresi

Pada tahap ini, sebuah *file audio* akan dikompresi menggunakan algoritma *Arithmetic Coding* dengan bilangan *integer* sehingga akan menghasilkan sebuah *output file* yang berekstensi *aci* (*Arithmetic Coding* dengan bilangan *Integer*).

Arithmetic Coding menggantikan kode-kode *integer* untuk simbol-simbol individu dengan menetapkan satu kode yang panjang untuk keseluruhan *file* input. Metode ini dimulai dengan interval tertentu, kemudian membaca input *file* dari simbol ke simbol, kemudian menggunakan probabilitas dari setiap simbol untuk mempersempit interval. Pada tahap ini juga terjadi *Shift Out* digit pertama yaitu mengambil digit pertama yang sama pada nilai *High* dan *Low* dan dijadikan sebagai output, kemudian menambahkan nilai 0 dan 9 pada nilai *Low* dan *High*.

Gambar 3.3 adalah *flowchart* dari proses kompresi menggunakan algoritma *Arithmetic Coding* dengan bilangan *integer*.

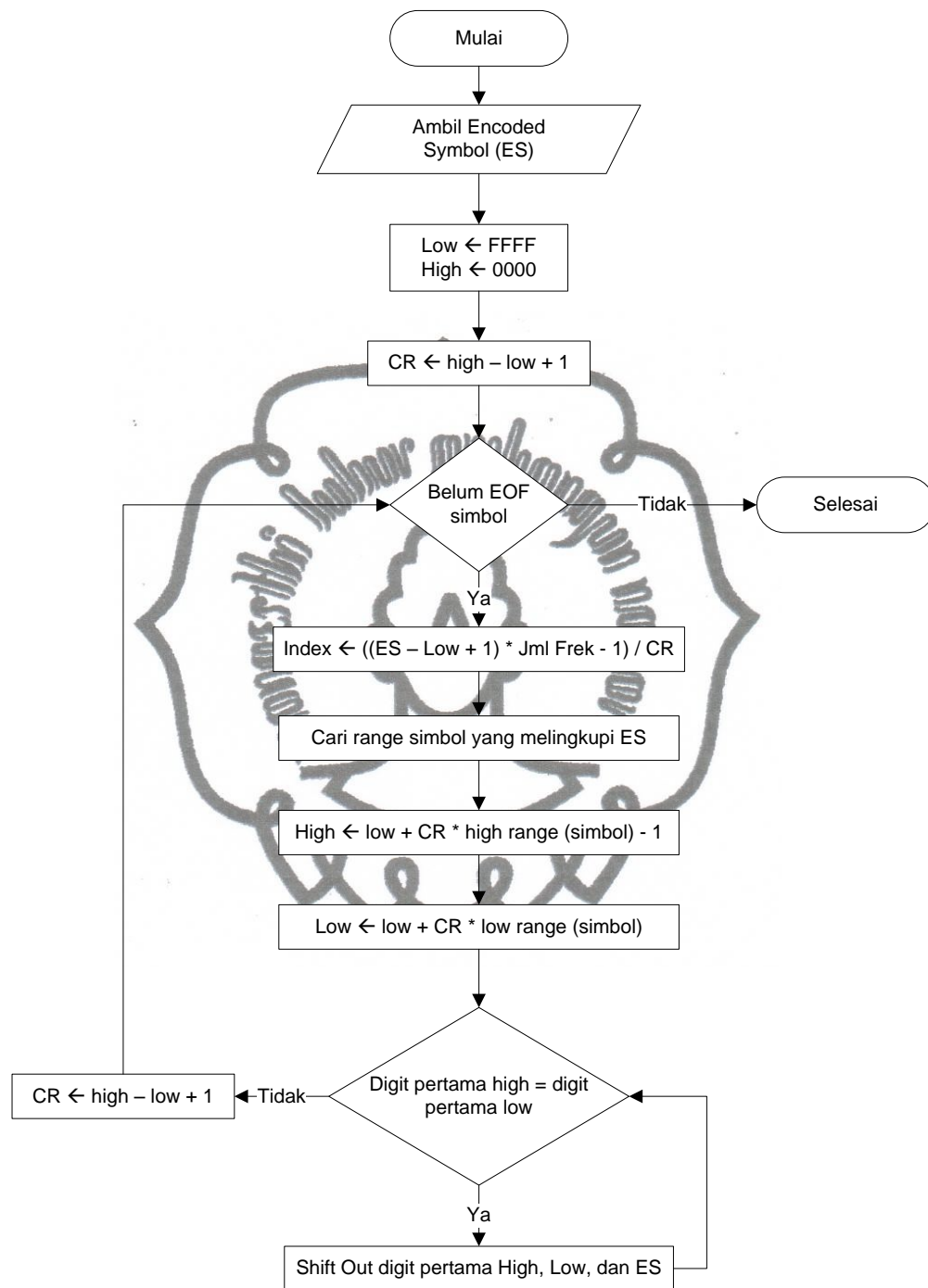


Gambar 3.3. *Flowchart* Proses Kompresi Menggunakan *Arithmetic Coding* dengan Bilangan *Integer*

3.3.2 Tahap Dekompresi

Tahap dekompresi adalah tahap mengembalikan data yang sudah di-*encoding* kembali lagi ke data awal sesuai dengan ukuran dan susunan-susunan datanya. Dalam tahap dekompresi ini akan dilakukan *decoding* terhadap suatu data yang sudah direpresentasikan sebagai sebuah simbol.

Pada tahap kompresi sebelumnya, sebuah inputan yang panjang akan menghasilkan simbol satu buah bilangan saja. Jadi, simbol yang hanya satu buah bilangan itu bisa jadi satu bilangan yang panjang atau bahkan panjang sekali. Misalkan sebuah *file* yang berukuran 1 MByte bisa di-*encoding* menjadi sebuah *file* yang berukuran 500 Kbyte yang hanya terdiri dari satu bilangan saja, sehingga untuk mengembalikan satu bilangan berukuran 500 Kbyte akan sangat kompleks dan membutuhkan waktu yang lama. Tahap ini dimulai dari mengambil simbol yang dihasilkan oleh tahap kompresi kemudian menentukan *range* simbol itu. Pada tahap ini, akan ditentukan nilai awal *Low* dan *High* seperti pada tahap kompresi. *Shift Out* digit pertama juga akan dilakukan dengan cara yang sama pada tahap kompresi. Perbedaan pada tahap dekompresi ini jika dibandingkan dengan tahap kompresi, ada sebuah rumus baru untuk menentukan *index*. *Index* nantinya yang akan berguna dalam menentukan *range* dari simbol. Untuk lebih jelasnya, bisa dilihat tahap dekompresi pada Gambar 3.4



Gambar 3.4. *Flowchart* Proses Dekompresi Menggunakan *Arithmetic Coding* dengan Bilangan *Integer*

3.4 Tahap Pengujian

Tahap ini akan membuktikan hipotesa awal, apakah hasil yang didapatkan sudah sesuai dengan yang diharapkan atau tidak. Tahap pengujian akan dilakukan dengan menghitung rasio kompresi, waktu kompresi, dan waktu dekompresi. *File* audio yang akan diujikan untuk proses kompresi menggunakan format *.wav*, *.aif*, *.au*, dan *.mid*. Sedangkan *file* yang akan dihasilkan dari proses kompresi berformat *.aci* (*Arithmetic Coding* dengan bilangan *Integer*). Pengujian akan dilakukan dengan skenario pengujian sebagai berikut :

1. *File* dengan format *.wav* dilakukan pengujian sebanyak 30 kali, dengan rincian :
 - Pengujian sebanyak 5 kali untuk file yang berukuran 0-1 MB
 - Pengujian sebanyak 5 kali untuk file yang berukuran 1-5 MB
 - Pengujian sebanyak 5 kali untuk file yang berukuran 5-10 MB
 - Pengujian sebanyak 5 kali untuk file yang memiliki satu jenis suara
 - Pengujian sebanyak 5 kali untuk file yang memiliki dua jenis suara
 - Pengujian sebanyak 5 kali untuk file yang memiliki beragam jenis suara
2. *File* dengan format *.aif* dilakukan pengujian sebanyak 15 kali
3. *File* dengan format *.au* dilakukan pengujian sebanyak 15 kali, dengan rincian :
 - Pengujian sebanyak 5 kali untuk file yang berukuran 0-1 MB
 - Pengujian sebanyak 5 kali untuk file yang berukuran 1-5 MB
 - Pengujian sebanyak 5 kali untuk file yang berukuran 5-10 MB
4. *File* dengan format *.mid* dilakukan pengujian sebanyak 15 kali
5. Pengujian kualitas hasil antara *file* sebelum dilakukan kompresi dan sesudah didekompresi sebanyak 1 kali untuk masing-masing jenis *file*.

File yang berformat *.aif*, *.au*, dan *.mid* hanya dilakukan pengujian masing-masing sebanyak 15 kali dikarenakan file jenis ini sulit dicari sesuai dengan pengelompokan ukurannya maupun jenis suaranya. Pengujian dilakukan dengan menghitung besarnya rasio dan waktu kompresi, kemudian dilanjutkan dengan menghitung besarnya waktu dekompresinya. Hipotesa awal menyebutkan bahwa

rasio kempresi, waktu kompresi, dan waktu dekompresi secara berturut-turut adalah 15 %, 4 detik, dan 2 detik. Untuk mendapatkan rasio kompresi, dapat menggunakan Rumus 3.1.

$$\text{Rasio Kompresi} = \frac{(\text{ukuran file asli} - \text{ukuran file terkompresi})}{\text{ukuran file asli}} \times 100\% \quad \dots 3.1$$



BAB IV

HASIL DAN PEMBAHASAN

4.1 Spesifikasi Perangkat

Perangkat yang digunakan untuk mengimplementasikan rancangan yang telah dibuat meliputi perangkat lunak dan perangkat keras, yaitu :

1. Ruang Lingkup Perangkat Lunak
 - *Microsoft Visual Basic 6*
 - *Sistem Operasi Windows 7*
2. Ruang Lingkup Perangkat Keras
 - *Intel Core i3, 2,2 Ghz*
 - *RAM 2048 MB*

4.2 Hasil Implementasi

Aplikasi yang dibangun memiliki 4 menu utama. Yaitu Menu *Kompresi/Dekompresi*, *Help*, *About*, dan *Quit*.

Menu *Kompresi/Dekompresi*

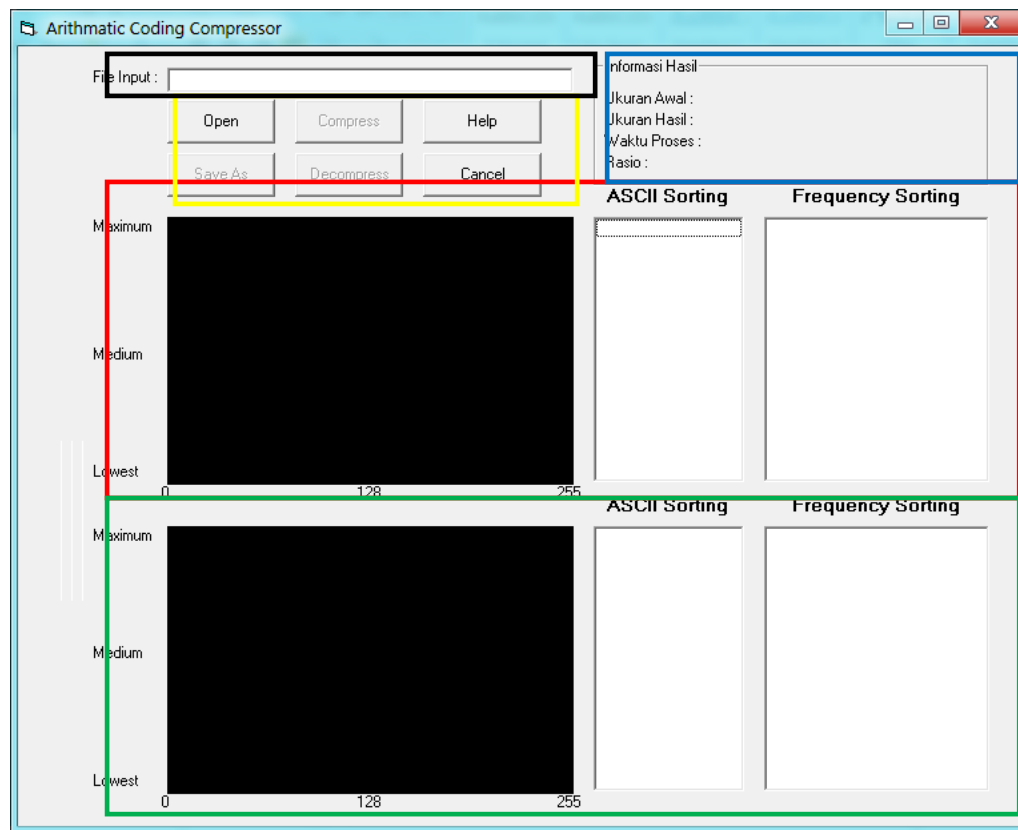
- a. *Input File* (yang diberi kotak warna hitam)

Digunakan untuk mengetahui inputan *file* beserta *path*-nya di dalam komputer.

- b. Bagian Tombol-tombol (yang diberi kotak warna kuning)

Bagian ini terdiri dari tombol *Open*, *Save As*, *Compress*, *Decompress*, *Help*, dan *Cancel*. *Open* berguna untuk membuka *file* inputan, sedangkan *Save As* untuk menyimpan *file* hasil kompresi. *Compress* berguna untuk mengompresi *file* sedangkan *Decompress* berguna untuk mendekompresi *file*. *Help* berguna untuk melihat bantuan cara kompresi/dekompresi *file*, dan *Cancel* untuk kembali ke menu utama.

commit to user



Gambar 4.1. Tampilan Menu Kompresi/Dekompresi

- c. Bagian Informasi Hasil (yang diberi kotak warna biru)

Bagian ini terdiri dari Ukuran Awal, Ukuran Hasil, Waktu Proses, dan Rasio.

- d. Bagian Statistik (yang diberi kotak warna merah dan hijau)

Terdiri dari dua bagian yaitu bagian statistik *file* input dan bagian statistik *file* hasil. Bagian statistik *file* input yang berada di bagian atas (kotak warna merah), sedangkan bagian statistik *file* hasil yang berada di bagian bawah (kotak warna hijau).

4.3 Hasil Pengujian dan Analisa

Pengujian dilakukan dengan cara menghitung rasio kompresi, waktu kompresi, dan waktu dekompresi. *File audio* yang akan diujikan untuk proses kompresi menggunakan format *.wav*, *.aif*, *.au*, dan *.mid*. Proses kompresi akan menghasilkan *file* yang berformat *aci* (*Arithmetic Coding* dengan bilangan *integer*), yang nantinya akan digunakan dalam proses dekompresi. Pengujian dilakukan sebanyak 75 kali, dengan rincian sebagai berikut :

1. *File* dengan format *.wav* dilakukan pengujian sebanyak 30 kali, dengan rincian :
 - Pengujian sebanyak 5 kali untuk file yang berukuran 0-1 MB
 - Pengujian sebanyak 5 kali untuk file yang berukuran 1-5 MB
 - Pengujian sebanyak 5 kali untuk file yang berukuran 5-10 MB
 - Pengujian sebanyak 5 kali untuk file yang memiliki satu jenis suara
 - Pengujian sebanyak 5 kali untuk file yang memiliki dua jenis suara
 - Pengujian sebanyak 5 kali untuk file yang memiliki beragam jenis suara
2. *File* dengan format *.aif* dilakukan pengujian sebanyak 15 kali
3. *File* dengan format *.au* dilakukan pengujian sebanyak 15 kali, dengan rincian :
 - Pengujian sebanyak 5 kali untuk file yang berukuran 0-1 MB
 - Pengujian sebanyak 5 kali untuk file yang berukuran 1-5 MB
 - Pengujian sebanyak 5 kali untuk file yang berukuran 5-10 MB
4. *File* dengan format *.mid* dilakukan pengujian sebanyak 15 kali
5. Pengujian kualitas hasil antara *file* sebelum dilakukan kompresi dan sesudah didekompresi sebanyak 1 kali untuk masing-masing jenis *file*.

4.3.1 Hasil Pengujian dan Analisa Rasio dan Waktu Kompresi

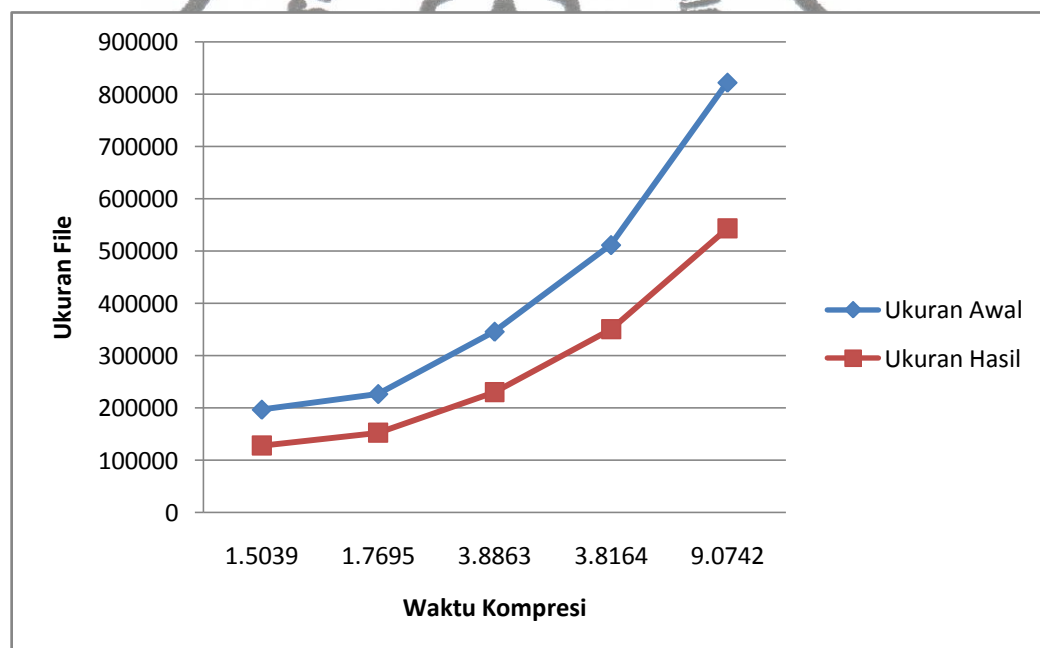
Untuk menghitung rasio kompresi dapat menggunakan Rumus 3.1 yang ada pada BAB III. Rumus 4.1 adalah rumus untuk menghitung rata-rata rasio kompresi.

$$\text{Rata – rata rasio kompresi} = \frac{\text{Total Rasio Kompresi}}{\text{Banyaknya Pengujian}} \quad \dots 4.1$$

4.3.1.1 Pengujian Kompresi *File Audio* yang Berformat *.wav*

Tabel 4.1. Rasio Kompresi untuk *File* Berformat *.wav* yang Berukuran 0 – 1 MB

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio1.wav	196761	128208	1.5039	34.84	0.0000076
File Audio2.wav	226670	153053	1.7695	32.48	0.0000078
File Audio3.wav	345884	230438	3.8863	33.38	0.0000112
File Audio4.wav	511530	350150	3.8164	31.55	0.0000075
File Audio5.wav	822188	543338	9.0742	33.92	0.0000110
Rata-rata				33.234	0.0000090

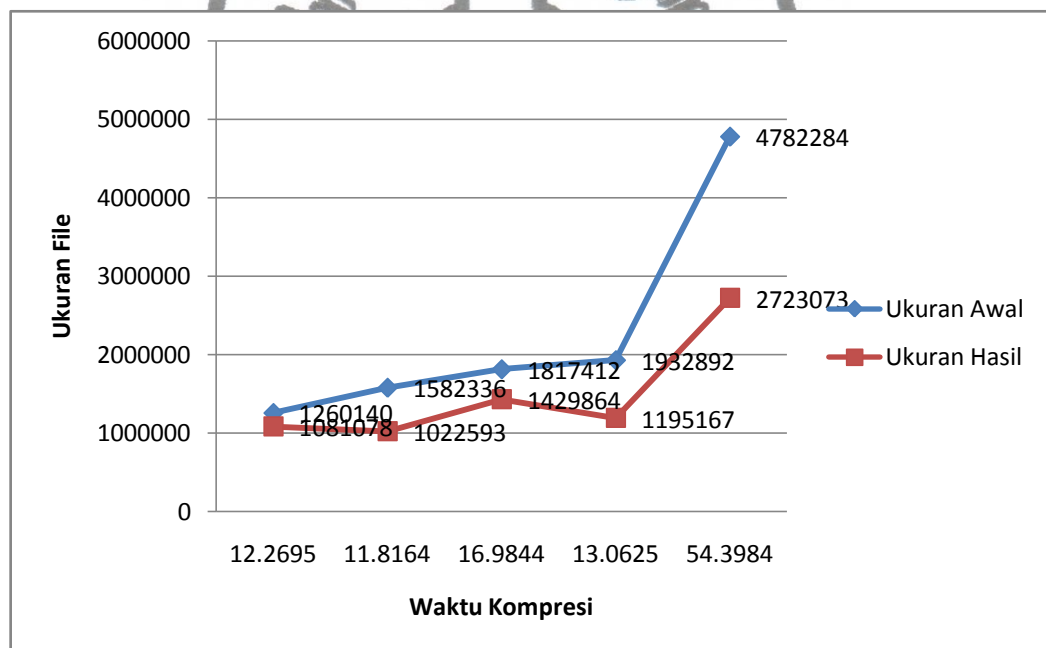


Gambar 4.2. Grafik Kompresi *File* Berformat *.wav* yang Berukuran 0 – 1 MB

Berdasarkan Waktu Kompresi Terhadap Ukuran *File*

Tabel 4.2. Rasio Kompresi untuk *File* Berformat .wav yang Berukuran 1 – 5 MB

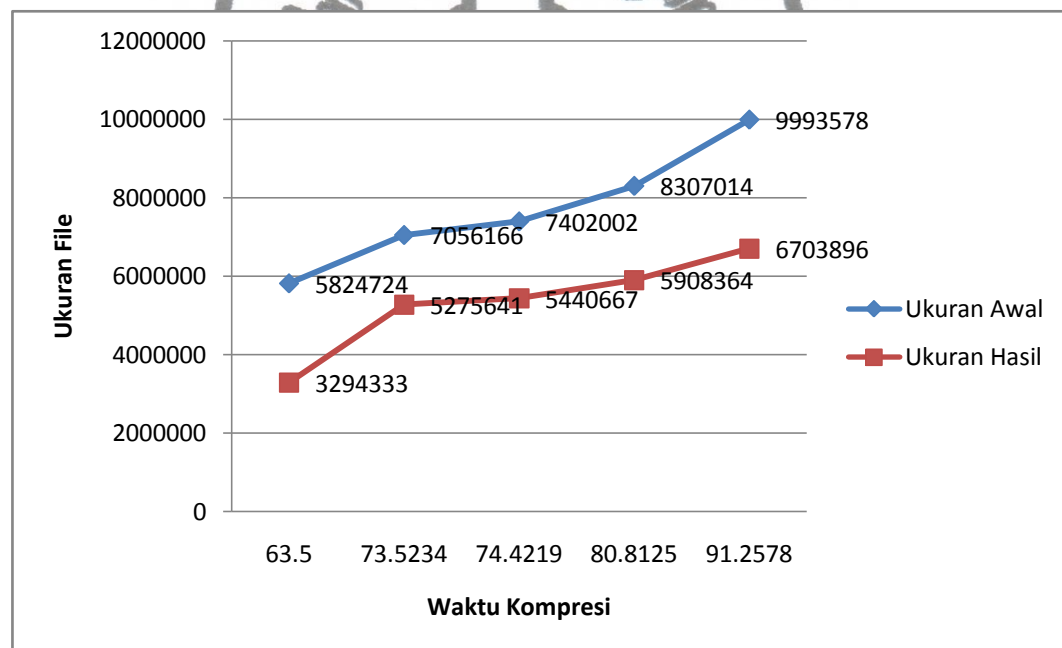
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio6.wav	1267904	604054	7.5156	52.36	0.0000059
File Audio7.wav	1582336	1022593	11.8164	35.37	0.0000075
File Audio8.wav	1817412	1429864	16.9844	21.32	0.0000094
File Audio9.wav	1932892	1195167	13.0625	38.17	0.0000068
File Audio10.wav	4782284	2723073	54.3984	43.06	0.0000114
Rata-rata				38.056	0.0000081



Gambar 4.3. Grafik Kompresi *File* Berformat .wav yang Berukuran 1 – 5 MB
Berdasarkan Waktu Kompresi Terhadap Ukuran *File*

Tabel 4.3. Rasio Kompresi untuk *File* Berformat .wav yang Berukuran 5 – 10 MB

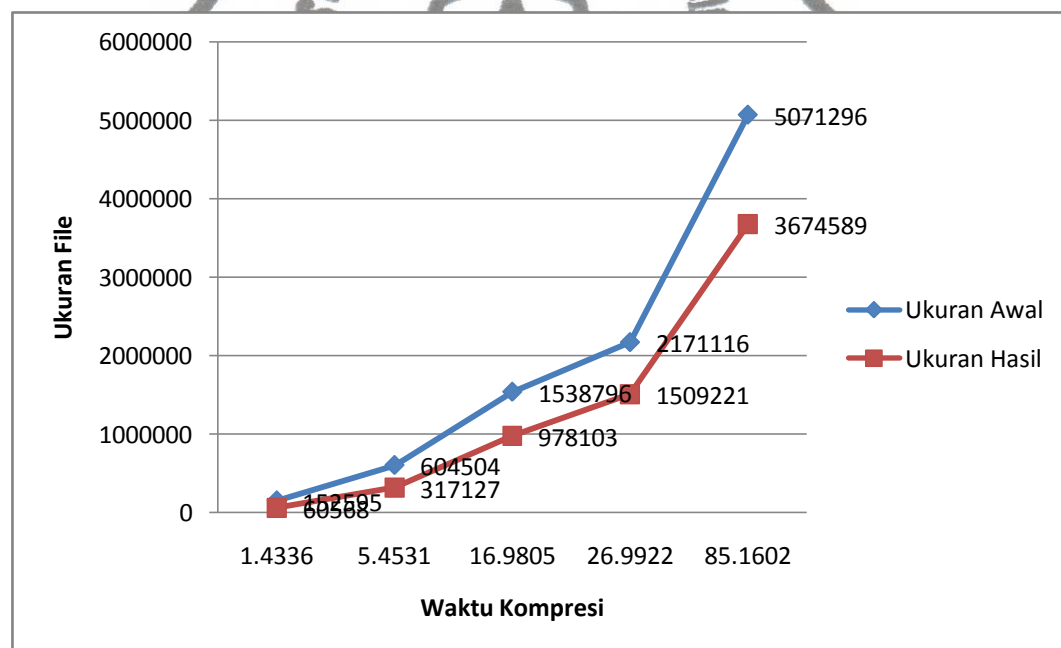
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio11.wav	5824724	3294333	63.5000	43.44	0.0000109
File Audio12.wav	7056166	5275641	73.5234	25.23	0.0000104
File Audio13.wav	7402002	5440667	74.4219	26.50	0.0000101
File Audio14.wav	8307014	5908364	80.8125	28.87	0.0000097
File Audio15.wav	9993578	6703896	91.2578	32.92	0.0000091
Rata-rata				31.392	0.0000100



Gambar 4.4. Grafik Kompresi *File* Berformat .wav yang Berukuran 5 – 10 MB
Berdasarkan Waktu Kompresi Terhadap Ukuran *File*

Tabel 4.4. Rasio Kompresi untuk *File* Berformat .wav yang Memiliki Satu Jenis Suara

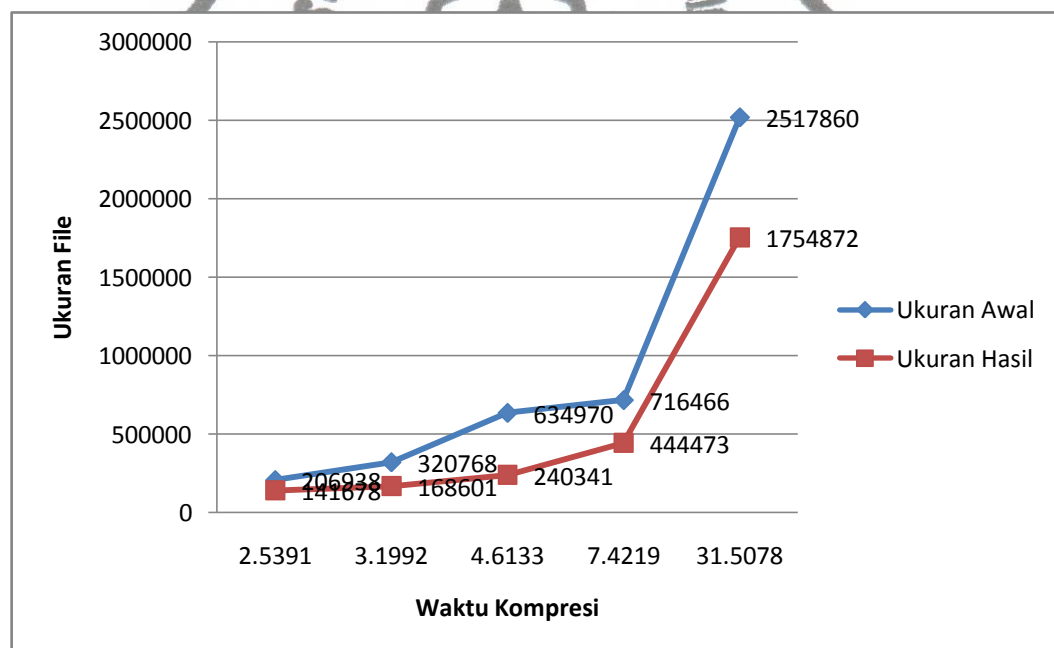
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio16.wav	152595	60568	1.4336	60.31	0.0000094
File Audio17.wav	604504	317127	5.4531	47.54	0.0000090
File Audio18.wav	1538796	978103	16.9805	36.44	0.0000110
File Audio19.wav	2171116	1509221	26.9922	30.49	0.0000124
File Audio20.wav	5071296	3674589	85.1602	27.54	0.0000168
Rata-rata				40.464	0.0000117



Gambar 4.5. Grafik Kompresi *File* Berformat .wav yang Memiliki Satu Jenis Suara Berdasarkan Waktu Kompresi Terhadap Ukuran *File*

Tabel 4.5. Rasio Kompresi untuk *File* Berformat .wav yang Memiliki Dua Jenis Suara

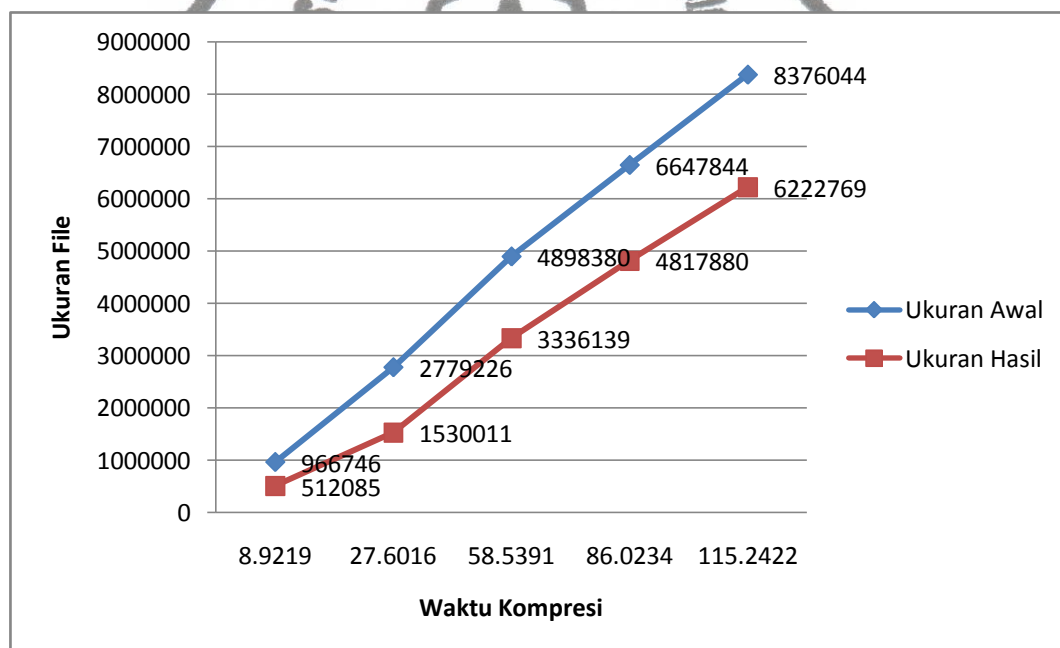
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio21.wav	206938	141678	2.5391	31.54	0.0000123
File Audio22.wav	320768	168601	3.1992	47.44	0.0000100
File Audio23.wav	634970	240341	4.6133	62.15	0.0000073
File Audio24.wav	716466	444473	7.4219	37.96	0.0000104
File Audio25.wav	2517860	1754872	31.5078	30.30	0.0000125
Rata-rata				41.878	0.0000105



Gambar 4.6. Grafik Kompresi *File* Berformat .wav yang Memiliki Dua Jenis Suara Berdasarkan Waktu Kompresi Terhadap Ukuran *File*

Tabel 4.6. Rasio Kompresi untuk *File* Berformat .wav yang Memiliki Beragam Jenis Suara

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio26.wav	966746	512085	8.9219	47.03	0.0000092
File Audio27.wav	2779226	1530011	27.6016	44.95	0.0000099
File Audio28.wav	4898380	3336139	58.5391	31.89	0.0000120
File Audio29.wav	6647844	4817880	86.0234	27.53	0.0000129
File Audio30.wav	8376044	6222769	115.2422	25.71	0.0000138
Rata-rata				35.422	0.0000116



Gambar 4.7. Grafik Kompresi *File* Berformat .wav yang Memiliki Beragam Jenis Suara Berdasarkan Waktu Kompresi Terhadap Ukuran *File*

Berdasarkan rata-rata rasio kompresi dari Tabel 4.1 sampai dengan Tabel 4.6, maka dapat dihitung rata-rata rasio kompresi keseluruhan *file .wav* yang ditunjukkan pada Tabel 4.7.

Tabel 4.7. Rata-rata Rasio Kompresi untuk Keseluruhan *File* yang Berformat *.wav*

Tabel	Rasio Kompresi (%)
4.1	33.234
4.2	38.056
4.3	31.392
4.4	40.464
4.5	41.878
4.6	35.422
Total	220.446
Rata-rata	36.741

Berdasarkan Tabel 4.4, Tabel 4.5, dan Tabel 4.6 dapat diketahui bahwa jenis suara yang beragam tidak mempengaruhi besarnya rasio kompresi. Sedangkan perbedaan besarnya ukuran file juga tidak secara signifikan mempengaruhi besarnya rasio kompresi, hal ini dapat kita lihat dari Tabel 4.1, Tabel 4.2, dan Tabel 4.3. Dari tabel-tabel tersebut, di ukuran file 1 – 5 MB rasio kompresi mempunyai rata-rata yang besar jika dibandingkan dengan file yang berukuran 0 – 1 MB dan 5 – 10 MB.

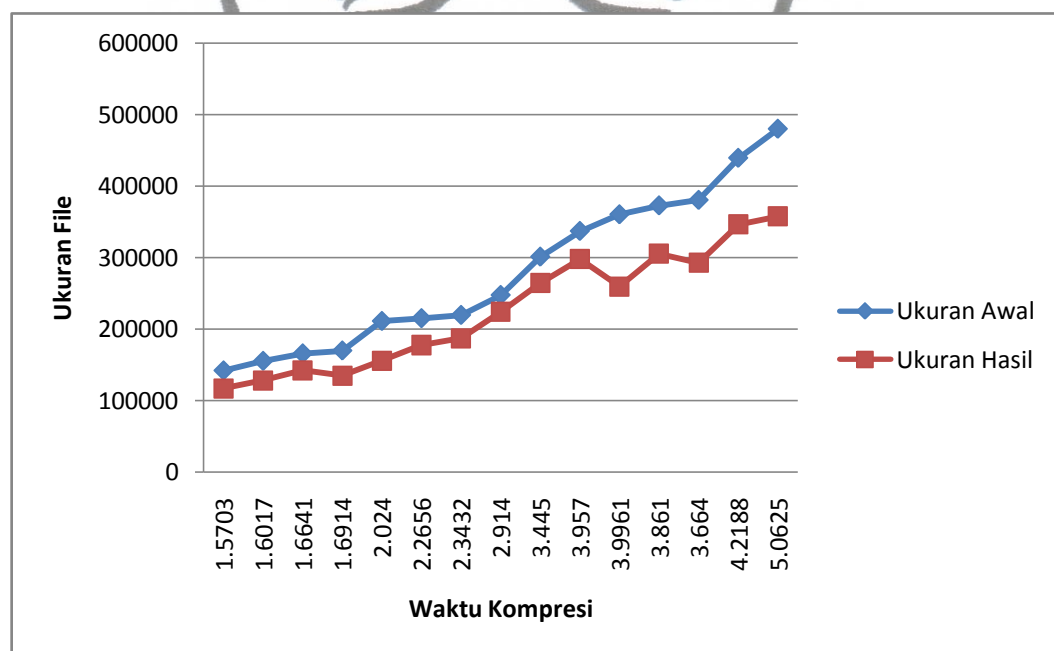
4.3.1.2 Pengujian Kompresi *File Audio* yang Berformat *.aif*

Tabel 4.8. Rasio Kompresi untuk *File* Berformat *.aif*

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio31.aif	142002	117007	1.5703	17.60	0.0000111
File Audio32.aif	155354	128286	1.6017	17.42	0.0000103
File Audio33.aif	165842	142565	1.6641	14.04	0.0000100
File Audio34.aif	169902	135124	1.6914	20.47	0.0000100
File Audio35.aif	211210	156065	2.0240	26.11	0.0000096
File Audio36.aif	215122	177766	2.2656	17.37	0.0000105

Tabel 4.8. Lanjutan

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio37.aif	219634	187264	2.3432	14.74	0.0000107
File Audio38.aif	247578	224085	2.9140	9.49	0.0000118
File Audio39.aif	301226	264393	3.4450	12.23	0.0000114
File Audio40.aif	337170	298294	3.9570	11.53	0.0000117
File Audio41.aif	360454	259621	3.9961	27.97	0.0000111
File Audio42.aif	372922	305493	3.8610	18.08	0.0000104
File Audio43.aif	380574	293089	3.6640	22.99	0.0000096
File Audio44.aif	439458	346660	4.2188	21.12	0.0000096
File Audio45.aif	480162	357819	5.0625	25.48	0.0000105
Rata-rata				18.443	0.0000106

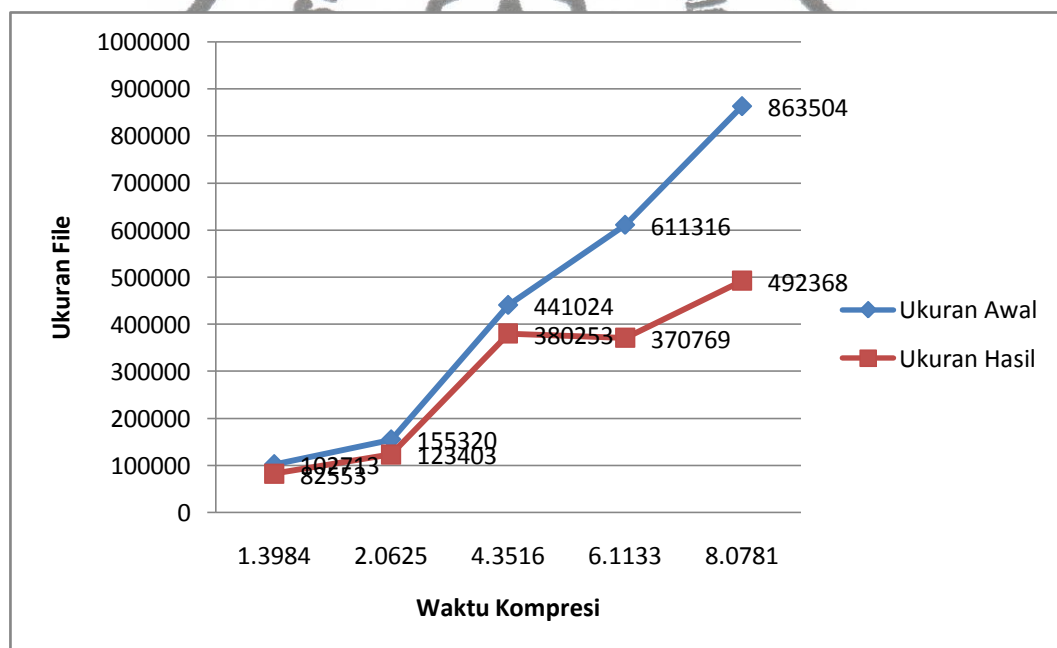


Gambar 4.8. Grafik Kompresi *File* Berformat *.aif* Berdasarkan Waktu Kompresi Terhadap Ukuran *File*

4.3.1.3 Pengujian Kompresi File Audio yang Berformat .au

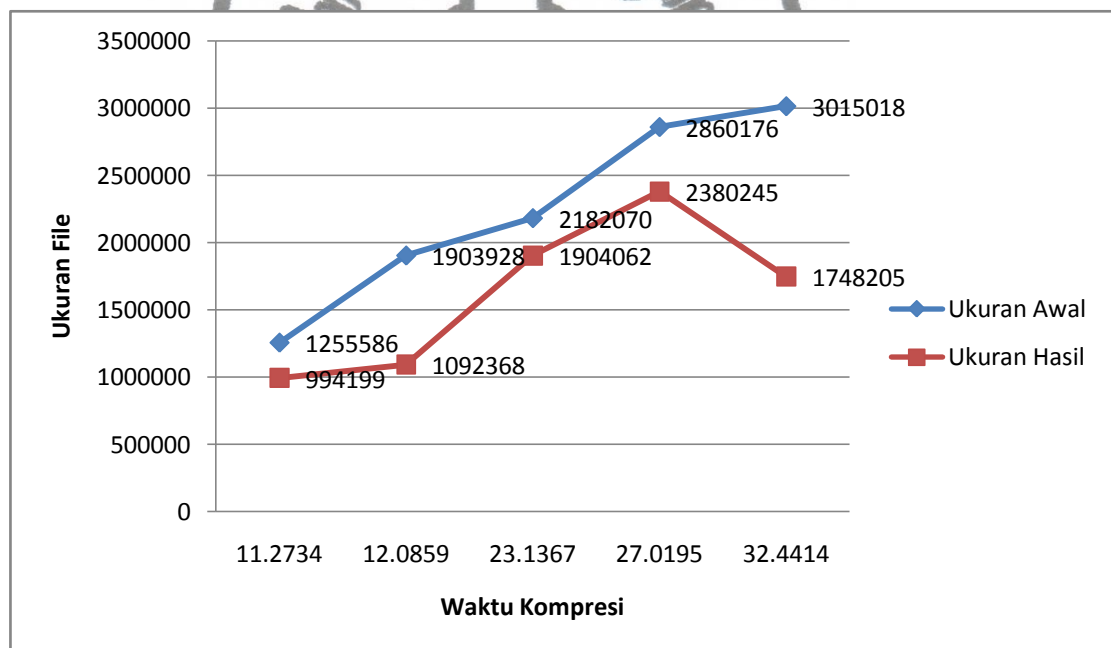
Tabel 4.9. Rasio Kompresi untuk File Berformat .au yang Berukuran 0 – 1 MB

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio46.au	102713	82553	1.3984	19.63	0.0000136
File Audio47.au	155320	123403	2.0625	20.55	0.0000133
File Audio48.au	441024	380253	4.3516	13.78	0.0000099
File Audio49.au	611316	370769	6.1133	39.35	0.0000100
File Audio50.au	863504	492368	8.0781	42.98	0.0000094
Rata-rata				27.258	0.0000112



Tabel 4.10. Rasio Kompresi untuk *File* Berformat *.au* yang Berukuran 1 – 5 MB

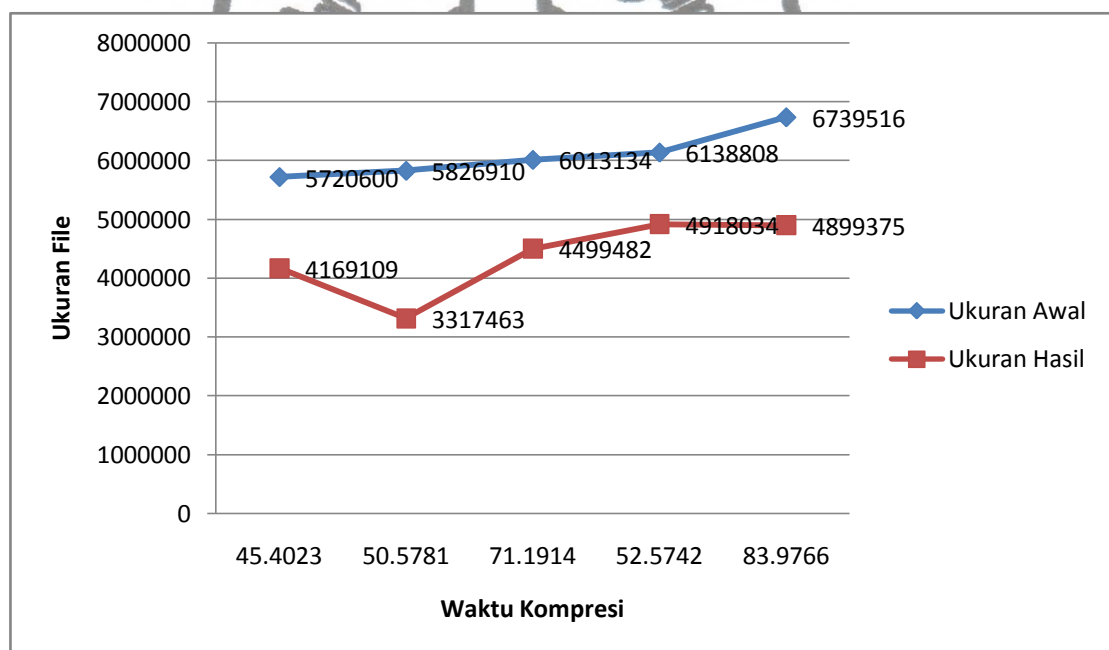
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio51.au	1255586	994199	11.2734	20.82	0.0000090
File Audio52.au	1903928	1092368	12.0859	42.63	0.0000064
File Audio53.au	2182070	1904062	23.1367	12.74	0.0000106
File Audio54.au	2860176	2380245	27.0195	16.78	0.0000095
File Audio55.au	3015018	1748205	32.4414	42.02	0.0000108
Rata-rata				26.998	0.0000092



Gambar 4.10. Grafik Kompresi *File* Berformat *.au* yang Berukuran 1 – 5 MB
Berdasarkan Waktu Kompresi Terhadap Ukuran *File*

Tabel 4.11. Rasio Kompresi untuk *File* Berformat *.au* yang Berukuran 5 – 10 MB

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio56.au	5720600	4169109	45.4023	27.12	0.0000079
File Audio57.au	5826910	3317463	50.5781	43.07	0.0000087
File Audio58.au	6013134	4499482	71.1914	25.17	0.0000118
File Audio59.au	6138808	4918034	52.5742	19.89	0.0000086
File Audio60.au	6739516	4899375	83.9766	27.30	0.0000125
Rata-rata				28.510	0.0000099



Gambar 4.11. Grafik Kompresi *File* Berformat *.au* yang Berukuran 5 – 10 MB
Berdasarkan Waktu Kompresi Ukuran *File*

Berdasarkan rata-rata rasio kompresi dari Tabel 4.9 sampai dengan Tabel 4.11, maka dapat dihitung rata-rata rasio kompresi keseluruhan *file .au* yang ditunjukkan pada Tabel 4.12.

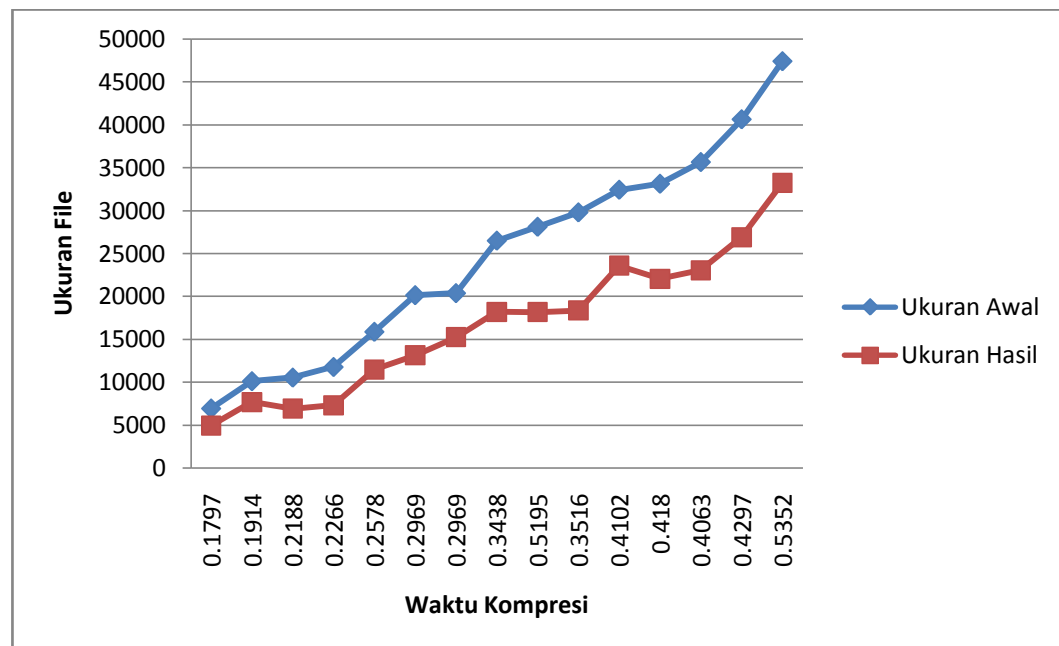
Tabel 4.12. Rata-rata Rasio Kompresi untuk Keseluruhan *File* yang Berformat *.au*

Tabel	Rasio Kompresi (%)
4.9	27.258
4.10	26.998
4.11	28.510
Total	82.766
Rata-rata	27.589

4.3.1.4 Pengujian Kompresi *File Audio* yang Berformat *.mid*

Tabel 4.13 Rasio Kompresi untuk *File* Berformat *.mid*

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Kompresi (s)	Rasio (%)	Waktu Kompresi/ Byte
File Audio61.mid	6960	4957	0.1797	28.78	0.0000258
File Audio62.mid	10133	7702	0.1914	23.99	0.0000189
File Audio63.mid	10556	6917	0.2188	34.37	0.0000207
File Audio64.mid	11789	7326	0.2266	37.86	0.0000192
File Audio65.mid	15886	11504	0.2578	27.49	0.0000162
File Audio66.mid	20151	13134	0.2969	34.82	0.0000147
File Audio67.mid	20387	15250	0.2969	25.20	0.0000146
File Audio68.mid	26500	18207	0.3438	31.29	0.0000130
File Audio69.mid	28115	18182	0.5195	35.33	0.0000185
File Audio70.mid	29793	18365	0.3516	38.36	0.0000118
File Audio71.mid	32426	23563	0.4102	27.33	0.0000127
File Audio72.mid	33123	22050	0.4180	33.43	0.0000126
File Audio73.mid	35652	23043	0.4063	35.37	0.0000114
File Audio74.mid	40629	26911	0.4297	33.76	0.0000106
File Audio75.mid	47391	33216	0.5352	29.91	0.0000113
Rata-rata				31.819	0.0000155



Gambar 4.12. Grafik Kompresi *File* Berformat *.mid* Berdasarkan Waktu Kompresi Terhadap Ukuran *File*

Berdasarkan Tabel 4.1 sampai dengan Tabel 4.13, bisa dihitung rata-rata rasio kompresi dan rata-rata waktu kompresi/byte untuk keseluruhan jenis *file* yang diujikan.

Tabel 4.14. Rata-rata Rasio Kompresi

	Rasio Kompresi (%)
Rata-rata file .wav	36.741
Rata-rata file .aif	18.443
Rata-rata file .au	27.589
Rata-rata file .mid	31.819
Total	114.592
Rata-rata	28.648

Tabel 4.15. Rata-rata Waktu Kompresi/Byte untuk File yang Berukuran 0 – 1 MB

No	Waktu Kompresi/Byte
Rata-rata file .wav	0.0000090
Rata-rata file .aif	0.0000106
Rata-rata file .au	0.0000112
Rata-rata file .mid	0.0000155
Total	0.0000463
Rata-rata	0.0000116

Tabel 4.16. Rata-rata Waktu Kompresi/Byte untuk File yang Berukuran 1 – 5 MB

No	Waktu Kompresi/Byte
Rata-rata file .wav	0.0000082
Rata-rata file .au	0.0000092
Total	0.0000174
Rata-rata	0.0000087

Tabel 4.17. Rata-rata Waktu Kompresi/Byte untuk File yang Berukuran 5 –10 MB

No	Waktu Kompresi/Byte
Rata-rata file .wav	0.0000100
Rata-rata file .au	0.0000099
Total	0.0000199
Rata-rata	0.0000100

Berdasarkan Tabel 4.14 diketahui bahwa rata-rata rasio kompresi untuk keseluruhan jenis *file* yang diujikan adalah 26.648 %. Dilihat dari Tabel 4.15, Tabel 4.16, dan Tabel 4.17, rata-rata waktu kompresi/byte untuk semua ukuran file hampir mempunyai rata-rata yang sama. Meskipun waktu kompresi dari setiap jenis *file* naik secara linier sesuai dengan ukuran *file*, tetapi waktu kompresi tiap byte-nya hampir sama untuk semua ukuran file. Semisal rata-rata waktu kompresi/byte yang berukuran 0 – 1 MB diambil kemudian dikalikan dengan sebuah file yang berukuran 500 KB, maka, Waktu Kompresi = $0.0000116 \times 500000 = 5.8 \text{ s}$

Waktu kompresi dikalikan dengan sebuah file yang berukuran 500 KB karena waktu kompresi pada hipotesa awal didasarkan atas waktu kompresi dengan ukuran file maksimum 500 KB. Pada hipotesa awal rata-rata rasio kompresi sebesar 15 % dan waktu kompresi sebesar 2 s, hal ini menunjukkan bahwa hasil akhir rasio kompresi telah sesuai dengan yang diharapkan, sedangkan waktu kompresi masih belum sesuai dengan yang diharapkan. Dari keseluruhan Tabel 4.1 sampai Tabel 4.14 dapat disimpulkan sebagai berikut :

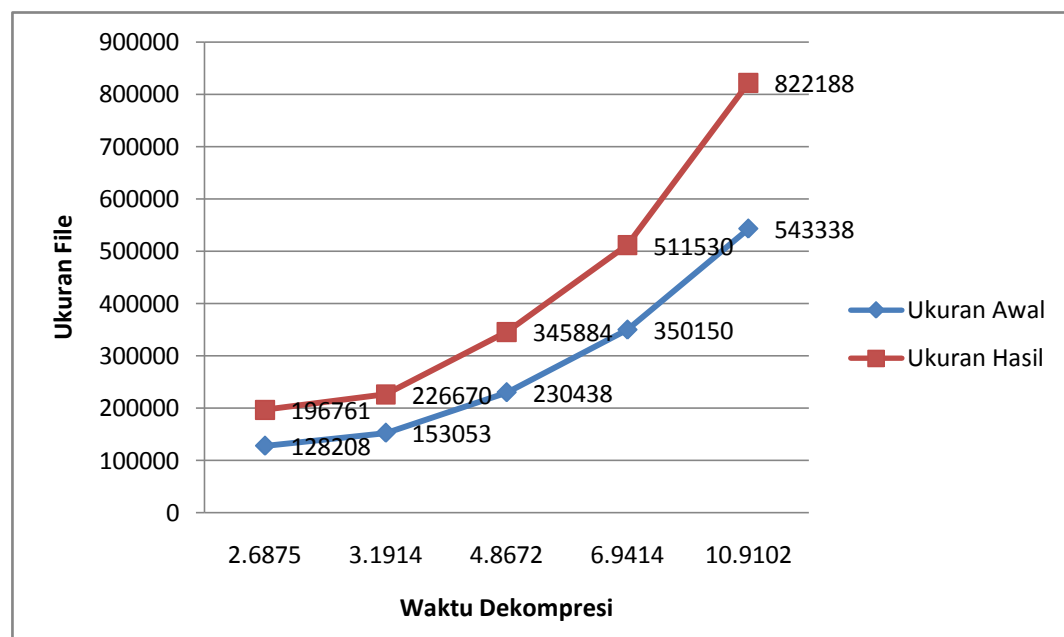
1. Rasio kompresi pada setiap jenis *file* yang diujikan berbeda (sesuai dengan Tabel 4.14), hal ini tergantung oleh banyaknya kesamaan nilai-nilai atau komponen-komponen yang ada pada file tersebut.
2. Waktu kompresi yang dihasilkan pada setiap jenis *file* tergantung oleh besarnya ukuran *file*, kondisi hardware dan susunan datanya.
3. Rata-rata rasio kompresi terbesar adalah berturut-turut pada *file* yang berformat *.wav*, *.mid*, *.au*, dan *.aif*.

4.3.2 Hasil Pengujian dan Analisa Waktu Dekompresi

4.3.2.1 Pengujian Dekompresi *File Audio* yang Berformat *.wav*

Tabel 4.18. Waktu Dekompresi *File* Berformat *.wav* yang Ukuran Awalnya 0 – 1 MB

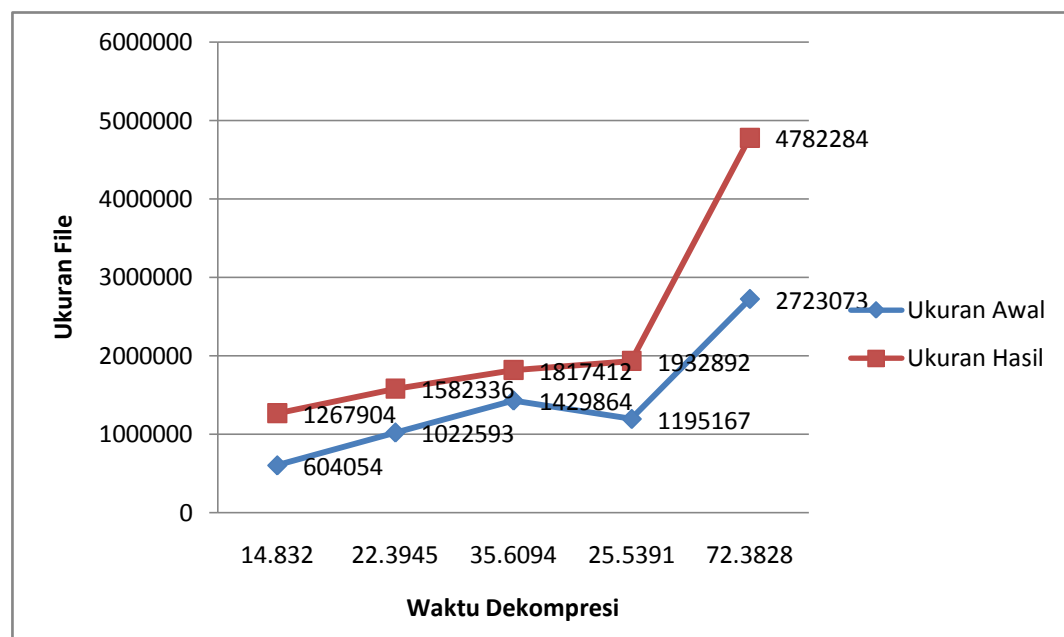
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi/ Byte
File Audio1.aci	128208	196761	2.6875	0.0000210
File Audio2.aci	153053	226670	3.1914	0.0000209
File Audio3.aci	230438	345884	4.8672	0.0000211
File Audio4.aci	350150	511530	6.9414	0.0000198
File Audio5.aci	543338	822188	10.9102	0.0000201
Rata-rata				0.0000206



Gambar 4.13. Grafik Dekompresi *File* Berformat *.wav* yang Ukurannya 0 – 1 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran *File*

Tabel 4.19. Waktu Dekompresi *File* Berformat *.wav* yang Ukurannya 1 – 5 MB

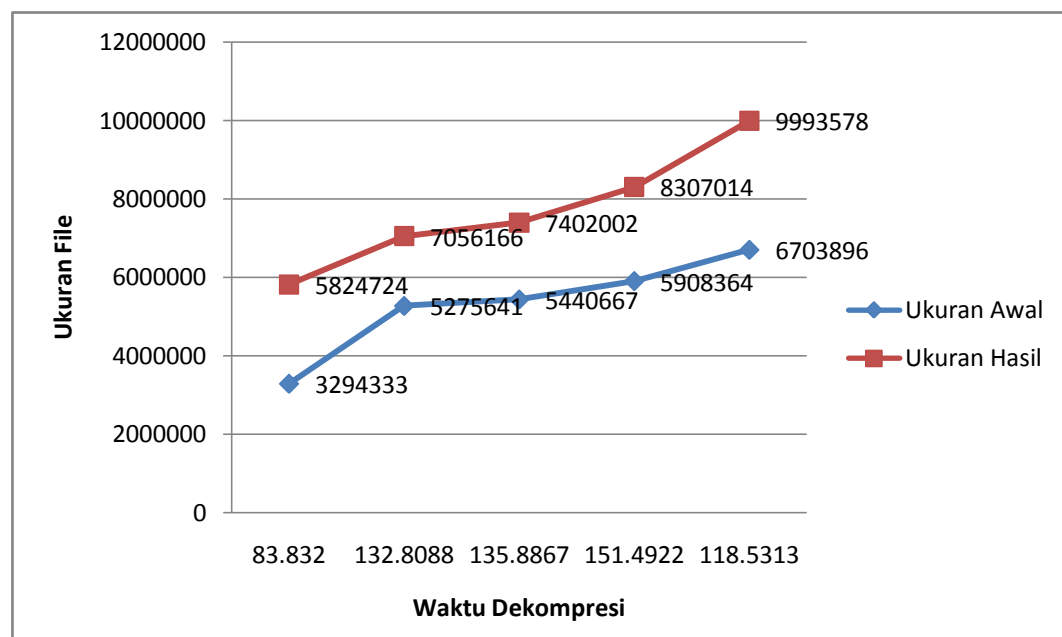
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi/ Byte
File Audio6.aci	604054	1267904	14.8320	0.0000246
File Audio7.aci	1022593	1582336	22.3945	0.0000219
File Audio8.aci	1429864	1817412	35.6094	0.0000249
File Audio9.aci	1195167	1932892	25.5391	0.0000214
File Audio10.aci	2723073	4782284	72.3828	0.0000266
Rata-rata				0.0000239



Gambar 4.14. Grafik Dekompresi *File* Berformat *wav* yang Ukuran Awalnya 1 – 5 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran *File*

Tabel 4.20. Waktu Dekompresi *File* Berformat *wav* yang Berukuran Awal 5–10 MB

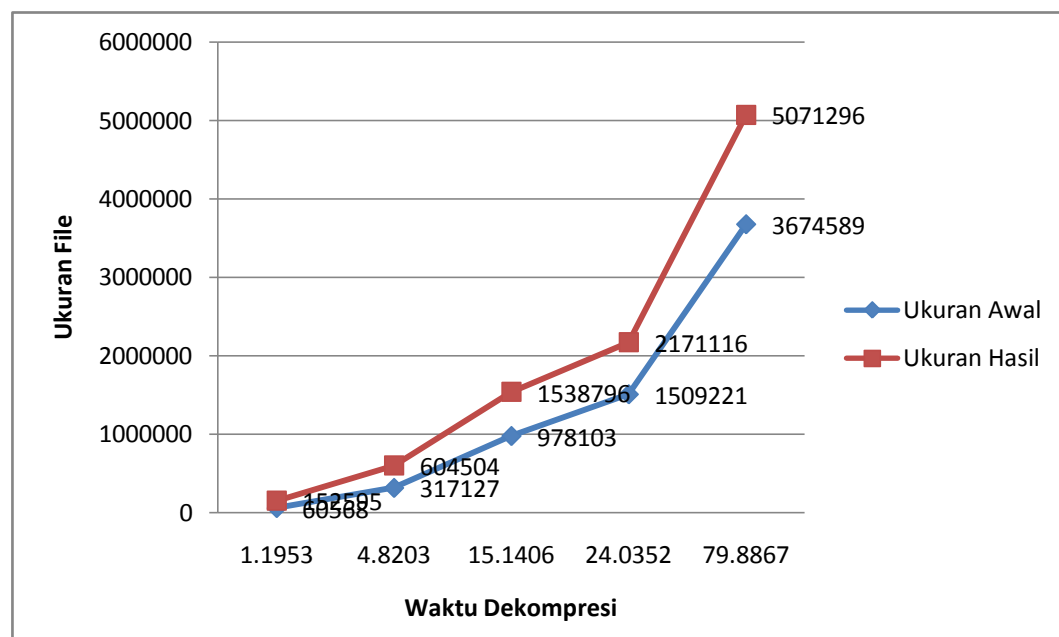
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi/ Byte
File Audio11.aci	3294333	5824724	83.8320	0.0000254
File Audio12.aci	5275641	7056166	132.8088	0.0000252
File Audio13.aci	5440667	7402002	135.8867	0.0000250
File Audio14.aci	5908364	8307014	151.4922	0.0000256
File Audio15.aci	6703896	9993578	118.5313	0.0000177
Rata-rata				0.0000238



Gambar 4.15. Grafik Dekompresi *File* Berformat *.wav* yang Ukurannya 5 – 10 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran *File*

Tabel 4.21. Waktu Dekompresi *File* Berformat *.wav* yang Memiliki Satu Jenis Suara

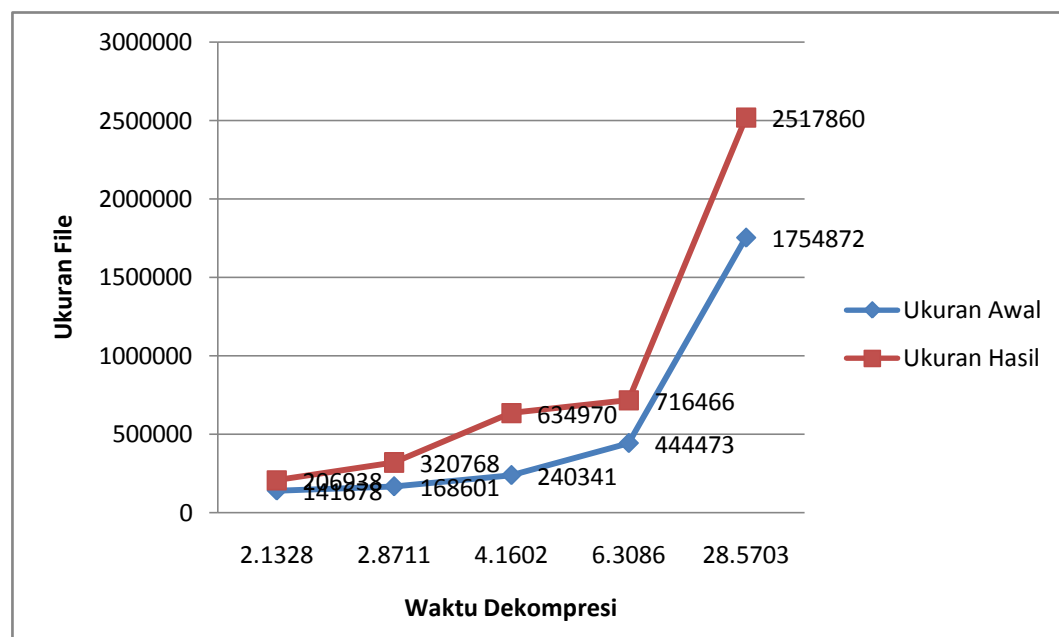
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi/ Byte
File Audio16.aci	60568	152595	1.1953	0.0000197
File Audio17.aci	317127	604504	4.8203	0.0000152
File Audio18.aci	978103	1538796	15.1406	0.0000155
File Audio19.aci	1509221	2171116	24.0352	0.0000159
File Audio20.aci	3674589	5071296	79.8867	0.0000217
Rata-rata				0.0000176



Gambar 4.16. Grafik Dekompresi *File* Berformat *.wav* yang Memiliki Satu Jenis Suara Berdasarkan Waktu Dekompresi Terhadap Ukuran *File*

Tabel 4.22. Waktu Dekompresi *File* Berformat *.wav* yang Memiliki Dua Jenis Suara

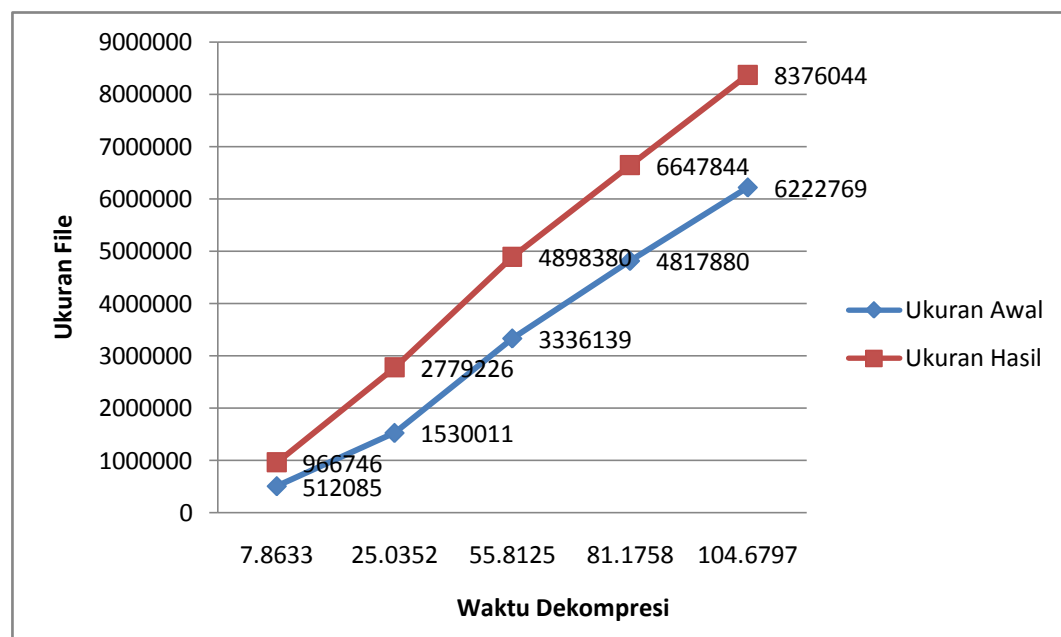
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi/ Byte
File Audio21.aci	141678	206938	2.1328	0.0000151
File Audio22.aci	168601	320768	2.8711	0.0000170
File Audio23.aci	240341	634970	4.1602	0.0000173
File Audio24.aci	444473	716466	6.3086	0.0000142
File Audio25.aci	1754872	2517860	28.5703	0.0000163
Rata-rata				0.0000160



Gambar 4.17. Grafik Dekompresi *File* Berformat *.wav* yang Memiliki Dua Jenis Suara Berdasarkan Waktu Dekompresi Terhadap Ukuran *File*

Tabel 4.23. Waktu Dekompresi *File* Berformat *.wav* yang Memiliki Beragam Jenis Suara

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi/ Byte
File Audio26.aci	512085	966746	7.8633	0.0000154
File Audio27.aci	1530011	2779226	25.0352	0.0000164
File Audio28.aci	3336139	4898380	55.8125	0.0000167
File Audio29.aci	4817880	6647844	81.1758	0.0000168
File Audio30.aci	6222769	8376044	104.6797	0.0000168
Rata-rata				0.0000164



Gambar 4.18. Grafik Dekompresi *File* Berformat *.wav* yang Memiliki Beragam Jenis Suara Berdasarkan Waktu Dekompresi Terhadap Ukuran *File*

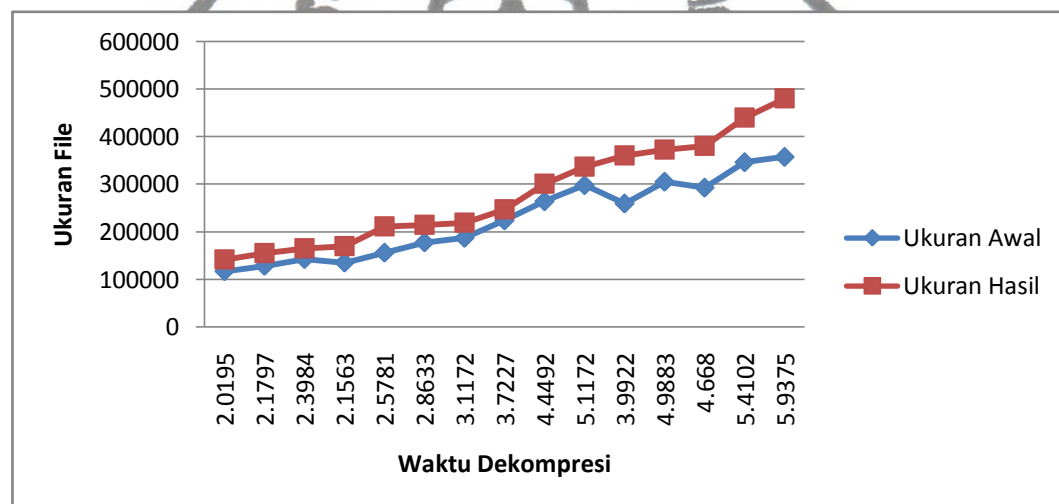
4.3.2.2 Pengujian Dekompresi *File Audio* yang Berformat *.aif*

Tabel 4.24. Waktu Dekompresi untuk *File* Berformat *.aif*

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi/ Byte
File Audio31.aci	117007	142002	2.0195	0.0000173
File Audio32.aci	128286	155354	2.1797	0.0000170
File Audio33.aci	142565	165842	2.3984	0.0000168
File Audio34.aci	135124	169902	2.1563	0.0000160
File Audio35.aci	156065	211210	2.5781	0.0000165
File Audio36.aci	177766	215122	2.8633	0.0000161
File Audio37.aci	187264	219634	3.1172	0.0000166
File Audio38.aci	224085	247578	3.7227	0.0000166
File Audio39.aci	264393	301226	4.4492	0.0000168

Tabel 4.24. Lanjutan

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi/ Byte
File Audio40.aci	298294	337170	5.1172	0.0000172
File Audio41.aci	259621	360454	3.9922	0.0000154
File Audio42.aci	305493	372922	4.9883	0.0000163
File Audio43.aci	293089	380574	4.6680	0.0000159
File Audio44.aci	346660	439458	5.4102	0.0000156
File Audio45.aci	357819	480162	5.9375	0.0000166
Rata-rata				0.0000164

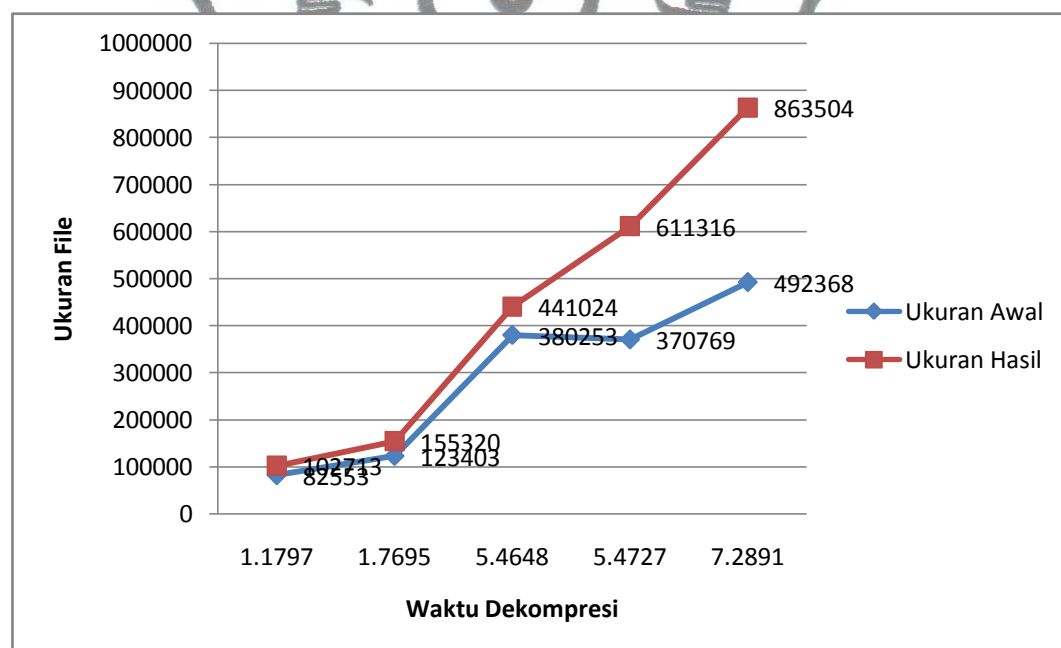


Gambar 4.19. Grafik Dekompresi *File* Berformat *.aif* Berdasarkan Waktu Dekompresi Terhadap Ukuran *File*

4.3.2.3 Pengujian Dekompresi *File Audio* yang Berformat *.au*

Tabel 4.25. Waktu Dekompresi *File* Berformat *.au* yang Ukuran Awalnya 0 – 1 MB

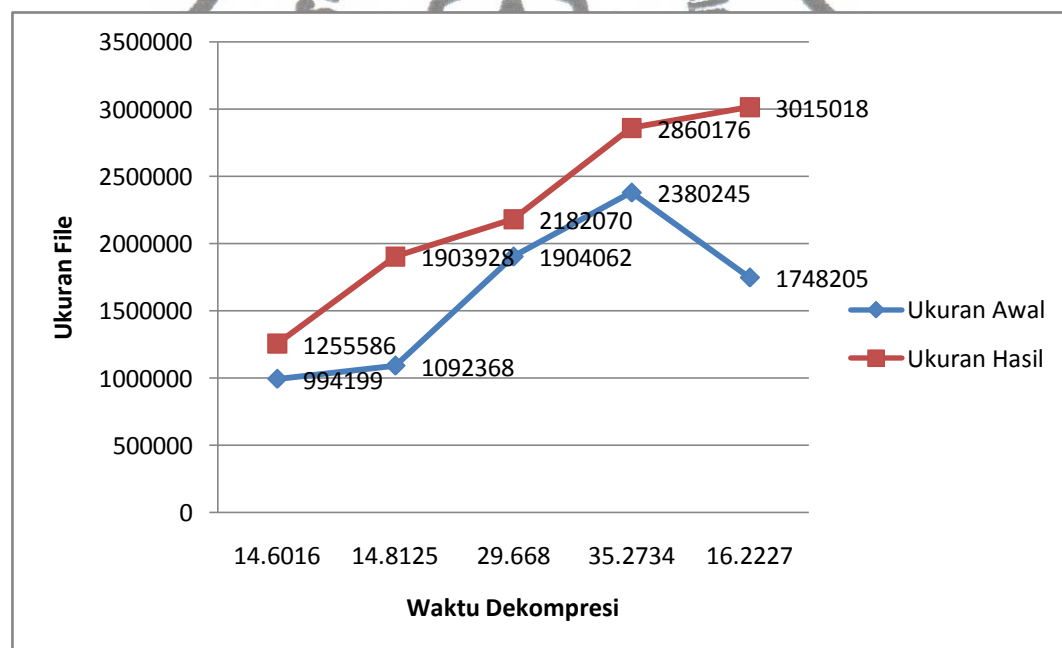
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi/ Byte
File Audio46.aci	82553	102713	1.1797	0.0000143
File Audio47.aci	123403	155320	1.7695	0.0000143
File Audio48.aci	380253	441024	5.4648	0.0000144
File Audio49.aci	370769	611316	5.4727	0.0000148
File Audio50.aci	492368	863504	7.2891	0.0000148
Rata-rata				0.0000145



Gambar 4.20. Grafik Dekompresi *File* Berformat *.au* yang Ukuran Awalnya 0 – 1 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran *File*

Tabel 4.26. Waktu Dekompresi *File* Berformat *.au* yang Ukuran Awalnya 1 – 5 MB

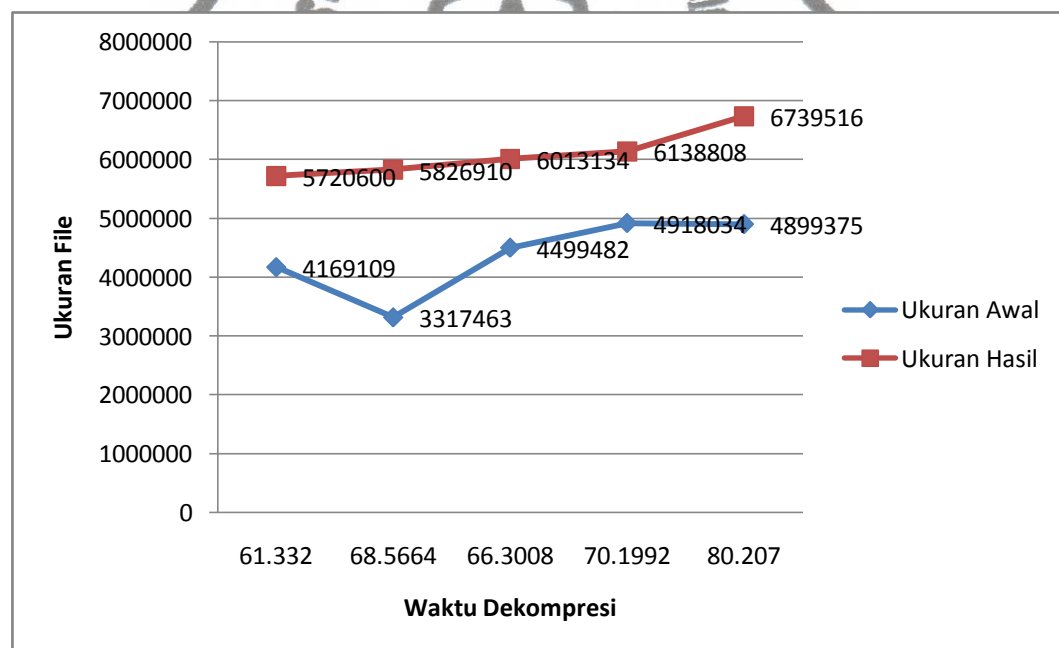
Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi/ Byte
File Audio51.aci	994199	1255586	14.6016	0.0000147
File Audio52.aci	1092368	1903928	14.8125	0.0000136
File Audio53.aci	1904062	2182070	29.6680	0.0000156
File Audio54.aci	2380245	2860176	35.2734	0.0000148
File Audio55.aci	1748205	3015018	16.2227	0.0000093
Rata-rata				0.0000136



Gambar 4.21. Grafik Dekompresi *File* Berformat *.au* yang Ukuran Awalnya 1 – 5 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran *File*

Tabel 4.27. Waktu Dekompresi *File* Berformat *.au* yang Ukuran Awalnya 5 – 10 MB

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi/ Byte
File Audio56.aci	4169109	5720600	61.3320	0.0000147
File Audio57.aci	3317463	5826910	68.5664	0.0000207
File Audio58.aci	4499482	6013134	66.3008	0.0000147
File Audio59.aci	4918034	6138808	70.1992	0.0000143
File Audio60.aci	4899375	6739516	80.2070	0.0000164
Rata-rata				0.0000162

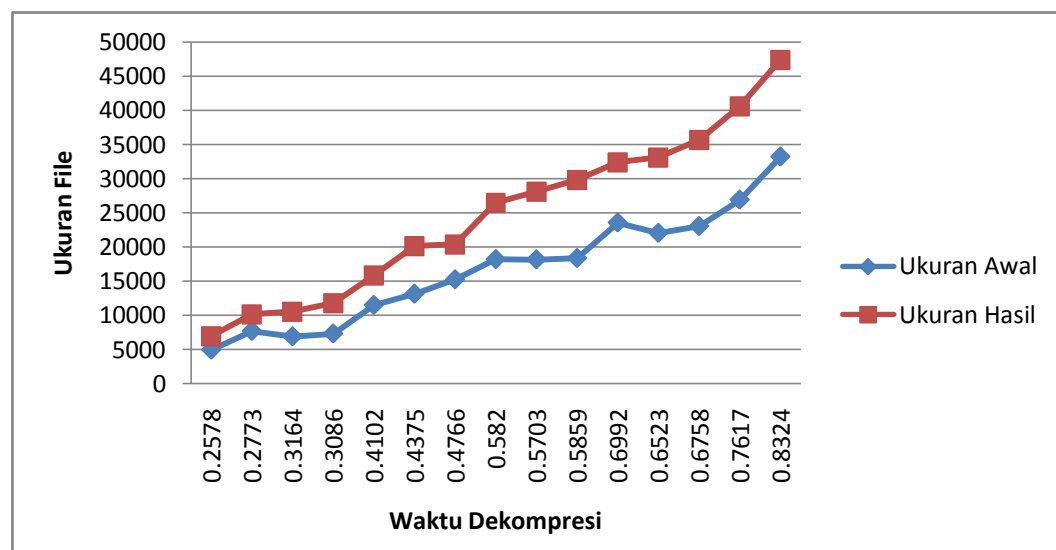


Gambar 4.22. Grafik Dekompresi *File* Berformat *.au* yang Ukuran Awalnya 5 – 10 MB Berdasarkan Waktu Dekompresi Terhadap Ukuran *File*

4.3.2.4 Pengujian Dekompresi *File Audio* yang Berformat *.mid*

Tabel 4.28. Waktu Dekompresi untuk *File* Berformat *.mid*

Nama File	Ukuran Awal (Byte)	Ukuran Hasil (Byte)	Waktu Dekompresi (s)	Waktu Dekompresi /Byte
File Audio61.aci	4957	6960	0.2578	0.0000520
File Audio62.aci	7702	10133	0.2773	0.0000360
File Audio63.aci	6917	10556	0.3164	0.0000457
File Audio64.aci	7326	11789	0.3086	0.0000421
File Audio65.aci	11504	15886	0.4102	0.0000357
File Audio66.aci	13134	20151	0.4375	0.0000333
File Audio67.aci	15250	20387	0.4766	0.0000313
File Audio68.aci	18207	26500	0.5820	0.0000320
File Audio69.aci	18182	28115	0.5703	0.0000314
File Audio70.aci	18365	29793	0.5859	0.0000319
File Audio71.aci	23563	32426	0.6992	0.0000297
File Audio72.aci	22050	33123	0.6523	0.0000296
File Audio73.aci	23043	35652	0.6758	0.0000293
File Audio74.aci	26911	40629	0.7617	0.0000283
File Audio75.aci	33216	47391	0.8324	0.0000251
Rata-rata			0.52293	0.0000342



Gambar 4.23. Grafik Dekompresi *File* Berformat *.mid* Berdasarkan Waktu Dekompresi Terhadap Ukuran *File*

Berdasarkan Tabel 4.18 sampai dengan Tabel 4.28, bisa dihitung rata-rata waktu dekompresi/byte untuk keseluruhan jenis *file* yang diujikan.

Tabel 4.29. Rata-rata Waktu Dekompresi/Byte untuk *File* yang Berukuran 0 – 1 MB

No	Waktu Dekompresi/Byte
Rata-rata file .wav	0.0000206
Rata-rata file .aif	0.0000164
Rata-rata file .au	0.0000145
Rata-rata file .mid	0.0000342
Total	0.0000857
Rata-rata	0.0000171

Tabel 4.30. Rata-rata Waktu Dekompresi/Byte untuk File yang Berukuran 1 – 5 MB

No	Waktu Dekompresi/Byte
Rata-rata file .wav	0.0000239
Rata-rata file .au	0.0000136
Total	0.0000375
Rata-rata	0.0000187

Tabel 4.31. Rata-rata Waktu Dekompresi/Byte untuk File yang Berukuran 5 – 10 MB

No	Waktu Dekompresi/Byte
Rata-rata file .wav	0.0000238
Rata-rata file .au	0.0000162
Total	0.0000400
Rata-rata	0.0000200

Dilihat dari Tabel 4.29, Tabel 4.30, dan Tabel 4.31, rata-rata waktu dekomposisi/byte untuk semua ukuran file hampir mempunyai rata-rata yang sama. Meskipun waktu dekomposisi dari setiap jenis *file* naik secara linier sesuai dengan ukuran *file*, tetapi waktu dekomposisi tiap byte-nya hampir sama untuk semua ukuran *file*. Semisal rata-rata waktu dekomposisi/byte yang berukuran 0 – 1 MB diambil kemudian dikalikan dengan sebuah file yang berukuran 400 KB, maka, Waktu Kompresi = $0.0000171 \times 400000 = 6.84 \text{ s}$

Waktu kompresi dikalikan dengan sebuah file yang berukuran 400 KB karena waktu kompresi pada hipotesa awal didasarkan atas waktu dekomposisi dengan ukuran file maksimum 400 KB. Pada hipotesa awal rata-rata waktu dekomposisi sebesar 4 s, hal ini menunjukkan bahwa hasil akhir waktu dekomposisi belum sesuai dengan yang diharapkan. Dari keseluruhan Tabel 4.18 sampai dengan Tabel 4.29, dapat disimpulkan sebagai berikut :

1. Ukuran hasil dekomposisi besarnya sama dengan ukuran awal *file* sebelum dilakukan kompresi.

2. Waktu dekompresi akan lebih lama dibanding dengan waktu kompresi, meskipun kadang juga tidak pasti lebih lama.
3. Waktu dekompresi yang dihasilkan pada setiap jenis *file* tergantung oleh besarnya ukuran *file*, kondisi hardware, dan susunan datanya.

4.3.3 Hasil Pengujian Kualitas Hasil

Pengujian kualitas hasil dilakukan dengan cara membandingkan antara *file* sebelum dilakukan proses kompresi dengan *file* yang telah didekompresi. Dilakukan dengan 4 kali pengujian yaitu untuk semua jenis *file*.

Tabel 4.32. Kesamaan Kualitas File Sebelum Dikompresi dan Sesudah Didekompresi

Jenis File	Kesamaan Data (%)
File .wav	100
File .aif	100
File .au	100
File .mid	100

Berdasarkan Tabel 4.32 diketahui bahwa kesamaan kualitas antara *file* sebelum dilakukan proses kompresi dengan *file* yang telah didekompresi adalah sama semua untuk semua jenis *file*. Hal ini membuktikan bahwa algoritma *Arithmetic Coding* dengan bilangan integer merupakan salah satu algoritma *lossless compression*.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil implementasi dan pengujian serta analisa yang telah dilakukan sebelumnya, maka dapat ditarik kesimpulan sebagai berikut :

1. Telah dihasilkan sebuah aplikasi kompresi *file audio* menggunakan algoritma *Arithmetic Coding* dengan bilangan *integer*.
2. Hasil dari pengujian yang dilakukan menunjukkan bahwa :
 - Besarnya rata-rata rasio kompresi, waktu kompresi, dan waktu dekompresi berturut-turut adalah 28.648 %, 5.8 s, dan 6.84 s.
 - Besarnya waktu kompresi dan waktu dekompresi naik secara linier sesuai dengan ukuran *file*, tetapi waktu kompresi/waktu dekompresi tiap byte-nya hampir sama untuk semua ukuran *file*.
 - Rata-rata rasio kompresi paling besar serta waktu kompresi dan dekompresi paling cepat berturut-turut adalah pada *file* yang berformat, *.wav*, *.mid* *.au*, dan *.aif*.
 - Kualitas struktur data antara file sebelum dilakukan proses kompresi dan file setelah didekompresi adalah sama persis.

5.2 Saran

Adapun saran yang penulis dapat berikan adalah :

1. Dapat dibangun aplikasi kompresi dengan *Arithmetic Coding* yang mampu mengkompresi semua jenis *file* seperti jenis *file* teks, *audio*, *video*, dan lain-lain.
2. Aplikasi masih lemah dalam hal waktu kompresi dan waktu dekompresi, sehingga perlu dilakukan optimalisasi yang bisa mempercepat waktu kompresi dan dekompresi.

commit to user