BAB III

METODOLOGI PENELITIAN

3.1 Alat dan Bahan Penelitian

3.1.1 Alat Penelitian

Dalam melakukan penelitian ini, berikut alat dan bahan penelitian yang digunakan:

1. Dari sisi perangkat keras, spesifikasi yang dipakai saat pembuatan aplikasi adalah:
   a. Processor Intel Pentium dual-core
   b. RAM 2 GB DDR2
   c. HDD 250 GB
   d. Mouse dan keyboard
2. Sedangkan dari sisi perangkat lunak, spesifikasi yang digunakan dalam pembuatan aplikasi kompresi data teks ini adalah sebagai berikut:
   a. Sistem operasi : Windows
   b. Tools : Matlab R2009a

3.1.2 Bahan Penelitian

Bahan penelitian merupakan entitas yang menjadi objek yang diolah atau diberi perlakuan-perlakuan tertentu, pengolahan atau perlakuan tersebut akan menghasilkan fenomena-fenomena yang dapat diamati, yang selanjutnya digunakan sebagai bahan kajian dalam penelitian (pedoman skripsi ilmu komputer

UPI, 2007). Bahan yang digunakan dalam penelitian ini adalah data teks yang berasal dari dokumen dengan format *.txt yang akan diolah ke dalam sistem.

3.2 Desain Penelitian

Desain penelitian menurut Mc Millan dalam Ibnu Hadjar (1999:102) adalah rencana dan struktur penyelidikan yang digunakan untuk memperoleh bukti-bukti empiris dalam menjawab pertanyaan penelitian. Sedangkan Lincoln dan Guba (1985:226) mendefinisikan rancangan penelitian sebagai usaha merencanakan kemungkinan-kemungkinan tertentu secara luas tanpa menunjukkan secara pasti apa yang akan dikerjakan dalam hubungan dengan unsur masing-masing.

Definisi lain mengatakan bahwa desain penelitian adalah rencana atau rancangan yang dibuat oleh peneliti, sebagai *ancar-ancar* kegiatan yang akan dilaksanakan (Suharsimi, 2006, h. 51).

Desain penelitian dibagi menjadi dua bagian, yaitu proses pengumpulan data dan metode pengembangan perangkat lunak. Proses pengumpulan data yang digunakan dalam penelitian ini adalah sebagai berikut:

1. Rumusan Masalah

   Sebuah penelitian dilakukan karena adanya suatu masalah yang akan dicari penyelesaian dan kebenarannya. Masalah itu diidentifikasi dan dibatasi, kemudian dirumuskan. Perumusan masalah dalam penelitian ini dapat dilihat pada bagian pendahuluan dalam perumusan masalah.

2. Studi Literatur

Studi literatur adalah penelusuran literatur yang bersumber dari buku, media, pakar ataupun dari hasil penelitian orang lain yang bertujuan untuk menyusun dasar teori yang kita gunakan dalam melakukan penelitian (Sayudjauhari, 2010). Studi literatur ini merupakan langkah berikutnya setelah melakukan perumusan masalah. Proses studi literatur dalam penelitian ini dilakukan dengan mempelajari literatur-literatur yang meliputi kompresi data teks, pohon biner Huffman dan Shannon-Fano. Literatur, yaitu buku, jurnal, *paper*, dan artikel ilmiah yang berhubungan dengan kompresi (khususnya algoritma Huffman dan algoritma Shannon-Fano).

3. Pengumpulan Data

Pada tahapan ini dilakukan pengumpulan data berupa data teks dengan kalimat panjang atau beberapa paragraf dan kalimat pendek atau kata. Data teks tersebut berupa dokumen dengan format *.txt yang nantinya akan di uji keefisienannya dengan menggunakan dua algoritma kompresi teks, yaitu algoritma Huffman dan algoritma Shannon-Fano.

Gambar 3.1 *Proses Pengumpulan Data*

Adapun metodologi rekayasa perangkat lunak yang digunakan adalah *Classic life cycle*. Nama model ini sebenarnya adalah "*Linear Sequential Model*". Model ini sering disebut dengan "*classic life cycle*" atau model *waterfall*. Model ini adalah suatu paradigma perangkat lunak yang menuntut suatu sistem yang sistematik, mulai dari suatu level sistem kemudian terus maju ke level berikutnya.

Tahapan dari model *waterfall* adalah sebagai berikut (Roger, 2001, h.28):

1. *System/Information Engineering*

   Pemodelan ini diawali dengan mencari kebutuhan dari keseluruhan sistem yang akan diaplikasikan ke dalam bentuk *software*. Hal ini sangat penting, mengingat *software* harus dapat berinteraksi dengan elemen-elemen yang lain seperti *hardware*, *database*, dan sebagainya.

2. *Analysis* (Analisis)

   Proses pencarian kebutuhan diintensifkan dan difokuskan pada *software*. Untuk mengetahui sifat dari program yang akan dibuat, maka para *software engineer* harus mengerti tentang domain informasi dari perangkat lunak, misalnya fungsi yang dibutuhkan, *user interface*, dan sebagainya.

3. *Design* (Desain)

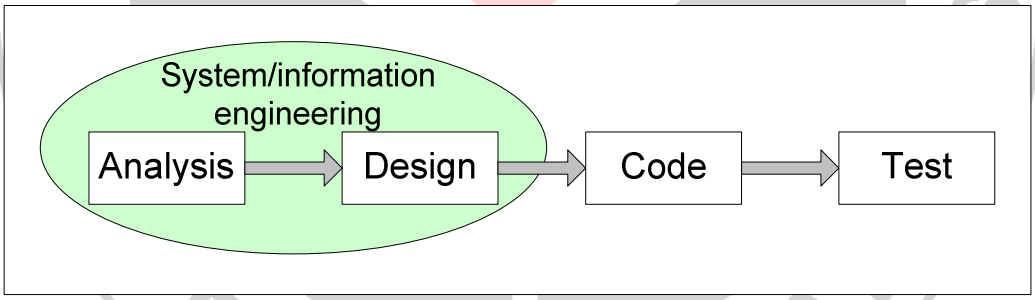
   Proses ini digunakan untuk mengubah kebutuhan-kebutuhan diatas menjadi representasi ke dalam bentuk bagan perangkat lunak sebelum pengkodean dimulai. Desain harus dapat mengimplementasikan kebutuhan yang telah disebutkan pada tahap sebelumnya. Seperti dua aktivitas sebelumnya, maka proses ini juga harus didokumentasikan sebagai konfigurasi dari perangkat lunak.

4. *Coding* (Pengkodean)

   Desain program harus diterjemahkan ke dalam bentuk program yang dimengerti komputer. *Coding* merupakan tahap berikutnya setelah tahap desain selesai. Pada tahap ini dilakukan konversi dari hasil rancangan (spesifikasi program) menjadi *source code*.

5. *Testing* (Pengujian)

   Setelah program dapat berjalan, selanjutnya dilakukan pengujian dengan memfokuskan pada logika internal dari perangkat lunak, fungsi eksternal dan mencari segala kemungkinan masalah. Selanjutnya memeriksa apakah perangkat lunak sudah sesuai dengan yang diharapkan atau belum. Pengujian merupakan proses untuk eksekusi program yang telah selesai dibuat untuk memeriksa apakah terdapat kesalahan atau tidak.



Gambar 3.2 *Waterfall Model*

3.3 Deskripsi Umum Sistem

Kompresi data teks merupakan proses memperkecil ukuran data teks yang berukuran besar menjadi lebih kecil dari ukuran aslinya, namun tidak mengurangi informasi yang terkandung di dalam data teks tersebut. Kebalikan dari kompresi adalah dekompresi yang merupakan proses pengembalian data teks yang sudah

dikompresi menjadi data yang utuh atau kembali menjadi data sebelum dilakukan kompresi. Tujuan dari proses kompresi ini yaitu menghemat tempat penyimpanan data teks yang terlalu banyak dan menghemat waktu dalam melakukan transfer data teks yang berukuran besar.

Manfaat dilakukan kompresi dapat menjadikan lebih efisien dan meminimalisasi dalam penyimpanan data. Hal ini akan dirasakan apabila data teks yang dimiliki sangat banyak dan semakin lama semakin menumpuk. Kompresi juga sangat membantu bagi pengguna yang sering melakukan pengiriman dan penerimaan data melalui jaringan internet, data teks yang sudah dikompresi akan lebih irit waktu apabila dibandingkan dengan data teks yang tidak dikompresi terlebih dahulu.

Metode yang digunakan dalam pengembangan perangkat lunak ini yaitu *Linear Sequential Model* atau model *waterfall*. Dengan menggunakan model ini, dalam pengerjaan perangkat lunak dilakukan secara berurutan mulai dari analisis, desain, pengkodean sampai pengujian.

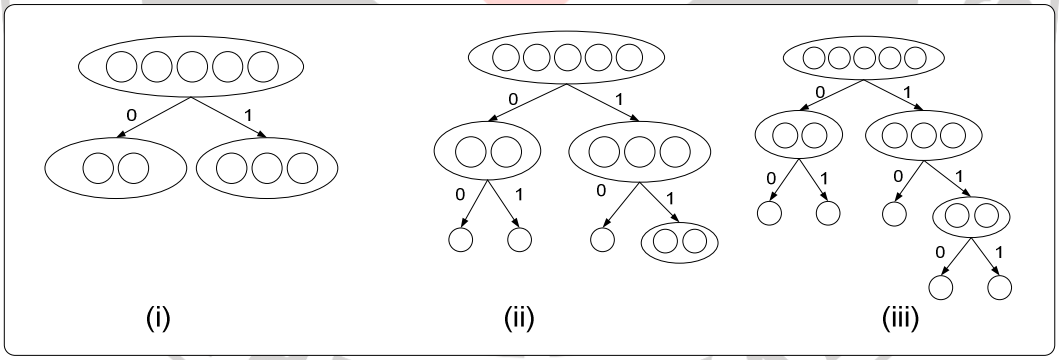
Aplikasi kompresi teks ini dibuat dengan menggunakan dua algoritma, yaitu algoritma Shannon-Fano dan algoritma Huffman. Perbedaan dari kedua algoritma ini yaitu dari pendekatan dalam pembuatan pohon biner.

1. Algoritma Shannon-Fano

   Algoritma Shannon-Fano ini dikembangkan oleh Claude Shannon dan Robert Fano. Seperti yang telah dijelaskan sebelumnya, prinsip kerja kompresi teks yaitu karakter yang memiliki frekuensi kemunculan lebih banyak akan membentuk kode bit yang lebih sedikit, sedangkan karakter yang memiliki

frekuensi kemuculan lebih sedikit akan membentuk kode bit yang lebih banyak. Algoritma Shannon-Fano ini menghasilkan sebuah kode dengan jumlah bit yang lebih sedikit untuk setiap karakter dengan membangun suatu pohon biner dibandingkan menggunakan kode yang mempunyai panjang tertentu seperti kode ASCII. Dalam pembuatan pohon biner, algoritma Shannon-Fano menggunakan pendekatan *top-down*.

Pembuatan pohon biner Shannon-Fano diawali dengan mengurutkan karakter berdasarkan frekuensi kemunculan dari terbesar sampai yang terkecil. Kemudian membagi dua bagian dengan bobot atau jumlah frekuensi kemunculan karakter yang hampir mendekati. Daun yang terdiri lebih dari satu karakter, dibagi lagi hingga daun berdiri sendiri dan mempunyai kode biner yang unik.



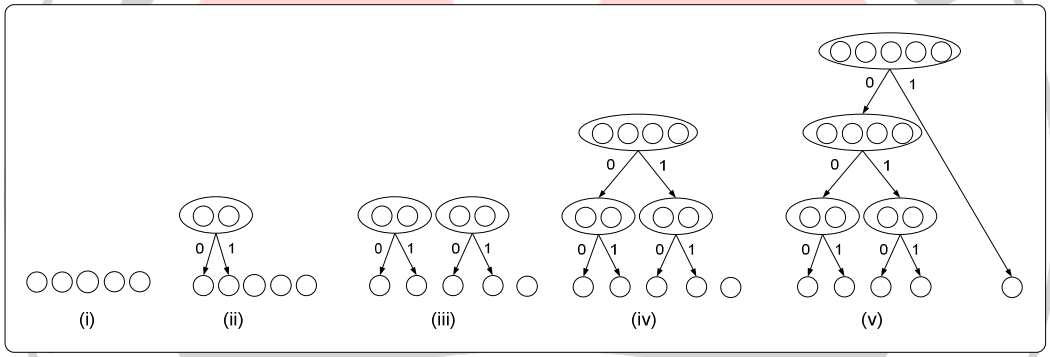
Gambar 3.3 *Pembuatan Pohon Biner Shannon-Fano*

2. Algoritma Huffman

Algoritma Huffman ini dikembangkan oleh David Huffman. Pada sejarahnya, Huffman sudah tidak dapat membuktikan apapun tentang kode yang efisien, tapi ketika tugasnya hampir selesai, ia mendapatkan ide

menggunakan pohon biner untuk menyelesaikan masalahnya mencari kode yang efisien. Pohon biner Huffman ini mempunyai kemiripan dengan pohon biner Shannon-Fano, yang membedakan yaitu pendekatan dalam pembuatan pohon biner. Pohon biner Huffman menggunakan pendekatan *a bottom-up*.

Pembuatan pohon biner Huffman diawali dengan pengurutan karakter berdasarkan frekuensi kemunculan. Kemudian dimulai dari daun, dilakukan penggabungan dua karakter dengan frekuensi kemunculan terkecil. Hal ini terus dilakukan hingga terbentuk akar. Hasil dari pembuatan pohon ini akan terbentuk kode biner unik dan bukan kode awalan dari karakter lainnya.



Gambar 3.4 *Pembuatan Pohon Biner Huffman*