IMPLEMENTASI ALGORITMA KOMPRESI LZ77 PADA *SMARTPHONE BLACKBERRY*

Tommy Gunardi

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
tommy_gunardi@hotmail.com

Rinaldi Munir

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan informatika
Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
rinaldi@informatika.org

ABSTRAK

Dewasa ini, seluruh aspek teknologi berkembang dengan sangat pesat. Hal ini juga dipicu oleh banyaknya *smartphone* seperti *BlackBerry* yang terus dikeluarkan. Kegunaan dari *smartphone* ini tidak hanya untuk menelpon atau sms saja, namun digunakan sebagai *email*, media hiburan (gambar, video, lagu), *games*, *chatting*, *social network* dan lain-lain. Tentu dengan kegunaannya yang semakin beragam dan lengkap, *smartphone* seperti *BlackBerry* membutuhkan *storage* atau tempat penyimpanan data yang cukup besar. Namun, aplikasi khusus untuk mengompresi/mendekompresi *file* pada *storage BlackBerry* belum ada.

Oleh karena itu, pada Penelitian ini dibuatlah suatu aplikasi untuk mengompresi dan mendekompresi data pada *storage BlackBerry*. Aplikasi ini bernama *BB-Compress*. Aplikasi ini dibuat sebagai suatu aplikasi khusus yang bertujuan mengompresi dan mendekompresi berbagai macam *file* yang ada pada *storage BlackBerry* dengan menggunakan algoritma kamus *LZ77*.

BB-Compress diimplementasikan pada *BlackBerry Bold* 9000. *BlackBerry* memiliki kemampuan perangkat keras dan lunak yang cukup memadai untuk melakukan proses kompresi dan dekompresi. *BB-Compress* ini diimplementasikan dengan menggunakan kakas *Eclipse Ganymede* dan *Netbeans*.

BB-Compress berhasil mengimplementasikan algoritma *LZ77* pada *smartphone BlackBerry* dan dapat melakukan fungsinya untuk mengompresi dan mendekompresi berbagai macam *file* pada *BlackBerry*. Untuk beberapa *file*, aplikasi ini menunjukkan hasil yang sangat baik dalam kompresinya. Namun waktu kompresi pada *BB-Compress* menjadi masalah karena proses kompresi berjalan cukup lambat.

KATA KUNCI

BB-Compress, LZ77, BlackBerry, kompresi dan dekompresi

I. Pendahuluan

Dewasa ini, seluruh aspek teknologi berkembang dengan sangat pesat. Hal ini juga dipicu oleh banyaknya alat komunikasi *mobile* seperti *BlackBerry* dan *iPhone* tipe baru yang terus dikeluarkan. Alat komunikasi tersebut saat ini sudah menjadi *smartphone*, yaitu sebuah *mobile phone* yang menawarkan kemampuan komputasi dan konektivitas lebih dari sekedar basic *phone* kontemporer. *Smartphone* mungkin dianggap sebagai komputer genggam yang terintegrasi dalam telepon seluler, sementara kebanyakan *mobile phone* menjalankan aplikasi berbasis *platform Java ME*, *smartphone* memungkinkan pengguna untuk meng*instal* dan menjalankan aplikasi lebih maju berdasarkan *platform* yang lebih spesifik. *Smartphone* menjalankan suatu sistem operasi lengkap dan menyediakan *platform* untuk pengembang aplikasi. Jadi saat ini kita dapat mempertimbangkan sebuah *smartphone* sebagai *Pocket Personal Computer* (PPC) dengan fungsi ponsel, karena sesungguhnya *device-device* ini adalah komputer, meskipun dengan ukuran lebih kecil daripada *desktop* komputer.

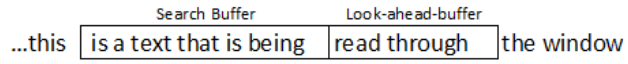
Algoritma kompresi *LZ77* adalah suatu metoda kompresi data *lossless* yang akan dibuat aplikasinya pada pengerjaan penelitian ini. Aplikasi ini bertujuan untuk melakukan pengompresian dan pendekompresian data secara khusus pada *BlackBerry*. Algoritma *LZ77* diperkenalkan untuk pertama kalinya pada tahun 1977 oleh Abraham Lempel dan Jacob Ziv. Algoritma *LZ77* merupakan suatu algoritma dasar yang banyak dikembangkan oleh orang-orang. Misalnya saja *LZW*, *LZSS*, dan *LZMA*. Algoritma *LZ77* sendiri sering disebut dengan *LZ1* (Salomon, 2006). Algoritma ini juga disebut algoritma "*sliding window*" karena melakukan kompresi data dengan cara menggerakkan *buffer* tempat simbol-simbol berada setiap kali satu atau lebih simbol terkompresi. *Buffer* yang dibutuhkan algoritma kompresi ini ada dua, yaitu *search buffer* dan *look-ahead buffer*. Kedua *buffer* ini memegang peran penting dalam melakukan kompresi terhadap suatu *file*. Alasan pemilihan algoritma ini adalah bahwa algoritma ini memberikan performa terbaik untuk beberapa jenis file dibandingkan algoritma-algoritma modifikasinya (Zeeh, 2003).

Menurut beberapa aplikasi terkait kompresi data pada *BlackBerry* (AppWorld, 2011), kebanyakan aplikasi tersebut menambahkan kompresi dan dekompresi sebagai fungsi tambahan dari fungsi utamanya. Aplikasi lainnya hanya dapat melakukan dekompresi terhadap suatu file yang terkompresi. Pihak *RIM* (Re*search in Motion*) yakni perusahaan yang mendevelop *BlackBerry* di Kanada, lebih mengkhususkan kompresi data terhadap data-data yang ada di internet. Hal ini dikarenakan *wireless* data merupakan prioritas RIM. Jumlah pemakaian *wireless* data antara semua alat komunikasi *mobile* naik sebesar 463 persen antara akhir tahun 2009 sampai pertengahan tahun 2010 (Finocchiaro 2010). Pengguna *BlackBerry* mengonsumsi sekitar 79.5 *megabyte* data per bulan untuk setiap *user*, dibandingkan dengan 375 *megabyte* untuk *iPhone user*. Namun karena terlalu mengkhususkan kepada *wireless* data yang diunduh dan diunggah, data yang ada di *storage* tidak dikompresi.

II. Algoritma yang digunakan

Metoda ini dikenalkan sebagai suatu algoritma kompresi data lossless yang dikeluarkan sebagai paper oleh Abraham Lempel dan Jacob Ziv pada tahun 1977. Algoritma ini juga dikenal sebagai algoritma LZ1. Algoritma ini menjadi dasar dari beberapa algoritma lainnya seperti LZW, LZSS, LZMA dan lainnya.

Algoritma kompresi LZ77 melakukan kompresinya dengan mengganti porsi data dengan suatu referensi untuk menyamakan data yang sudah dilewati oleh encoder dan decoder. Algoritma ini menggunakan suatu "sliding window" (Muller 2008) yang terdiri dari search buffer dan look-ahead buffer. Search buffer digunakan sebagai kamus, sedangkan look-ahead buffer merupakan buffer yang berisi string yang akan dikompresi. Search buffer merupakan buffer yang sudah melalui tahap kompresi, namun digunakan sebagai kamus. Kedua buffer tersebut dapat dilihat pada contoh berikut :



Gambar 1 Search buffer dan Look-ahead buffer (Salomon 2006)

Metode ini melakukan pencarian satu (atau lebih) simbol pada look-ahead buffer yang sama dengan simbol pada search buffer. Pencarian dilakukan dari kanan ke kiri. Prakteknya, perbandingan ukuran search buffer dengan look-ahead buffer kira-kira adalah beberapa ribu banding 10.

Algoritma ini melakukan pencarian terhadap kumpulan simbol terpanjang yang sama pada awal look-ahead buffer terhadap search buffer dan mengeluarkan sebuah pointer terhadap kesamaan tersebut. Secara umum, sebuah token LZ77 memiliki tiga buah bagian yakni : offset, panjang, dan simbol selanjutnya pada look-ahead buffer. Offset dan panjang merupakan pointer yang menunjukkan posisi dan jumlah kecocokan. Apabila ditemukan kecocokan, LZ77 menambahkan suatu pointer dengan suatu simbol di belakangnya. Apabila tidak ada yang cocok, maka dihasilkan null pointer dan simbol unik. Selanjutnya sliding window digeser sebanyak simbol yang terkompresi berdasarkan pointernya.

III. Analisis

III.1 Analisis Existing System

Pada bab ini akan dijelaskan mengenai analisis existing system, yakni sistem dimana aplikasi akan dikembangkan, yaitu deskripsi *BlackBerry Bold* 9000. Hal-hal yang akan dijelaskan mengenai *encoding* pada *BlackBerry* dan kesesuaian algoritma pada *BlackBerry*.

IANA (*Internet Assigned Numbers Authority*) (IANA, 2011) adalah suatu organisasi yang bertanggungjawab terhadap alokasi dari nama dan angka yang digunakan dalam internet protocols yang dipublikasikan oleh dokumen RFC (*Request for Comments*). IANA mengelola banyak parameter dalam protokol internet. Beberapa contohnya adalah nama skema *encoding* karakter yang direkomendasikan untuk digunakan pada internet.

Encoding adalah pemetaan karakter-karakter dalam format tertentu yang bertujuan untuk mengefisienkan transmisi atau *storage*. Beberapa karakter *encoding* yang disuport oleh *BlackBerry* antara lain :

1. "ISO-8859-1"
   ISO-8859-1 merupakan *encoding* default *BlackBerry*. ISO 8859-1 mengencode "Latin alphabet no.1" yang berisi 191 karakter dari skrip latin, masing-masing dikodekan dengan sebuah nilai kode berukuran 8 bit (Wordiq, 2011). *Encoding* ini digunakan di hampir semua bahasa Eropa Barat.
2. *UTF-8*
   UTF-8 (*UCS Transformation Format-8bit*), dikatakan *backward compatible* dengan ASCII karena dapat membaca standar input *ASCII*. *Encoding* ini digunakan karena *BlackBerry* merupakan *device smartphone* yang mendukung fasilitas email dan internet. UTF-8 merupakan *encoding* dominan untuk *world-wide web* yang digunakan lebih dari setengah total web yang ada.
3. "UTF-16BE"
   UTF-16BE (Unicode Transformation Format – 16bit Big Endian) merupakan *encoding* karakter yang memetakan kode poin dari karakter set unicode ke rangkaian 2 *byte*s. *Encoding* ini digunakan pada internet protocol dan beberapa bahasa pemrograman bahkan aplikasi-aplikasi software, namun bukan merupakan *encoding* standar pada internet protocol.
4. "US-ASCII"
   American Standard Code for Information Interchange meruoajan skema *encoding* karakter berbasis urutan dari alfabet bahasa inggris. Kode ASCII merepresentasikan teks dalam komputerm peralatan komunikasi dan *device* lain yang menggunakan teks. Kebanyakan skema *encoding* karakter modern berbasis ASCII, meskipun kadang mereka mensupport lebih banyak karakter dibandingkan ASCII.

Banyaknya *encoding* yang dapat disupport *BlackBerry* tidak membuat proses kompresi menjadi merepotkan karena format string pada Java adalah Unicode. Oleh karena itu tidak perlu dilakukan konversi terhadap data dengan menggunakan *encoding* seperti ISO-8859-1 atau UTF-8 (Strange, 2010).

III.2 Kesesuaian Algoritma LZ77 pada device BlackBerry

Pengimplementasian Algoritma *LZ77* pada *BlackBerry* dengan memperhitungkan *encoding* default *BlackBerry* (ISO-8859-1), yakni per 8 bit atau 1 *byte*, sehingga pengompresian dilakukan per *byte*. Pengompresian dilakukan dalam beberapa tahap, tergantung besarnya *file*. Algoritma dilakukan per tahap supaya tidak boros memori.

III.2.1 Kompresi

Berikut Pseudocode standar kompresi *LZ77* :

```
while( lookAheadBuffer not empty )
          {
          get pointer ( position, match ) to the
longest match in the window
               for the lookahead buffer;

          if( length > MINIMUM_MATCH_LENGTH )
               {
               output a ( position, length ) pair;
               shift the window length characters along;
               }
          else
               {
```

```
            output the first character in the
lookahead buffer;
            shift the window 1 character along;
            }
        }
```

Pertama-tama program membaca *file* sebesar y, lalu dimasukkan ke dalam suatu array of *byte* dengan ukuran x. Apabila ukuran y lebih besar dari x maka isi *file* yang diambil hanya sebesar x terlebih dahulu. Sebelumnya *file* output diberi header yang menunjukkan bahwa *file* ini benar telah terkompresi oleh algoritma *LZ77*. Pengompresian dilakukan terhadap buffer tersebut. Bagian yang disimpan adalah isi dari *byte* tersebut dan posisi *byte*. Setelah seluruh isi buffer tersebut terkompresi, isi buffer dimasukkan ke *file* output. Buffer lalu diisi kembali sisa dari *file* dengan ukuran y-x (karena sudah diambil sebanyak x pada kompresi sebelumnya) sebanyak x lalu dilakukan kompresi terhadap buffer tersebut. Hal ini dilakukan terus-menerus sampai *file* dengan ukuran y sudah selesai terkompresi seluruhnya.

III.2.2 Dekompresi

Pertama-tama program membaca header *file* apakah benar *file* ini terkompresi oleh program kompresi *LZ77*. Lalu program membaca isi *file* terkompresi dengan format yang diletakkannya adalah isi, posisi, dan panjang *byte*(-*byte*) tersebut. Setelah terbaca, program mengekstraksi *file* seperti keadaan semula.

IV. Implementasi

IV.1 Lingkungan Implementasi

Implementasi Aplikasi *BB-Compress* mencakup dua macam lingkungan implementasi yakni lingkungan implementasi peranti keras dan peranti lunak. Lingkungan implementasi peranti keras adalah sebagai berikut :

***BlackBerry Bold* 9000**

-	Display Type	65k colors
-	Display Monitor Size	480 x 320 pixels, 2.6 inches
-	Memory Internal	1 GB *storage*, 128 MB RAM
-	Memory Card	microSD 1 GB
-	CPU	Intel Xscale 624MHz
-	OS	v5.0.0.822 (Bundle 1385, Platform 5.2.0.76)

Lingkungan implementasi peranti lunaknya menggunakan bahasa pemrograman Java for *BlackBerry*. Beberapa kakas pemrograman dan peranti lunak lain yang digunakan adalah sebagai berikut :

- *Netbeans 6.8*
- *Eclipse Ganymede*
- *BlackBerry Simulator*
- *Adobe Photoshop*

IV.2 Batasan Implementasi

Berikut merupakan batasan implementasi yang ada pada Penelitian ini :

1. Tidak menangani kompresi folder
2. Kompresi/dekompresi dilakukan hanya jika tempat untuk menampung hasil kompresi/dekompresi tersedia (lebih atau sama dengan hasil).
3. Diimplementasikan pada *BlackBerry Bold* 9000
4. Hasil Kompresi/dekompresi diletakkan pada direktori *file* yang sama dengan sumber.

IV.3 Implementasi Kelas

Implementasi kelas tidak berbeda jauh dari hasil perancangan kelas. Berikut Diagram kelas yang ada dapat dilihat pada tabel V-1:

Tabel 1 Implementasi Kelas

Kelas Rancangan	Implementasi
BitIO	`BitIO.java`
LZ77Compress	`LZ77Compress.java`
LZ77Decompress	`LZ77Decompress.java`

IV.4 Implementasi Antarmuka

Implementasi antarmuka agak berbeda dari perancangan antarmuka karena terjadi beberapa penambahan fitur seperti kompresi terhadap banyak *file* sekaligus dan penambahan progress bar. Antarmuka program adalah sebagai berikut :

Gambar 1 Antarmuka utama aplikasi BB-Compress

V. Pengujian

V.1 Tujuan Pengujian

Tujuan pengujian penelitian ini adalah memeriksa apakah kebutuhan fungsional sudah terpenuhi yakni dapat mengompresi dan mendekompresi *file* yang ada pada *smartphone BlackBerry*. Tujuan lain pengujian ini adalah untuk mengukur performa aplikasi yang mencakup rasio kompresi, entropi, dan waktu kompresi.

V.2 Rancangan Kasus Uji

Beberapa rancangan kasus uji yang dibuat dalam Penelitian ini mencakup dua macam kasus uji. Pengujian pada Penelitian ini mencakup beberapa kasus uji sebagai berikut :

1. Pengujian terhadap beberapa ukuran *Search Buffer*
   Pengujian dilakukan terhadap ukuran search buffer yang berbeda-beda. Pada pengujian kali ini, ukuran search buffer yang diuji adalah 4095 dan 1023.
2. Pengujian terhadap berbagai macam tipe *file*
   Pengujian dilakukan terhadap berbagai jenis *file* seperti *file* gambar (jpg, gif, png), audio(mp3 dan wav) dan teks(txt).
3. Pengujian terhadap beberapa *file* sekaligus
   Pengujian dilakukan terhadap banyak *file* sekaligus dalam *storage BlackBerry*. Pada kasus ini dua buah sekaligus.

V.3 Batasan Pengujian

Beberapa batasan yang ada pada pengujian Penelitian ini adalah

1. Pengujian dilakukan hanya di *BlackBerry Bold* 9000
2. Perhitungan performa kompresi dihitung otomatis oleh program

Perhitungan waktu kompresi diasumsikan *BlackBerry* hanya menjalankan aplikasi ini dan tidak menyalakan aplikasi lain yang diperkirakan akan menggunakan prosesor secara besar.

V.4 Hasil Pengujian

Hasil pengujian akan ditampilkan pada lampiran.

V.5 Analisis Pengujian

Pada bagian ini akan dijelaskan mengenai analisis pengujian berdasarkan pelaksanaan pengujian untuk beberapa kasus *file* pada sub bab sebelumnya. Kompresi yang terjadi dapat dibedakan menjadi dua jenis file yaitu file teks dan file binary. Kompresi terhadap *file* teks dapat mengompresi hampir 50% terhadap ukuran semula. Waktu kompresi file teks cukup lambat. Pada kasus nomor 1, file teks berukuran 111 KB membutuhkan waktu 210 detik untuk mengompresinya menjadi 58.2 KB. Kasus nomor 2 membutuhkan waktu 72.42 detik untuk mengompresinya menjadi 20 KB. Dibutuhkan waktu 795.29 detik untuk file teks berukuran 368 KB pada kasus nomor 3 dan 97.054 detik untuk mengompresi 53 KB file nomor 4. Sehingga dapat diketahui kecepatan kompresi terhadap file teks berkisar antara 0.5- 1 byte per detiknya. Rasio kompresi file teks berkisar antara 50-60%. Entropi dari file teks berkisar sekitar 7.93 – 7.94.

Lain halnya dengan file binary. Beberapa *file* binary yang dikompresi pada pengujian sudah dalam format terkompresi. Apabila kita melakukan kompresi *LZ77* terhadap *file* yang sudah terkompresi kemungkinan besar yang terjadi adalah ekspansi yakni ukuran *file* hasil kompresi lebih besar dari ukuran awal. Lima dari tujuh kasus menunjukkan bahwa setelah mengimplementasikan algoritma LZ77, terjadi ekspansi. Ada dua kasus dimana hasil kompresi terhadap file tersebut berhasil mengurangi ukuran file. Waktu kompresi sangat lambat. Untuk binary *file* proses kompresi berjalan lebih lambat daripada teks karena variasi byte yang jauh lebih besar. Entropi dari file binary berkisar antara 7.86 – 7.92. Rata-rata terjadi ekspansi terhadap jenis file ini. Hal ini dikarenakan untuk file *binary*, byte-byte yang merepresentasikannya ada 256 byte, sedangkan untuk file teks kebanyakan kurang dari itu. Karena kesamaan terhadap file teks relatif lebih banyak dibandingkan file *binary*, otomatis rasio kompresi untuk jenis file ini lebih terjamin.

Pengubahan terhadap ukuran search buffer mengakibatkan perubahan rasio kompresi dan kecepatan kompresi dan dekompresi. Semakin kecil ukuran search buffer, maka kecepatan kompresi semakin kecil, dengan konsekuensi rasio kompresi yang semakin besar. Karena file kompresinya besar, kecepatan dekompresinya pun lebih lambat. Semakin besar ukuran search buffer, maka kecepatan kompresi semakin lambat, dengan keuntungan rasio kompresi yang semakin kecil. Karena hasil kompresinya makin kecil, maka kecepatan dekompresinya makin cepat.

Dengan ini, kebutuhan fungsional untuk melakukan kompresi dan dekompresi dapat terpenuhi. Pengukuran performa kompresi juga berhasil diketahui dan dapat dibandingkan terhadap file yang berbeda-beda.

VI. Kesimpulan

Berdasarkan hasil pengujian terhadap kompresi data LZ77 pada smartphone BlackBerry, didapat beberapa kesimpulan sebagai berikut :

1. Aplikasi BB-Compress berhasil mengimplementasikan algoritma kompresi LZ77 pada BlackBerry Bold 9000
2. Algoritma LZ77 dapat diimplementasikan terhadap semua jenis file, namun tidak menjamin terjadinya kompresi. Bahkan dapat terjadi ekspansi.
3. Algoritma LZ77 menunjukkan performasi terbaiknya terhadap file teks karena jumlah simbol yang muncul lebih sedikit. Sedangkan byte-byte dalam file binary lebih bervariasi, maka kemungkinan kemunculan pola byte yang sama jarang terjadi.
4. Kurang efektifnya waktu kompresi dikarenakan prosesor BlackBerry yang kurang memadai. Semakin cepat prosesor, maka proses kompresi dan dekompresi akan semakin cepat.
5. Pemrosesan terhadap file teks lebih cepat daripada file binary karena variasi byte file teks lebih kecil daripada file binary.
6. Ukuran Search buffer mempengaruhi performa kompresi. Apabila diinginkan rasio kompresi makin baik bila ukuran search buffer diperbesar, sebaliknya untuk kecepatan kompresi yang terbaik didapat dengan memperkecil ukuran search buffer.

References

(Salomon, 2006) Salomon, David. *Data Compression The Complete Reference Fourth Edition*. Springer., 2006.

(Muller, 2008) Muller, Jens. *Data Compression LZ77*. Universitat Stuttgart., 2008.

(Strange, 2010) Strange, Peter. http://supportforums.*BlackBerry*.com/t5/Java-Development/Default-encoding/td-p/656701 dibuka pada tanggal 14 April 2011 pukul 07.18

(Finocchiaro,2010) Finocchiaro, Peter. *Mobile data consumption continues to accelerate: study*. http://www.mobilemarketer.com/cms/news/research/7319.html., 2010. Diakses pada tanggal 29 November 2010 pukul 22.05.

(Wordiq, 2011) Wordiq, http://www.wordiq.com/definition/ISO_8859-1 dibuka pada tanggal 13 April 2011 pukul 16.51.

(AppWorld, 2011) AppWorld. http://appworld.*BlackBerry*.com/webstore/category/76?lang=en dibuka pada tanggal 19 April 2011 pukul 00.09.

(Krzysztow, 2011) Krzysztow, http://krzysztow.com/BBNotePad/ dibuka pada tanggal 19 April 2011 pukul 11.06.

(IANA, 2011) IANA, http://www.iana.org/protocols/ dibuka pada tanggal 19 April 2011 pukul 12.01

(FileScout, 2011) *File*Scout, http://www.emacberry.com/bb*file*scout.html dibuka pada tanggal 19 April 2011 pukul 11.57.

(UnRAR, 2010) UnRAR, http://www.berryindo.com/*BlackBerry*-unrar/ dibuka pada tanggal 19 April 2011 pukul 11.58.

Lampiran

Data hasil pengujian dengan buffer 4095

No	File Awal				File Hasil Kompresi				Dekompresi
	Nama File	Tipe	Detail	Ukuran (KB)	Ukuran (KB)	Rasio (%)	Waktu(s)	Entropi	Waktu (s)
1.	bil	teks	*File* Bibliografi	111.3	58.2	53.64	332.69	7.938	206.62
2.	progc	teks	Source Code bahasa C	38.6	20.3	52.39	117.35	7.942	100.63
3.	news	teks	*USENET batch file*	368	217	59.0	1181.3	7.938	637.68
4.	paper1	teks	*Technical Paper*	53.1	28	54.3	158.25	7.940	91.23
5.	ERP.docx	*binary*	*Microsoft Word Document*	24	24.2	100.17	299.74	7.923	106.79
6.	imk.pdf	*binary*	*Portable Document Format*	96	101	105.1	1329.6	7.957	310.47
7.	bellring .wav	*audio*	*Wave Sound*	15	13.6	89.57	44.092	7.881	25.211
8.	teddy.gif	*image*	*GIF*	12	13.2	110.26	90.796	7.860	19.231
9.	roar.mp3	*audio*	*MP3*	7	7.06	107.43	50.807	7.900	15.923
10	IU.jpg	*image*	*JPEG*	16	140	72.85	102.71	7.911	23.653
	Scandal7 .jpg	*image*	*Joint Photographic Expert Group*	120		107.68	990.99	7.956	320.786

Data hasil pengujian dengan buffer 1023

No	File Awal				File Hasil Kompresi				Dekompresi
	Nama File	Tipe	Detail	Ukuran (KB)	Ukuran (KB)	Rasio (%)	Waktu(s)	Entropi	Waktu (s)
1.	bib	teks	*File* Bibliografi	111.3	68	66.31	102.76	7.907	236.23
2.	progc	teks	Source Code bahasa C	38.6	22	56.85	48.037	7.903	120.21
3.	news	teks	*USENET batch file*	368	250	67.78	584.1	7.887	628.21
4.	paper1	teks	*Technical Paper*	53.1	32	61.37	64.175	7.901	121.23
5.	ERP.docx	*binary*	*Microsoft Word Document*	24	24	100.89	92.122	7.916	126.79
6.	imk.pdf	*binary*	*Portable Document Format*	96	100	105.32	392.65	7.955	359.47
7.	bellring .wav	*audio*	*Wave Sound*	15	13.6	88.92	26.905	7.832	32.211
8.	teddy.gif	*image*	*GIF*	12	13.3	111.41	51.632	7.807	26.231
9.	roar.mp3	*audio*	*MP3*	7	8.21	107.27	20.3	7.894	31.923
10	IU.jpg	*image*	*JPEG*	16	142	72.35	50.96	7.883	28.653
	Scandal7 .jpg	*image*	*Joint Photographic Expert Group*	120		108.61	518.23	7.945	320.78