BAB III

ANALISIS DAN PERANCANGAN

III.1. Analisis Masalah

Saat ini masyarakat masih kurang memahami mengenai bahasa daerah yang ada di Indonesia salah satunya bahasa Jawa. Bahasa Jawa merupakan salah satu bahasa daerah yang ada di Indonesia, mungkin sebagian masyarakat di Indonesia belum memahami mengenai bahasa Jawa. Oleh karena itu, penulis merasa ingin membuat suatu aplikasi bahasa Jawa ke bahasa Indonesia untuk membantu masyarakat yang belum memahami mengenai bahasa Jawa tersebut agar dapat sedikit memahami bahasa jawa.

III.2. Analisis Kebutuhan Sistem

Pada saat ini pembuatan aplikasi penerjemah Bahasa Jawa kedalam Bahasa Indonesia ini diperlukan beberapa aspek yaitu aspek perangkat keras dan aspek perangkat lunak.

a). Aspek Perangkat Keras

Spesifikasi perangkat keras yang dibutuhkan yaitu :

1) Komputer/PC minimal Intel Dual Core
2) Memori DDR3 2 *Gbyte*
3) Smartphone Android versi 5.1.1

b). Aspek Perangkat Lunak

Adapun spesifikasi perangkat lunak yang dibutuhkan yaitu:

1) Sistem Operasi Windows 7
2) *Eclipse* dan *Database SQLite*

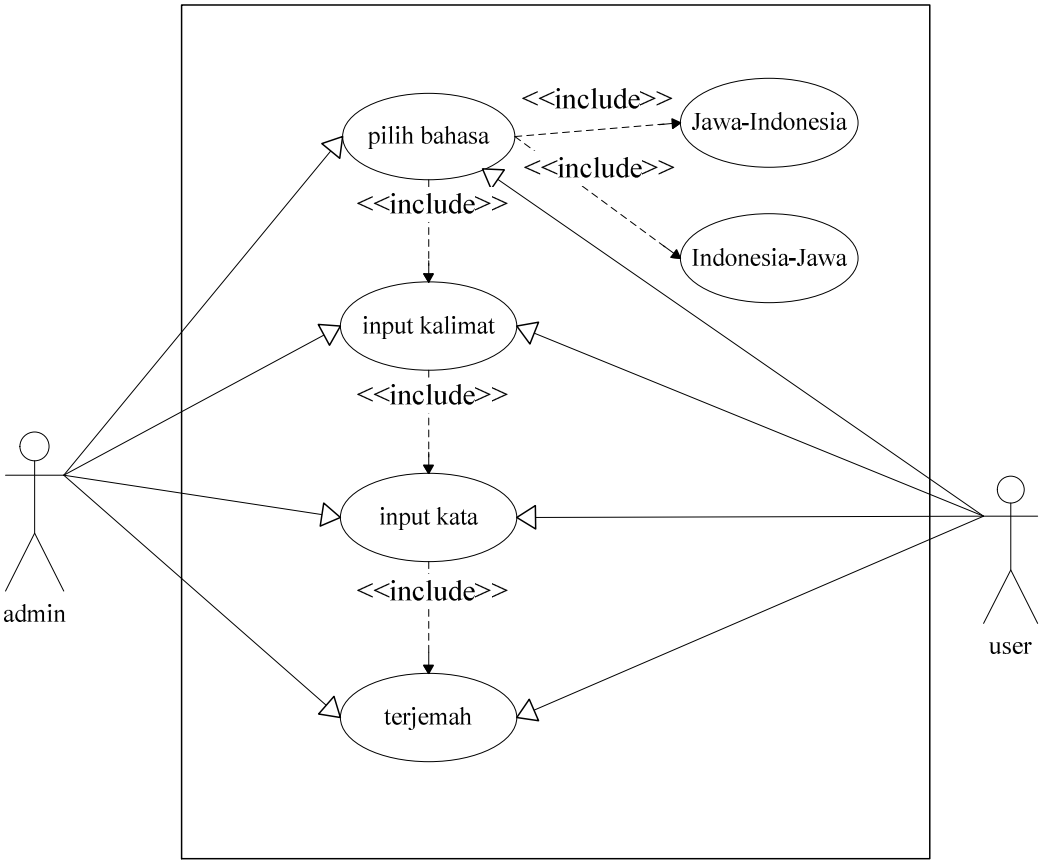
III.3. Perancangan Sistem

Perancangan sistem adalah sebuah teknik pemecahan masalah yang saling melengkapi (dengan analisis sistem) yang merangkai kembali bagian – bagian komponen sebuah sistem yang lengkap. Perancangan sistem dilakukan untuk memberikan gambaran dan mempermudah dalam melakukan implementasi ataupun evaluasi terhadap sistem yang akan dibangun. Pada perancangan sistem terdapat *use case diagram, activity diagram, sequence diagram* dan perancangan antarmuka (*interface*).

III.3.1. *Use Case Diagram*

Use Case Diagram adalah suatu kegiatan atau interaksi yang saling berkaitan antara sistem dan aktor. *Use case* bekerja dengan cara mendeskripsikan tipe interaksi antara user sebuah sistem dengan sistemnya itu sendri melalui sebuah cerita bagaimana sebuah sitem dipakai.

Use Case pada Gambar III.1 menjelaskan bahwa *user* dapat melakukan proses dengan menentukan bahasa yang akan dicari terrjemahannya, kemudian melakukan *input* data, setelah *input* data, ada proses identifikasi *pattern* yang dilakukan oleh algoritma *Knuth Morris Pratt* yaitu pencarian *string* dari kiri ke kanan, setelah itu didapatkan hasil pencarian.



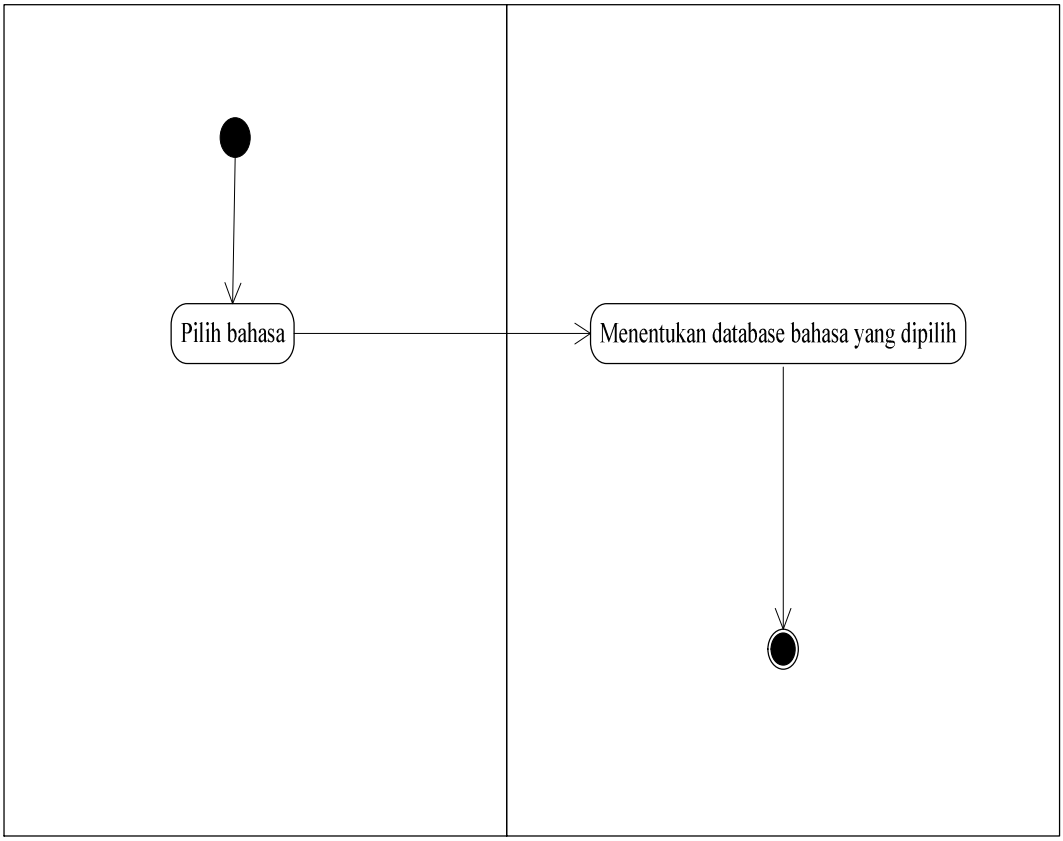
Gambar III.1. *Use Case Diagram* Pada Sistem

III.3.2. *Activity Diagram*

Activity Diagram menggambarkan berbagai alur kerja yang berisi aktivitas dan tindakan dalam suatu sistem yang sedang di rancang.

1. *Activity Diagram* pilih bahasa

Adapaun diagram aktivitas pemilihan bahasa pada sistem dapat dilihat pada gambar III.2.

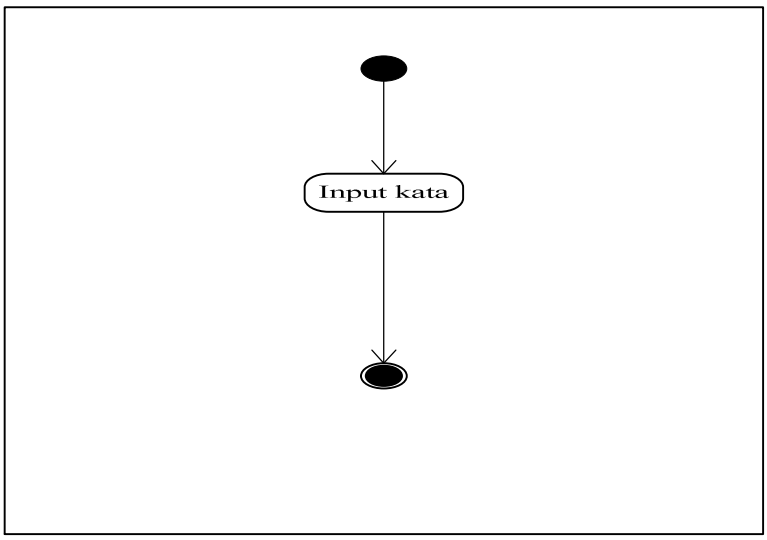


Gambar III.2. *Activity Diagram* Pilih Bahasa

Activity Diagram pada gambar III.2. Menjelaskan aktifitas pemilihan bahasa dimana *user* memilih bahasa yang akan di terjemahkan kemudian sistem memilih *database* bahasa yang dipilih oleh *user*.

2. *Activity Diagram* input kata

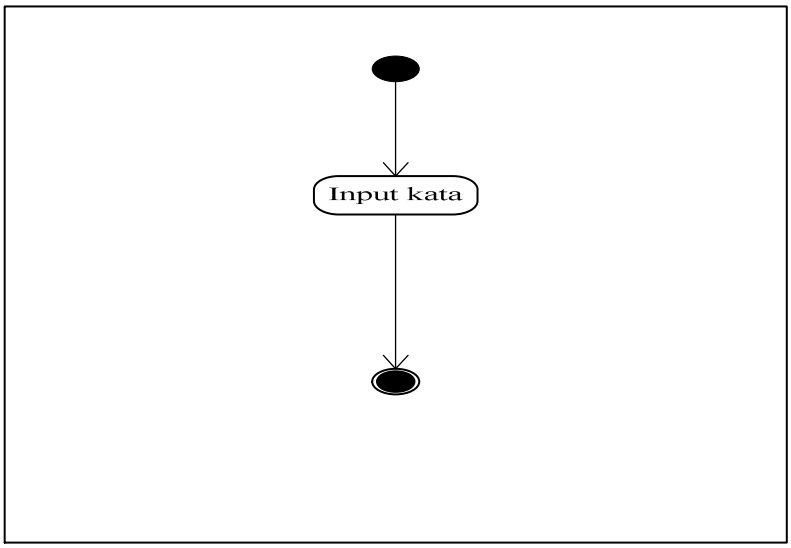
Berikut ini adalah aktivitas input kata dapat dilihat pada gambar III.3.



Gambar III.3. *Activity Diagram* Input Kata

3. *Activity Diagram* input kata

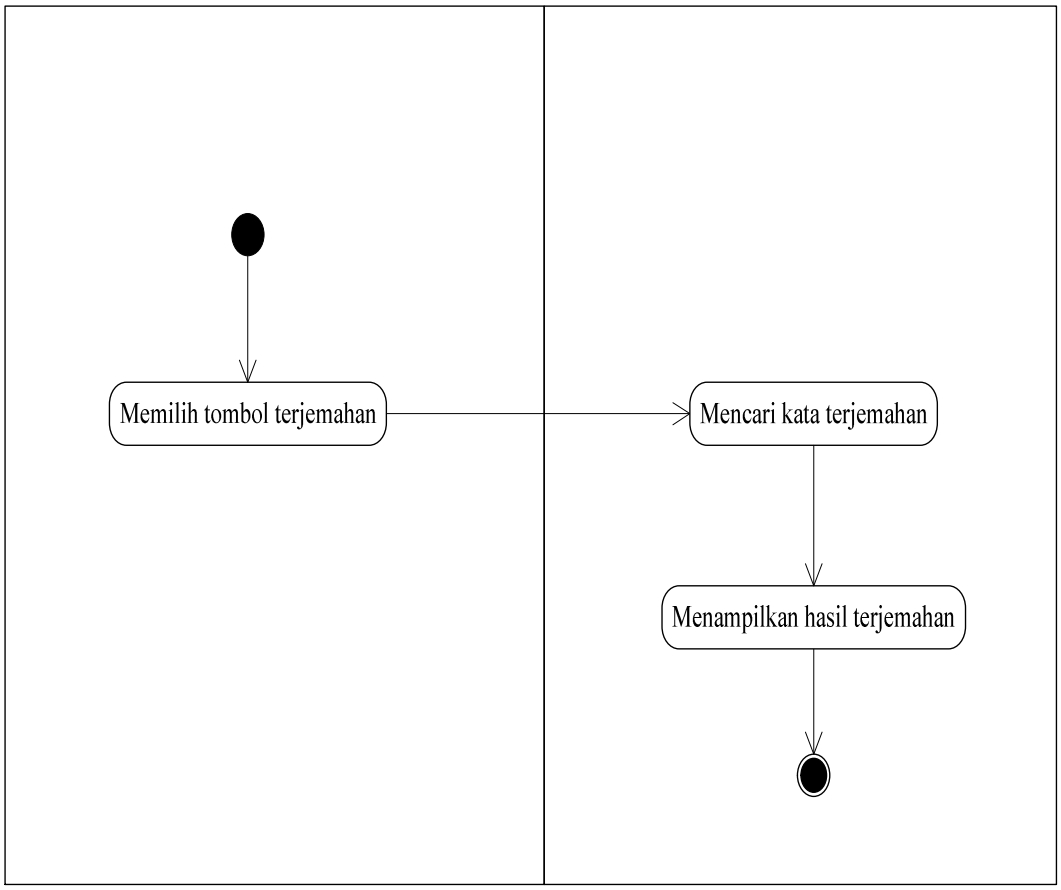
Berikut ini adalah aktivitas input kata dapat dilihat pada gambar III.4.



Gambar III.4. *Activity Diagram* Input Kalimat

4. *Activity Diagram* terjemahan

Adapun aktivitas proses penerjemahan kata yang diinput oleh *user* dapat dilihat pada gambar III.5.



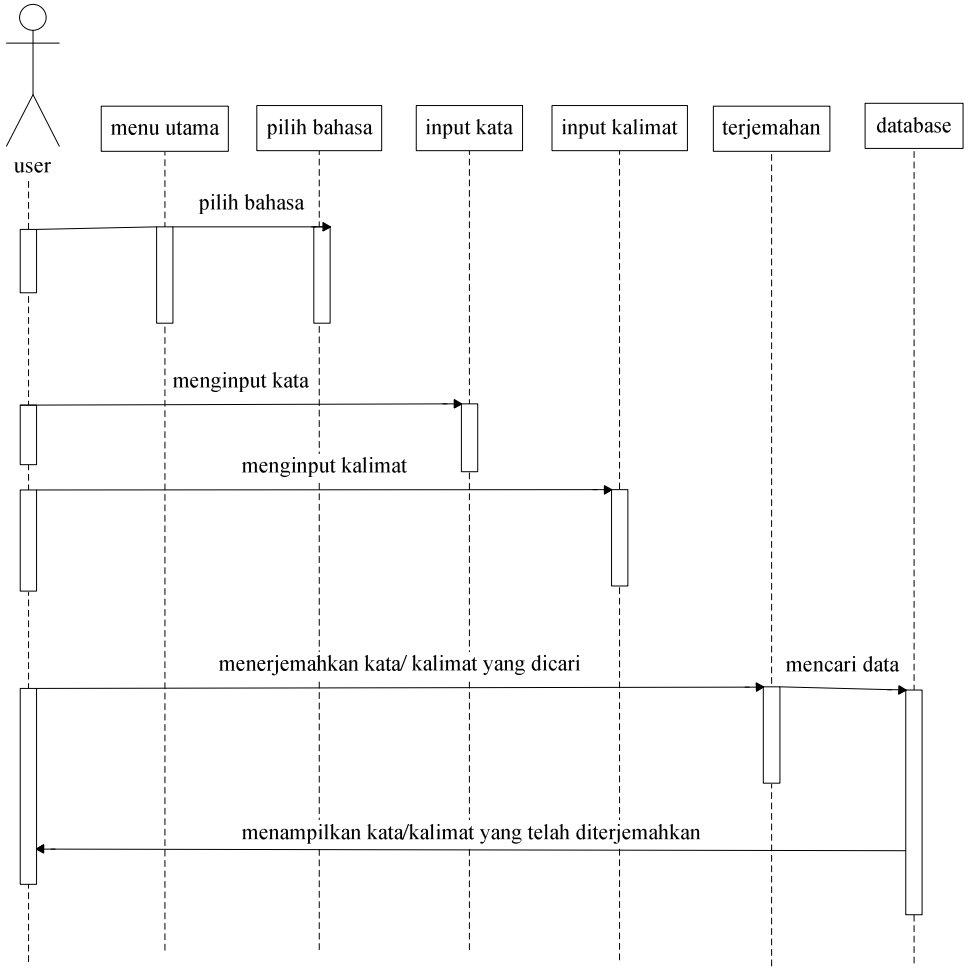
Gambar III.5. *Activity Diagram* Terjemahan

Activity Diagram pada gambar III.5. Menjelaskan aktivitas pada proses penerjemah yang dilakukan oleh *user* dengan memilih tombol terjemahan dan sistem mencari kata terjemahan yang cocok di dalam *database* dan kemudian sistem menampilkan hasil dari terjemahan kata tersebut.

III.3.3. *Sequence Diagram*

Sequence diagram adalah *diagram* yang mengambarkan kolaborasi dinamis antara sejumlah *object*. Kegunaannya untuk menunjukan rangkaian pesan yang dikirim antara object juga interaksi antara *object*.

Pada Gambar III.6. memiliki empat objek yaitu *user*, menu utama, input kata, ipnut kalimat,terjemah,dan *database* yang digambarkan dengan simbol objek UML. Referensi pada *use case* digambarkan dengan *lifeline* – garis vertical putus – putus. Kotak persegi empat yang berada pada masing masing objek merupakan *behavior* atau operasi yang perlu dilakukan oleh masing masing objek. Kotak – kotak tersebut menggambarkan kode program. Kemudian, anak panah antara garis menggambarkan interaksi atau pesan yang telah dikirim ke objek tertentu untuk menginvokasi salah satu operasinya untuk memenuhi permintaan.



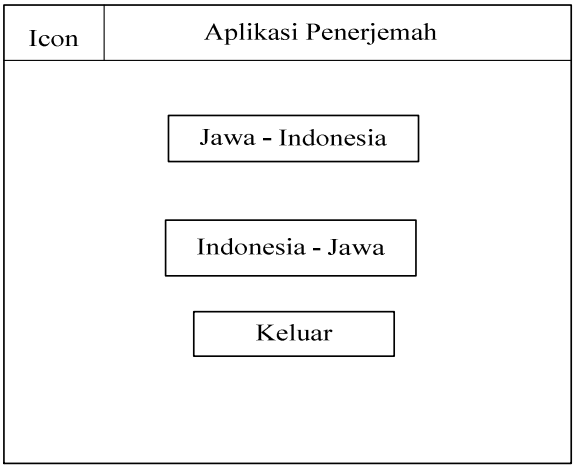
Gambar III.6. *Sequence Diagram* Aplikasi penerjemah

III.4. Perancangan Antarmuka (*Interface*)

Secara umum, sistem yang akan dibangun memiliki beberapa halaman sebagai berikut :

A. Halaman Awal

Halaman awal merupakan tampilan awal ketika aplikasi pertama kali dijalankan. Rancangan halaman utama aplikasi yang akan dibuat dapat dilihat pada Gambar III.7.



Gambar III.7. Rancangan Tampilan Awal

B. Halaman Jawa – Indonesia

Halaman Jawa – Indonesia, halaman ini menampilkan tempat menerjemahkan sebuah kalimat yang ingin di-*input* oleh *user* (pengguna). Halaman ini dapat dilihat pada Gambar III.8.

Gambar III.8. Rancangan Tampilan Penerjemah Jawa - Indonesia

C. Halaman Indonesia – Jawa

Halaman Indonesia - Jawa, halaman ini menampilkan tempat menerjemahkan sebuah kalimat yang ingin di-*input* oleh *user* (pengguna). Halaman ini dapat dilihat pada Gambar III.9.

Gambar III.9. Rancangan Tampilan Penerjemah Indonesia - Jawa

III.5 Pengujian *Knuth Morris Pratt*

Database aplikasi penerjemah ini memiliki 1000 kosa kata yang akan di cek pada saat pengguna memasukkan kata yang ingin dicari terjemahannya. Tabel III.1. adalah sampel kata yang ada di dalam database untuk melakukan tes pencarian secara manual. Kata yang menjadi *pattern* adalah "Ayam".

Tabel III.1. Sampel Kata di dalam *Database*

No.	Kata
1.	Api
2.	Air
3.	Ayam

Teks = Beli Ayam

Pattern = Ayam

Secara sistematis, langkah-langkah yang dilakukan algoritma *Knuth Morris Pratt* pada saat mencocokan *pattern* adalah:

1. Algoritma *Knuth Morris Pratt* mulai mencocokkan *pattern* pada awal teks.
2. Dari kiri ke kanan, algoritma ini akan mencocokkan karakter per karakter *pattern* dengan karakter di teks yang bersesuaian, sampai salah satu kondisi berikut dipenuhi:
   a. Karakter di *pattern* dan di teks yang dibandingkan tidak cocok (*mismatch*).
   b. Semua karakter di *pattern* cocok, kemudian algoritma akan memberitahukan penemuan di posisi ini.
3. Algoritma kemudian menggeser *pattern* berdasarkan tabel next, lalu mengulangi langkah 2 sampai *pattern* berada di ujung teks.

a. Beikut adalah proses algoritma *Knuth Morris Pratt* untuk pencarian *pattern* dalam teks:

Tabel III.2. Tahap Pertama Pencocokan *Pattern*

Teks	B	E	L	I		A	Y	A	M
Pattern	A	Y	A	M					

Dalam proses pencocokan pertama tidak ditemukan kecocokan antara *pattern* dengan teks tersebut, maka *pattern* akan bergeseser satu posisi ke kanan.

Tabel III.3. Tahap Kedua Pencocokan *Pattern*

Teks	B	E	L	I		A	Y	A	M
Pattern		A	Y	A	M				

Dalam proses pencokokan kedua masih belum ditemukan kecocokan antara *pattern* dengan teks tersebut, maka *pattern* bergeser satu posisi ke kanan.

Tabel III.4. Tahap Ketiga Pencocokan *Pattern*

Teks	B	E	L	I		A	Y	A	M
Pattern			A	Y	A	M			

Dalam proses pencocokan ketiga masih belum ditemukan kecocokan antara *pattern* dengan teks tersebut, maka *pattern* bergeser satu posisi ke kanan.

Tabel III.5. Tahap Keempat Pencocokan *Pattern*

Teks	B	E	L	I		A	Y	A	M
Pattern				A	Y	A	M		

Dalam proses pencocokan keempat masih belum ditemukan kecocokan antara *pattern* dengan teks tersebut, maka *pattern* bergeser satu posisi ke kanan.

Tabel III.6. Tahap Kelima Pencocokan *Pattern*

Teks	B	E	L	I		A	Y	A	M
Pattern					A	Y	A	M	

Dalam proses pencocokan kelima masih belum ditemukan kecocokan antara *pattern* dengan teks tersebut, maka *pattern* bergeser satu posisi ke kanan

Tabel III.7. Tahap Keenam Pencocokan *Pattern*

Teks	B	E	L	I		A	Y	A	M
Pattern						A	Y	A	M

Dalam proses pencocokan keenam akhirnya ditemukan kecocokan *pattern* pada teks tersebut, maka algoritma *Knuth Morris Pratt* akan menyimpan informasi dan *pattern* tidak akan melakukan pergeseran kembali.

b. Dalam contoh lain algoritma *Knuth Morris Pratt* memiliki perhitungan khusus dalam melakukan pergeseran *string* dengan menggunakan fungsi pinggiran. Misalkan untuk pencocokan kata "abcabd" dalam "abcabceabcd" prosesnya adalah sebagai berikut:

   P = " ABCABD"

   S = "ABCABCABD"

Langkah pertama kita akan mencari fungsi pinggiran dari *pattern*. Berikut ini adalah tabel nilai fungsi pinggiran dari *pattern*.

Tabel III.8. Fungsi Pinggiran *Pattern*

P	A	B	C	A	B	D
I	0	0	0	1	2	3

Langkah kedua samakan ujung kiri *pattern* dengan ujung kiri sample.

Tabel III.9. Proses Pertama Pergeseran Dengan Fungsi Pinggiran *Pattern*

S	A	B	C	A	B	C	A	B	D
P	A	B	C	A	B	D			

Karakter – karakter pada posisi 1-5 sama, tetapi pada posisi ke-6 karakter S dan P berbeda. Maka akan dilakukan pergeseran dengan memperhatikan fungsi awalan dari P yang bersesuaian. Awalan ysng bersesuain adalah ABCAB yang memiliki panjang 1 =5. Pinggiran terpanjang dari P[1..5] adalah AB yang panjangnya adalah 2. Maka jarang pergeseran adalah 5 – 2 = 3 . Jadi *pattern* yang akan di geser sejauh 3 karakter.

Tabel III.10. Proses Kedua Pergeseran Dengan Fungsi Pinggiran *Pattern*

S	A	B	C	A	B	C	E	A	D
P				A	B	C	A	B	D

Langkah selanjutnya akan dicocokan P dan S mulai dari karakter ke-4 hingga ke-9 memiliki kesamaan dengan karakter S. Maka dari itu akan mendapatkan bahwa P cocok dengan S.