BAB III
ANALISIS

3.1 Analisis Masalah

Berdasarkan hasil wawancara, dilingkungan perpustakaan dan peneliti Pusat Penelitian dan Pengembangan Sumber Daya Air untuk menentukan kata kunci dokumen masih dilakukan dengan membaca ulang isi dari abstrak dokumen kemudian baru ditentukan kata kuncinya. Terdapat sebagian dokuman yang belum diberikan kata kunci. Oleh karena itu, pegawai perpustakaan mengalami kesulitan dalam menentukan kata kunci yang tepat untuk dokumen yang belum memiliki kata kunci.

Kata kunci mencerminkan isi dari keseluruhan dokumen. Dengan kata kunci kita dapat mengetahui secara abstrak isi dari dokumen tersebut sebelum kita membaca keseluruhan dokumen. Biasanya seseorang yang ingin mencari data dari dokumen, orang tersebut tidak akan membaca secara keseluruhan dari dokumen itu. Ia akan melihat indeks atau kata kunci yang terkait dengan dokumen itu. Kemudian barulah membaca abstraknya. Jika yang dicari sesuai maka dokumen akan dibaca secara keseluruhan.

Abstrak adalah inti dari keseluruhan inti dokumen. Apabila kata kunci yang digunakan tidak sesuai dengan abstrak maka kata kunci tersebut tidak sesuai dengan isi dokumen.

Jadi, kata kunci dibutuhkan seseorang dalam pencarian suatu dokumen yang dicarinya. Dengan kata lain, kata kunci dapat mengarahkan kita kedokumen yang ingin dicari. Untuk itu pemilihan kata kunci yang tepat sangatlah penting dalam memberikan informasi mengenai dokumen itu dan seseorang dalam pencarian suatu dokumen melalui mesin pencari. Jika kata kunci yang dipilih tidak tepat maka dokumen yang dibutuhkan pula tidak terarahkan dengan baik melalui mesin pencarian.

Salah satu algoritma yang dapat digunakan dalam penentuan kata kunci adalah Algoritma *Baum Welch.* Algoritma *Baum Welch* merupakan algoritma yang

digunakan pada proses learning dengan mengoptimalkan nilai parameter Model Markov Tersembunyi atau *Hidden Markov Method* (HMM). Dalam menentukan rangkaian yang belum diketahui, algoritma *Baum Welch* mengoptimalkan nilai parameter yang diberikan.

Setiap algoritma memiliki performansi yang berbeda-beda. Nilai performansi suatu algortima dilihat dari kemangkusannya. Suatu algoritma dikatakan bagus apabila algoritma tersebut mangkus atau efisien. Salah satu melihat kemangkusan algoritma adalah dengan menghitung kompleksitas waktu asimptotik menggunakan notasi Big-O.

Berdasarkan permasalahan di atas maka diperlukan teknik pencarian kata kunci suatu dokumen dengan menganalisis performansi Algoritma *Baum Welch* dengan memaksimalkan nilai probabilitas dari kata yang terdapat dalam abstrak.

3.2 Analisis Algoritma

Algoritma merupakan sekumpulan langkah atau perintah untuk dapat menyelesaikan suatu masalah [18]. Suatu algortima dikatakan bagus apabila algortima tersebut mangkus atau efisien. Untuk menilai suatu Algoritma dikatakan bagus dapat menggunakan kemangkusan algortima. Untuk itu diperlukan analisis untuk mengukur performansi suatu algortima dengan mengukur kemangkusannya dengan menghitung nilai dari kompleksitas waktu asimptotiknya menggunakan notasi "O-Besar" (*Big-O*), dimana meningkatnya nilai n dapat mempengaruhi kebutuhan waktu Algoritma [7] [8].

3.1.1 Perhitungan kompleksitas waktu Asimptotik Algoritma *Baum Welch*

Perhitungan kompleksitas waktu asimptotik Algoritma *Baum Welch* dapat dihitung dengan menelusuri setiap langkah Algoritma *Baum Welch* pada *pseudocode*-nya. *Pseudocode* algoritma *Baum Welch* yang telah diberikan notasi Big-O disetiap langkahnya dapat dilihat pada Tabel III.1.

Tabel III.1*Pseudocode* Algoritma *Baum Welch* dengan notasi Big-O

Isi *pseudocode*	penanda
procedure BaumWelch(input jumlah_kata:integer, p:integer, p_paskata:LarikReal, p_kata:LarikReal, output A:LarikReal, B:LarikReal, Phi:LarikReal) {I.S : serangkaian *state*, jumlah_kata (n) sudah	

```
terdefinisi}                                                          |
{F.S: nilai probabilitas transisi, probabilitas emisi dan             |
state awal maksimum atau optimal}                                     |
Kamus:                                                                |
   Stop_iterasi :boolean                                              |
   i,j,k,t: integer                                                   |
   z: real                                                            |
Algoritma:                                                            |
   {langkah 1 : inisialisasi nilai N, M A, B dan π}                   |
   {inisialisasi N dan iterasi}                                       |
   N←jumlah_kata                                          O(1)        |
   iterasi ← 0 {inisialisasi iterasi bernilai 0}          O(1)        |
   InisialisasiParameterAwal(N,p,p_paskata,p_kata,A,B,Phi,            |
   stop_iterasi,prob_lama)                                O(N^2)  …1  |
   T←N {banyaknya waktu T sebanyak state observasi}       O(1)        |
   Repeat                                                             |
   {Langkah 2 : menghitung probabilitas forward dan                   |
backward}                                                             |
   ProbabilitasForward(N,A,B,Phi,alpha,lamda,prob_forward)            |
                                                          O(N^3)  …2  |
   ProbabilitasBackward(N,A,B,beta)                       O(N^3)  …3  |
   {Langkah 3 : menghitung kembali probabilitas transisi              |
dan emisi}                                                            |
    ParameterBaru(N,A,B,Phi,beta,alpha,A_bar,B_bar,                   |
    Phi_bar)                                                          |
                                                          O(N^3)  …4  |
   {menghitung P(O|λ̄)}                                                |
    ProbabilitasOLamda(N,A,B,Phi,gamma,epsilonl,                      |
    total_epsilon,total_gamma,prob_baru)                  O(N^3)  …5  |
     if ((iterasi≥1)&&((prob_baru < prob_lama) ||                     |
   (iterasi←batas_iterasi))) then                         O(1)        |
      stop_iteras←true                                    O(1)        |
   else                                                               |
      for i←1 to N do                                     O(N)        |
         for j←1 to N do                                  O(N)        |
           a[i][j] ← a_bar[i][j]                          O(1)        |
         endfor                                                       |
      endfor                                                          |
```

```
      for j←1 to N do                                          O(N)
         for k←1 to M do                                       O(N)
            b[j][k] ← b_bar[j][k]                              O(1)
         endfor
      endfor
      for i←1 to N do                                          O(N)
         phi[i] ← phi_bar[i]                                   O(1)
      endfor
      prob_lama ← prob_baru                                    O(1)
      iterasi ← iterasi + 1                                    O(1)
   endif
      {Langkah 4 : kembali kelangkah 2 sampai 3 hingga
nilai P(O|λ̄) konvergen atau mendekati 1}
   until (stop_iterasi ← true || iterasi = batas_iterasi)
                                                               O(N)      ...6
   {Langkah 5 :mendapatkan nilai probabilitas maksimal}
   if(prob_baru >= prob_lama) then                             O(1)
      for i←1 to N do                                          O(N)
         for j←1 to N do                                       O(N)
            a[i][j] ← a_bar[i][j]                              O(1)
         endfor
      endfor
      for j←1 to N do                                          O(N)
         for k←1 to M do                                       O(N)
            b[j][k] ← b_bar[j][k]                              O(1)
         endfor
      endfor
      for i←1 to N do                                          O(N)
         phi[i] ← phi_bar[i]                                   O(1)
      endfor
   endif
endprocedure
```

Perhitungan kompleksitas waktu Asimptotik Algoritma *Baum Welch* dengan notasi Big-O adalah sebagai berikut :

– Perhitungan Big-O pada *pseudocode* yang ditandai kotak warna merah :

= O(1) + max((O(N).O(N).O(1)),(O(N).O(N).O(1)),(O(N).O(1)))

$= O(1) + \max(O(N^2),O(N^2),O(N))$

$= O(1) + O(N^2)$

$= \max(O(1),O(N^2))$

$= O(N^2)$

– Perhitungan nilai Big-O pada *pseudocode* yang ditandai kotak warna biru :

$= \max (O(1),\max((O(N).O(N).O(1)),(O(N).O(N).O(1)),(O(N).O(1))))$

$= \max (O(1),\max(O(N^2),O(N^2),O(N)))$

$= \max (O(1),O(N^2))$

$= O(N^2)$

– Nilai kompleksitas waktu asimptotik Algoritma *Baum Welch*

$= \max(O(1),O(1), O(1),O(N^2),(\max(O(1),O(N^3),O(N^3),O(N^3),O(N^3),O(N^2)).O(N))$
$,O(N^2))$

$= \max(O(1),O(1), O(1),O(N^2),(O(N^3).O(N)),O(N^2))$

$= \max(O(1),O(1), O(1),O(N^2),O(N^4),O(N^2))$

$= \mathbf{O(N^4)}$ **…6**

Berdasarkan perhitungan kompleksitas waktu asimptotik dengan notasi Big-O yang diuraikan diatas, maka Algoritma *Baum Welch* memiliki kompleksitas waktu dengan nilai Big-O adalah $\mathbf{O(N^4)}$. Berdasakan nilai laju pertumbuhan N, Algoritma *Baum Welch* termasuk dalam kelompok algoritma polinomial dengan derajat polinomial adalah 4. Algoritma polinomial termasuk algoritma yang bagus atau mangkus dengan syarat polinomial berderajat rendah. Algoritma *Baum Welch* memiliki derajat yang rendah yaitu 4. Parameter-parameter yang dapat mempengaruhi performansi Algoritma *Baum Welch* yang dapat dilihat pada *pseudocode*-nya adalah sebagai berikut:

1. N merupakan jumlah *state* atau jumlah kata pada abstrak tanpa pengulangan kata.
2. T merupakan variable waktu (sejumlah dengan nilai jumlah kata pada abstrak tanpa pengulangan kata).
3. I merupakan jumlah batas maksimal iterasi yang dilakukan.

Maka besarnya nilai waktu kompleksitas akan dipengaruhi tiga parameter yaitu jumlah kata pada abstrak tanpa pengulangan, variable T dan jumlah iterasi atau pengulangan.

Pada *pseudocode* Algoritma *Baum Welch*, masih terdapat beberapa prosedur didalamnya yang dapat diuraikan kembali. Berikut adalah beberapa prosedur dengan notasi Big-O dan perhitungan kompleksitas waktunya :

1. *Pseudocode* prosedur inisialisasi parameter awal dapat dilihat pada Tabel III.2 beserta notasi Big-O dan perhitungan waktu kompleksitasnya.

Tabel III.2 Prosedur inisialisasi parameter awal

Isi *pseudocode*	
procedure InisialisasiParameterAwal(input N:integer,p:integer, p_paskata:LarikReal, p_kata:LarikReal, output A:LarikReal, B:LarikReal, Phi:LarikReal,stopiterasi:Boolean, prob_lama:real)	
{I.S : jumlah kata telah terdefinisi}	
{F.S: mendapatkan nilai $P(O\|\lambda)$} awal, nilai parameter awal yaitu probabilitas transisi (A), probabilitas emisi (B) dan *state* awal (phi)}	
Kamus:	
i,j,k,b1: integer	
Algoritma:	
{inisialisasi nilai awal probabilitas A={a_{ij}} }	
for i←1 to N do	O(N)
for j←1 to N do	O(N)
a[i][j]←(p*p_paskata[i][j])+((1-p)*p_kata[j])	O(1)
endfor	
endfor	
{inisialisasi nilai awal probabilitas B={$b_j(Ot)$} }	
for j←1 to N do	O(N)
for k←1 to N do	O(N)
b1 ← p_kata[j]	O(1)
if (k=j) then	O(1)
b[j][k] ← b1	O(1)
else	
b[j][k] ← (1-b1)/(N-1)	O(1)
endif	

```
      endfor
   endfor
   {inisialisasi nilai state awal phi }
   for i←1 to N do                                          O(N)
      phi[i] ← 1 / N                                        O(1)
   endfor
   stop_iterasi ← false {inisialisasi  iterasi belum berhenti}
                                                            O(1)
   prob_lama ← prob_baru {inisialisasi nilai P(O|λ)}        O(1)
endprocedure
```

Perhitungan waktu kompleksitas pada prosedur inisialisasi parameter awal :

=

max((O(N).O(N).O(1)),(O(N).O(N).max(O(1),O(1),O(1),O(1))),(O(N).O(1)),O(1),O(1))

= max((O(N).O(N).O(1)),(O(N).O(N).O(1)),(O(N).O(1)),O(1),O(1))

= max($O(N^2)$,$O(N^2)$,O(N),O(1),O(1))

= $O(N^2)$ …1

2. *Pseudocode* prosedur probabilitas forward dapat dilihat pada Tabel III.3 beserta dengan notasi Big-O dan perhitungan kompleksitas waktunya.

Tabel III.3 *Pseudocode* prosedur probabilitas forward

Isi *pseudocode*
Procedure ProbabilitasForward(input N:integer, A:LarikReal, B:LarikReal, Phi:LarikReal, output alpha:LarikReal, lamda:LarikReal, prob_forward:real) {I.S : probabilitas transisi, emisi dan *state* awal telah terdefinisi} {F.S: mendapatkan probabilitas forward} **Kamus:** i,t,z: integer **Algoritma:** {Langkah 1 : Inisialisasi} for i←1 to N do O(N) {nilai lamda awal dari nilai probabilitas awal dengan t=1} alpha[1][i] ← phi[i]*b[i][1] {b[i][t]=$b_i(O_t)$} O(1)

```
   endfor

   {langkah 2 : Induksi}
   {menghitung nilai lamda pada waktu t=1 sampai t=T-1}
   for t←2 to T do                                              O(N)
      for j←1 to N do                                           O(N)
         z ← 0                                                  O(1)
         for i←1 to N do                                        O(N)
            z ← ( alpha[t-1][i] * a[i][j] ) + z                 O(1)
         endfor
         lamda[t][j] ← z * b[j][t] {b[i][t]=b_i(O_t)}           O(1)
      endfor
   endfor

   {langkah 3 : Terminasi}
   {pada state terakhir T=N}
   for i←1 to N do                                              O(N)
      prob_forward ← lamda[T][i] + prob_forward {P(O|λ)}
   {∑_{i=1}^{N} α_T(i)}                                         O(1)
   endfor
endprocedure
```

Perhitungan waktu kompleksitas prosedur probabilitas forward :

= max((O(N).O(1)), (O(N).O(N).max(O(1),(O(N).O(1)),O(1))), (O(N).O(1)))

= max(O(N), (O(N).O(N).max(O(1),O(N),O(1))), O(N))

= max(O(N), (O(N).O(N).O(N)), O(N))

= max(O(N), $O(N^3)$, O(N))

= $O(N^3)$ …2

3. *Pseudocode* prosedur probabilitas backward dapat dilihat pada Tabel III.4 beserta notasi Big-O dan perhitungan kompleksitas waktunya.

Tabel III.4 *Pseudocode* prosedur probabilitas backward

Isi *pseudocode*
Procedure ProbabilitasBackward(input N:integer, A:LarikReal, B:LarikReal, output : A:LarikReal, B:LarikReal, output beta:LarikReal) {I.S : probabilitas transisi dan emisi telah terdefinisi}

```
{F.S: mendapatkan probabilitas  backward}
Kamus:
   i,t,j: integer
Algoritma:
  {Langkah 1 : Inisialisasi}
   for i←1 to N do                                          O(N)
      beta[T][i] ← 1/N                                      O(1)
   endfor
   {langkah 2 : Induksi}
   for t←T-1 down to 1 do                                   O(N)
      for i←1 to N do                                       O(N)
         beta[t][i] ← 0                                     O(1)
         for j←1 to N do                                    O(N)
            beta[t][i]←(a[i][j]*b[j][t+1]*beta[t+1][j])+
                       beta[t][i] {b[i][t]=b_i(O_t)}        O(1)
         endfor
      endfor
   endfor
endprocedure
```

Perhitungan kompleksitas waktu prosedur probabilitas backward :

= max((O(N).O(1)), (O(N).O(N).max(O(1),(O(N).O(1))))

= max(O(N), (O(N).O(N).max(O(1),O(N),O(1))), O(N)))

= max(O(N), (O(N).O(N).O(N)), O(N))

= max(O(N), O(N^3), O(N))

= O(N^3) …3

4. *Pseudocode* Prosedur Parameter Baru dapat dilihat pada Tabel III.5 beserta notasi Big-O dan perhitungan kompleksitas waktunya.

Tabel III.5 *Pseudocode* prosedur menghitung parameter baru

Isi *pseudocode*
Procedure ParameterBaru(input N:integer, A:LarikReal, B:LarikReal, Phi:LarikReal, beta:LarikReal , alpha:LarikReal, output A_bar:LarikReal, B_bar:LarikReal, Phi_bar:LarikReal) {I.S : probabilitas forward dan backward telah terdefinisi} {F.S: mendapatkan nilai parameter baru yaitu probabilitas transisi (A), probabilitas emisi (B) dan *state* awal (phi)}

```
Kamus:
   i,t,j,k,s: integer
Algoritma:
  {menghitung nilai ε(i,j)}
   total_epsilon ← 0                                          O(1)
   {menghitung nilai epsilon}
   for t←1 to T-1 do                                          O(N)
      sum_epsilon← 0                                          O(1)
      for i←1 to N do                                         O(N)
         for j←1 to N do                                      O(N)
            sum_epsilon←alpha[t][i]*a[i][j]*b[s][j][t+1]*
                        beta[t+1][j]                          O(1)
            total_epsilon← total_epsilon + sum_epsilon        O(1)
         endfor
      endfor
      for i←1 to N do                                         O(N)
         for j←1 to N do                                      O(N)
            epsilon[t][i][j]←(alpha[t][i]*a[i][j]*b[s][j][t+1]*
                             beta[t+1][j])/sum_epsilon        O(1)
         endfor
      endfor
   endfor
   {menghitung nilai gamma γ_t(i)}
   total_gamma ← 0                                            O(1)
   for t←1 to T do                                            O(N)
      sum_gamma ← 0                                           O(1)
      for i←1 to N do                                         O(N)
         sum_gamma ← alpha[t][i]*beta[t][i] + sum_gamma       O(1)
         total_gamma ← sum_gamma+total_gamma                  O(1)
      endfor
      for i←1 to N do                                         O(N)
         gamma[t][i] ← alpha[t][i]*beta[t][i] + sum_gamma     O(1)
      endfor
   endfor
   {menghitung Probabilitas transisi atau nilai a_ij}
   for i←1 to N do                                            O(N)
      for j←1 to N do                                         O(N)
```

```
            sum_gamma_bawah← 0                                          O(1)
            sum_epsilon_atas← 0                                         O(1)
            for t←1 to T-1 do                                           O(N)
               sum_gamma_bawah ← gamma[t][i]+sum_gamma_bawah             O(1)
               sum_epsilon_atas ←epsilon[t][i][j]+sum_epsilon            O(1)
            endfor
            a_bar[i][j] ← sum_epsilon_atas/sum_gamma_bawah              O(1)
         endfor
      endfor
      {Probabilitas emisi}
      for j←1 to N do                                                   O(N)
         for k←1 to M do                                                O(N)
            sum_gamma_atas ← 0                                          O(1)
            sum_gamma_atas ← 0                                          O(1)
            for t←1 to T do                                             O(N)
               sum_gamma_bawah ← gamma[t][j]+sum_gamma_bawah             O(1)
               {jika Ob[t] =Ob[j]}
               if (k←t) then                                            O(1)
                  sum_gamma_atas ← gamma[t][j]+sum_gamma_atas            O(1)
               endif
            endfor
            b_bar[j][k] ← sum_gamma_atas /sum_ gamma_bawah              O(1)
         endfor
      endfor
      {inisialisasi nilai state awal phi }
      for i←1 to N do                                                   O(N)
         phi_bar[i] ← gamma[1][i]                                       O(1)
      endfor
endprocedure
```

Perhitungan waktu kompleksitas prosedur menghitung parameter baru pada :

- waktu kompleksitas pada langkah menghitung epsilon

  = O(N).max(O(1),(O(N).O(N).max(O(1),O(1))),(O(N).O(N).O(1)))

  = O(N).max(O(1),(O(N).O(N).O(1)),O(N^2))

  = O(N).max(O(1),O(N^2),O(N^2))

  = O(N).O(N^2)

  = O(N^3)

- waktu kompleksitas pada langkah menghitung gamma
  = max(O(1),(O(N).max(O(1),(O(N).max(O(1),O(1))),(O(N).O(1)))))
  = max(O(1),(O(N).max(O(1),O(N)),O(N))))
  = max(O(1),(O(N).O(N)))
  = max(O(1),O(N^2))
  = O(N^2)
- waktu kompleksitas pada langkah menghitung probabilitas transisi
  =O(N).O(N).max(O(1),O(1),(O(N).max(O(1),O(1))),O(1))
  =O(N).O(N).max(O(1),O(1),(O(N).O(1)),O(1))
  =O(N).O(N).max(O(1),O(1),O(N),O(1))
  =O(N).O(N).(N)
  =O(N^3)
- waktu kompleksitas pada langkah menghitung probabilitas emisi
  =O(N).O(N).max(O(1),O(1),(O(N).max(O(1),O(1))),O(1))
  =O(N).O(N).max(O(1),O(1),(O(N).O(1)),O(1))
  =O(N).O(N).max(O(1),O(1),O(N),O(1))
  =O(N).O(N).O(N)
  =O(N^3)
- waktu kompleksitas pada langkah menghitung nilai phi
  = O(N).O(1)
  = O(N)

Nilai waktu kompleksitas pada prosedur ParameterBaru adalah
= max (O(1),(epsilon),(gamma),(transisi),(emisi),(phi))
= max (O(1), O(N^3), O(N^2), O(N^3), O(N))
= O(N^3) …4

5. *Pseudocode* prosedur menghitung $P(O|\lambda)$baru dapat dilihat pada Tabel III.6 beserta notasi Big-O dan perhitungan kompleksitas waktunya.

Tabel III.6 *Pseudocode* Menghitung $\boldsymbol{P(O|\lambda)}$baru

Isi *pseudocode*
Procedure ProbabilitasOLamda(input N:integer,A:LarikReal, B:LarikReal,Phi:LarikReal,gamma:LarikReal,epsilon:LarikReal,tota

```
l_epsilon:real,total_gamma:real, output prob_baru:real)
{I.S : nilai probabilitas transisi, emisi dan state awal yang
baru telah terdefinisi}
{F.S: mendapatkan nilai P(O|λ) baru}
Kamus:
   i,t,j,k: integer
Algoritma:
   P_forward←0                                                    O(1)
   for i←1 to N do                                                O(N)
      for j←1 to N do                                             O(N)
         for t←1 to T do                                          O(N)
            P_forward←((epsilon[t][i[j]/total_epsilon)*
                      (ln(a_bar[i][j])/ln(10)))+P_forward         O(1)
         endfor
      endfor
   endfor

   P_backward←0                                                   O(1)
   for i←1 to N do                                                O(N)
      for k←1 to M do                                             O(N)
         for t←1 to T do                                          O(N)
            if(k←t) then                                          O(1)
               P_backward←((gamma[t][j]/total_gamma)*
                     (ln(b_bar[i][j])/ln(10)))+ P_backward        O(1)
            endif
         endfor
      endfor
   endfor
   prob_baru ← P_forward+P_backward                               O(1)
endprocedure
```

Perhitungan kompleksitas waktu prosedur menghitung $P(O|\lambda)$baru :

=

max(O(1),(O(N).O(N).O(N).O(1)),O(1),(O(N).O(N).O(N).max(O(1),O(1))),O(1)

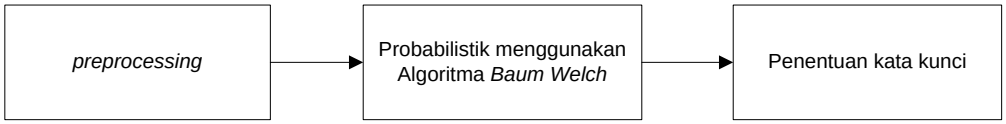
= max(O(1),O(N^3),O(1),(O(N).O(N).O(N).O(1)),O(1)

= max(O(1),O(N^3),O(1),O(N^3),O(1))

= O(N^3) …5

3.1.2 Tahapan Alur Kerja Menentukan Kata Kunci Terhadap Sampel Kasus

Block diagram menggambarkan setiap blok atau bagian dari alur kerja. *Block diagram* menentukan kata kunci menggunakan algoritma *Baum Welch* dapat dilihat pada Gambar III.1.

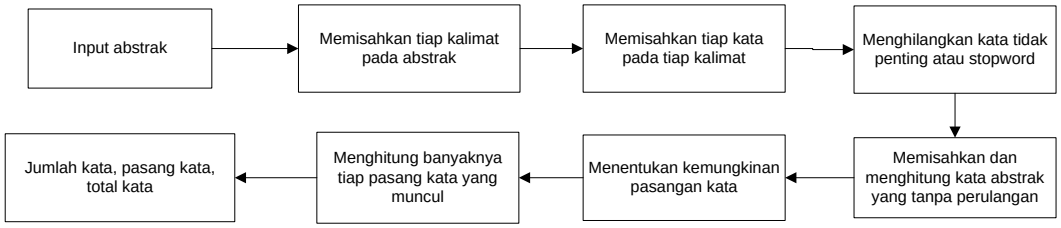


Gambar III.1 *Block diagram* menentukan kata kunci menggunakan algoritma *baum welch*

Berikut adalah penjelasan dari setiap proses pada *block diagram* Gambar III.1, yang terdiri dari proses *preprocessing*, proses menentukan probabilitas menggunakan Algoritma *Baum Welch* dan proses menentukan kata kunci.

3.1.1.1 *Preprocessing*

Preprocessing merupakan tahapan untuk mendapatkan data yang dibutuhkan pada proses Algoritma *Baum Welch*. Tahapan-tahapan dari *preprocessing* dapat dilihat pada Gambar III.2.



Gambar III.2 Block diagram preprocessing algoritma *baum welch*

1. **Input Abstrak**

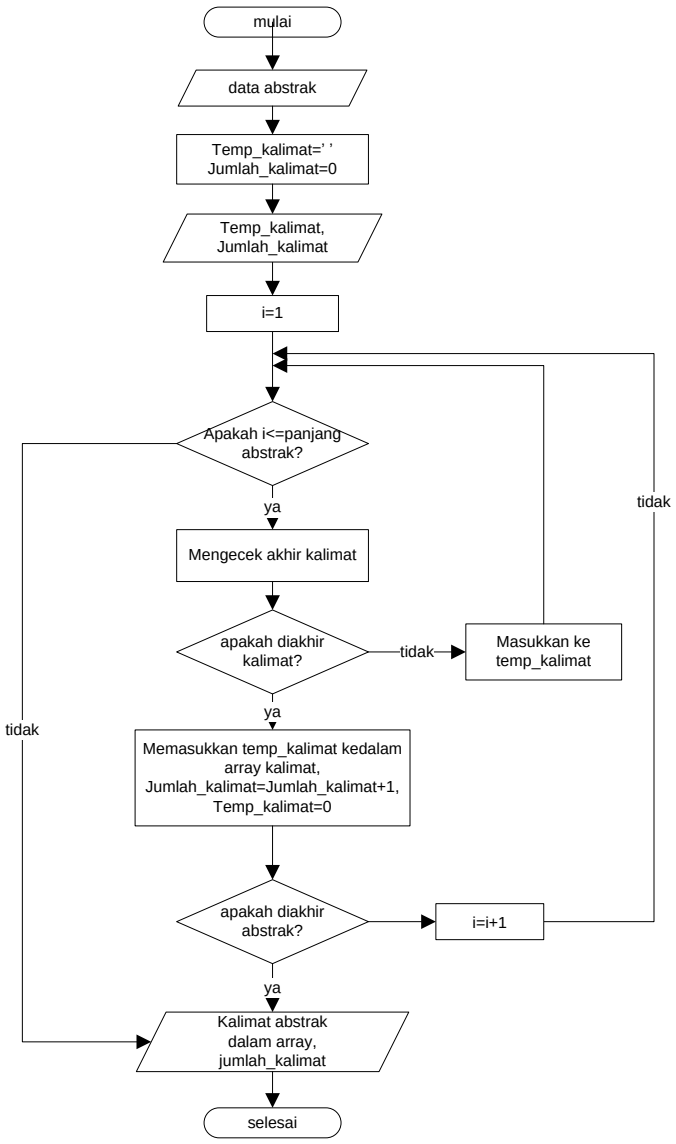
   Inputan yang dimasukkan berupa abstrak suatu dokumen. Abstrak yang diinputkan akan dihitung jumlah katanya tanpa pengulangan untuk dijadikan inputan pada Algoritma *Baum Welch*. Sebelum penghitungan jumlah kata, inputan abstrak terlebih dahulu dipisahkan setiap kalimatnya pada langkah kedua. salah satu Sampel abstrak yang digunakan adalah sebagai berikut :

Tabel III.7 Sampel abstrak yang dijadikan contoh kasus

Isi abstrak
waduk wonogiri adalah salah satu pengendali banjir di bengawan solo berfungsi sebagai pengendali banjir, waduk tersebut memiliki multi guna yaitu untuk air irigasi, pembangkit tenaga listrik, perikanan, pariwisata dll. mengingat das waduk wonogiri tataguna lahannya sudah banyak berubah, alih fungsi lahan dari hutan menjadi lahan pertanian akibat perkembangan kebutuhan manusia, maka terjadi laju erosi yang tinggi, dan akhirnya mengendap di waduk. hal ini umur waduk akan menjadi berkurang. salah satu pemasok sedimen terbesar adalah dari sub das keduang, selain luas das nya yang paling besar juga kondisinya sudah kritis.tindakan konservasi sangat di perlukan dalam pengendalian laju erosi. upaya reboisasi yang diterapkan harus secara efisien dan tepat guna. tujuan dari penelitian ini adalah untuk menetukan alternatif teknik reboisasi sehingga bisa menurunkan laju erosi dan aliran permukaan. untuk mengetahui laju erosi permukaan dianalisis dengan metode usle dan dikalibrasi dengan pengukuran langsung, sedangkan aliran permukaan dianalisis berdasarkan kondisi aktual dan setelah direboisasi dengan rumusan rational yang dikalibrasi pengukuran debit aliran secara langsung. berdasarkan hasil penelitian dapat diketahui bahwa reobisasi lahan tegal dan hutan dengan tanaman reboisasi terpilih merupakan skenario yang paling efisien dalam penangulangan erosi.

2. Memisahkan Tiap Kalimat Pada Abstrak

Langkah kedua adalah memisahkan tiap kalimat yang ada pada abstrak dengan menelusuri setiap huruf yang terdapat dapat asbtrak. Aturan pemisahan kalimat berdasarkan tanda baca titik (.), tanda seru (!) dan tanda tanya (?). Setiap kalimat disimpan ke dalam array. *Flowchart* pemisahan tiap kalimat abstrak dapat dilihat pada Gambar III.3.



Gambar III.3 *Flowchart* pemisahan kalimat pada abstrak

Jumlah kalimat yang terdapat pada abstrak adalah sejumlah 9, dimasukkan pada variabel jumlah_kalimat. Kalimat pada abstrak yang telah dipisahkan dapat dilihat pada Tabel III.8 adalah sebagai berikut :

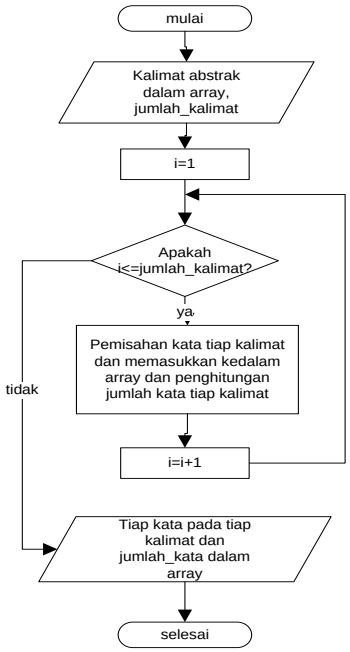
Tabel III.8 Pemisahan tiap kalimat pada abstrak

Kalimat ke-1	waduk wonogiri adalah salah satu pengendali banjir di bengawan solo berfungsi sebagai pengendali banjir, waduk tersebut memiliki multi guna yaitu untuk air irigasi, pembangkit tenaga listrik, perikanan, pariwisata dll
Kalimat ke-2	mengingat das waduk wonogiri tataguna lahannya sudah banyak berubah, alih fungsi lahan dari hutan menjadi lahan pertanian akibat

	perkembangan kebutuhan manusia, maka terjadi laju erosi yang tinggi, dan akhirnya mengendap di waduk
Kalimat ke-3	hal ini umur waduk akan menjadi berkurang
Kalimat ke-4	salah satu pemasok sedimen terbesar adalah dari sub das keduang, selain luas das nya yang paling besar juga kondisinya sudah kritis
Kalimat ke-5	tindakan konservasi sangat di perlukan dalam pengendalian laju erosi
Kalimat ke-6	upaya reboisasi yang diterapkan harus secara efisien dan tepat guna
Kalimat ke-7	tujuan dari penelitian ini adalah untuk menetukan alternatif teknik reboisasi sehingga bisa menurunkan laju erosi dan aliran permukaan
Kalimat ke-8	untuk mengetahui laju erosi permukaan dianalisis dengan metode usle dan dikalibrasi dengan pengukuran langsung, sedangkan aliran permukaan dianalisis berdasarkan kondisi aktual dan setelah direboisasi dengan rumusan rational yang dikalibrasi pengukuran debit aliran secara langsung
Kalimat ke-9	berdasarkan hasil penelitian dapat diketahui bahwa reobisasi lahan tegal dan hutan dengan tanaman reboisasi terpilih merupakan skenario yang paling efisien dalam penangulangan erosi

3. Memisahkan tiap kata pada tiap kalimat abstrak

Setelah memisahkan kalimat yang ada pada abstrak, kemudian dilakukan pemisahan setiap kata pada setiap kalimat dan dimasukkan ke dalam array serta dihitung jumlah keseluruhan kata. Aturan pemisahan kata berdasarkan spasi. Proses perhitungan jumlah kata pada abstrak dapat dilihat pada gambar *flowchart* dibawah ini.



Gambar III.4 *Flowchart* perhitungan jumlah kata pada abstrak

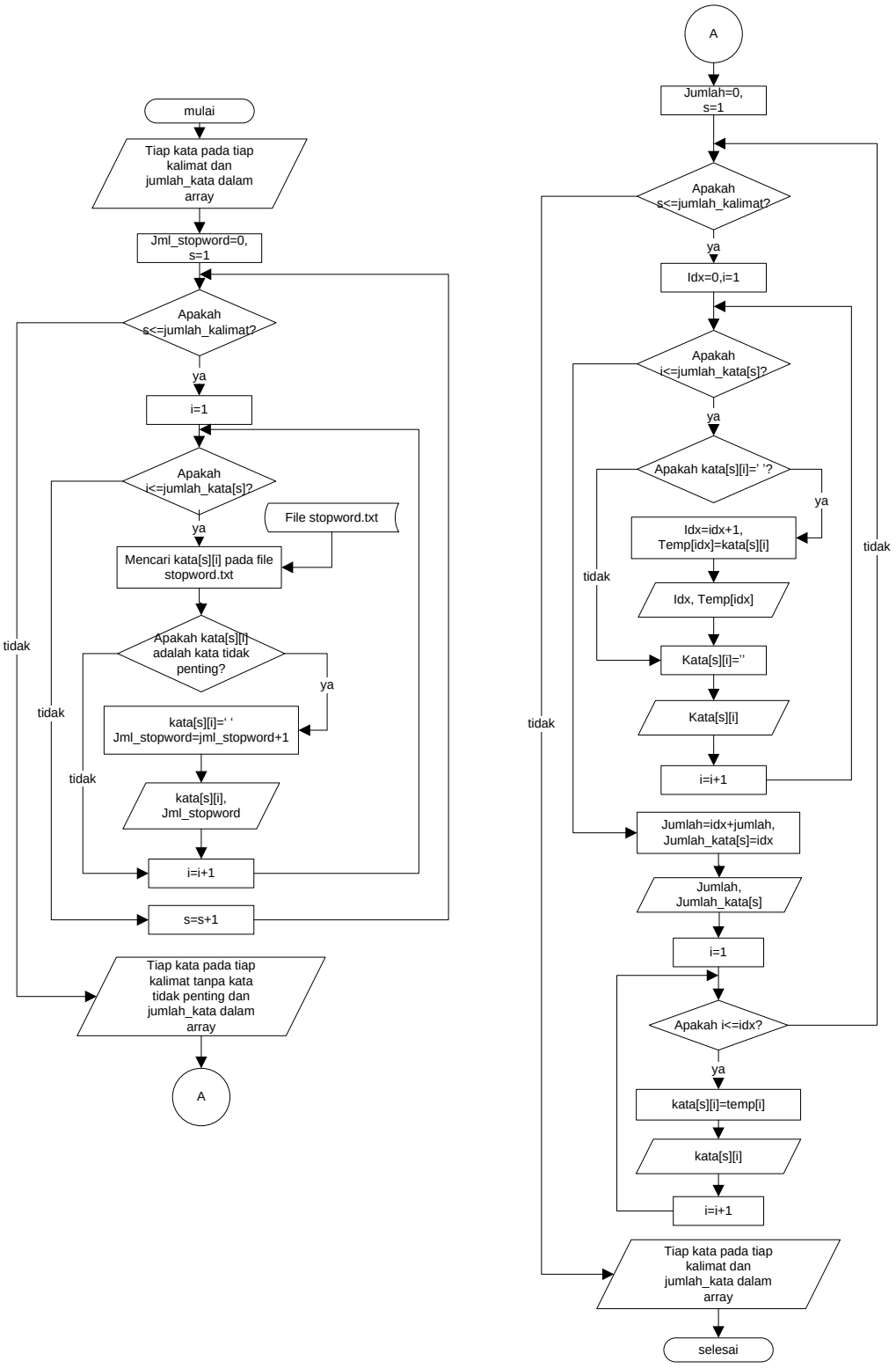
Setiap kata pada setiap kalimat dimasukkan ke dalam array. Jumlah kata pada setiap kaimat berbeda-beda dan setiap jumlahnya disimpan di dalam array jumlah_kata. Kata-kata yang telah dipisahkan disetiap kalimatnya dapat dilihat pada Tabel III.9.

Tabel III.9 Pemisahan tiap kata setiap kalimat

	Kalimat ke-1	Kalimat ke-2	Kalimat ke…	Kalimat ke-9
Kata ke-1	waduk	mengingat	…	berdasarkan
Kata ke-2	wonogiri	das	…	hasil
Kata ke-3	adalah	waduk	…	penelitian
Kata ke-4	salah	wonogiri	…	dapat
Kata ke-5	satu	tataguna	…	diketahui
Kata ke-6	pengendali	lahannya	…	bahwa
Kata ke-7	banjir	sudah	…	reboisasi
Kata ke-..	…	…	…	…
Kata diakhir kalimat	dll	waduk	…	erosi

4. Menghilangkan kata tidak penting atau *stopword*

Setelah pemisahan tiap kata pada setiap kalimat, kemudian penghilangan kata tidak penting *(stopword)* pada setiap kata yang ada pada setiap. Kata tidak penting atau *stopword* disimpan dalam file berextensi txt. Setiap kata diperiksa apakah merupakan kata tidak penting atau bukan. Kata yang merupakan kata tidak penting akan digantikan terlebih dahulu dengan spasi. Setelah itu baru kata disusun kembali dan yang berisikan spasi akan dihilangkan. Proses menghilangkan kata tidak penting atau *stopword* dapat dilihat pada Gambar III.5.



Gambar III.5 Penghilangan kata tidak penting atau *stopword*

Penggantian kata tidak penting menjadi spasi dapat dilihat pada tabel III.10.

Tabel III.10 tiap kata pada setiap kalimat penggantian kata tidak penting atau *stopword* menjadi spasi

	Kalimat ke-1	Kalimat ke-2	Kalimat ke…	Kalimat ke-8
Kata ke-1	waduk	mengingat	…	‘ ‘
Kata ke-2	wonogiri	das	…	‘ ‘
Kata ke-3	‘ ‘	waduk	…	penelitian
Kata ke-4	‘ ‘	wonogiri	…	‘ ‘
Kata ke-5	‘ ‘	tataguna	…	diketahui
Kata ke-6	pengendali	lahannya	…	‘ ‘
Kata ke-7	banjir	‘ ‘	…	reboisasi
Kata ke-..	…	…	…	…
Kata diakhir kalimat	dll	waduk	…	erosi

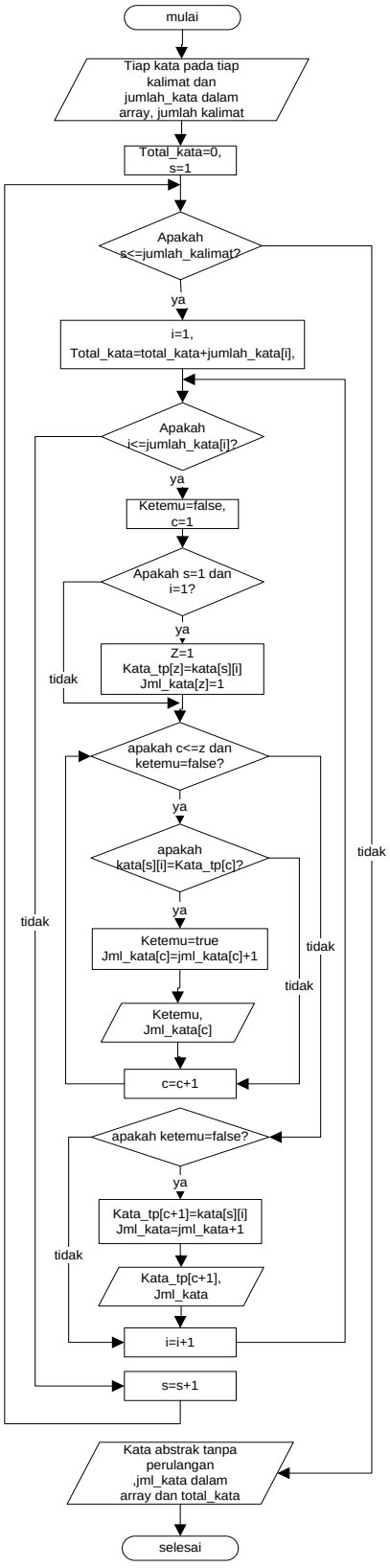
Setelah itu baru kata disusun kembali dan yang berisikan spasi akan dihilangkan dapat dilihat pada tabel III.11.

Tabel III.11 tiap kata pada setiap kalimat setelah penghilangan kata tidak penting atau *stopword*

	Kalimat ke-1	Kalimat ke-2	Kalimat ke…	Kalimat ke-8
Kata ke-1	waduk	mengingat	…	penelitian
Kata ke-2	wonogiri	das	…	diketahui
Kata ke-3	pengendali	waduk	…	reboisasi
Kata ke-4	banjir	wonogiri	…	lahan
Kata ke-5	bengawan	tataguna	…	tegal
Kata ke-6	solo	lahannya	…	hutan
Kata ke-7	berfungsi	berubah	…	tanaman
Kata ke-..	…	…	…	…
Kata diakhir kalimat	dll	waduk	…	erosi

5. Memisahkan dan menghitung kata abstrak tanpa pengulangan kata

Proses selanjutnya adalah memisahkan dan menghitung kata tanpa pengulangan kata yang sama. Kata-kata tersebut dimasukkan ke dalam array dan dihitung jumlahnya. Nilai inilah yang dijadikan nilai N. *Flowchart* memisahkan dan menghitung kata abstrak tanpa pengulangan kata dapat dilihat pada Gambar III.6. kata tanpa perulangan kata pada *flowchart* dimasukkan ke dalam array kata_tp dan jumlah setiap katanya jml_kata.



Gambar III.6 Memisahkan dan menghitung kata abstrak tanpa pengulangan kata

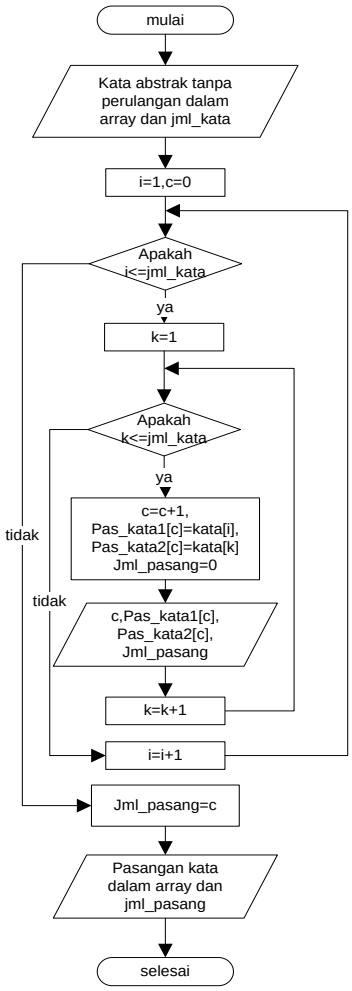
Jumlah kata tanpa dihitung kembali kata yang sama atau kata tanpa pengulangan kata sebanyak 76 kata dan total kata abstrak sebanyak 107 kata. Contoh kata tanpa pengulangan kata dari sampel abstrak yang dijadikan contoh kasus dapat dilihat pada Tabel III.12.

Tabel III.12 Kata tanpa pengulangan kata

Kata ke-	Kata abstrak	Jumlah kata
1	waduk	5
2	wonogiri	2
3	pengendali	2
4	banjir	2
5	bengawan	1
6	solo	1
7	multi	1
…	…	…
N=76	erosi	1

6. Menentukan Kemungkinan Pasangan Kata

Langkah selanjutnya adalah menentukan kemungkinan pasang kata apa saja yang muncul dalam abstrak. Menentukan pasang kata apa saja yang kemungkinan muncul pada abstrak dengan memasangkan kata tanpa pengulangan kata dengan kata tersebut dan dimasukkan ke dalam array ,pasangan kata ke-1 adalah kata ke-1 pada kata tanpa pengulangan kata dipasangkan dengan kata ke-2 pada kata tanpa pengulangan kata, pasangan kata ke-2 adalah kata ke-2 pada kata tanpa pengulangan kata dengan kata ke-3 pada kata tanpa pengulangan kata, begitu seterusnya hingga kata terakhir pada kata tanpa pengulangan serta jumlah tiap pasangan disimpan pada variabel jumlah pada tabel III.13 sedangkan pada *flowchart* variabelnya jml_pasang. Pemberian nilai inisialisasi awal untuk jumlah setiap pasang kata adalah 0 (nol). Proses menentukan kemungkinan pasangan kata dapat dilihat pada gambar *flowchart* III.7.



Gambar III.7 Flowchar menentukan pasangan kata

Berikut adalah tabel yang berisikan pasangan kata dengan nilai awal jumlah setiap pasang adalah 0 (nol).

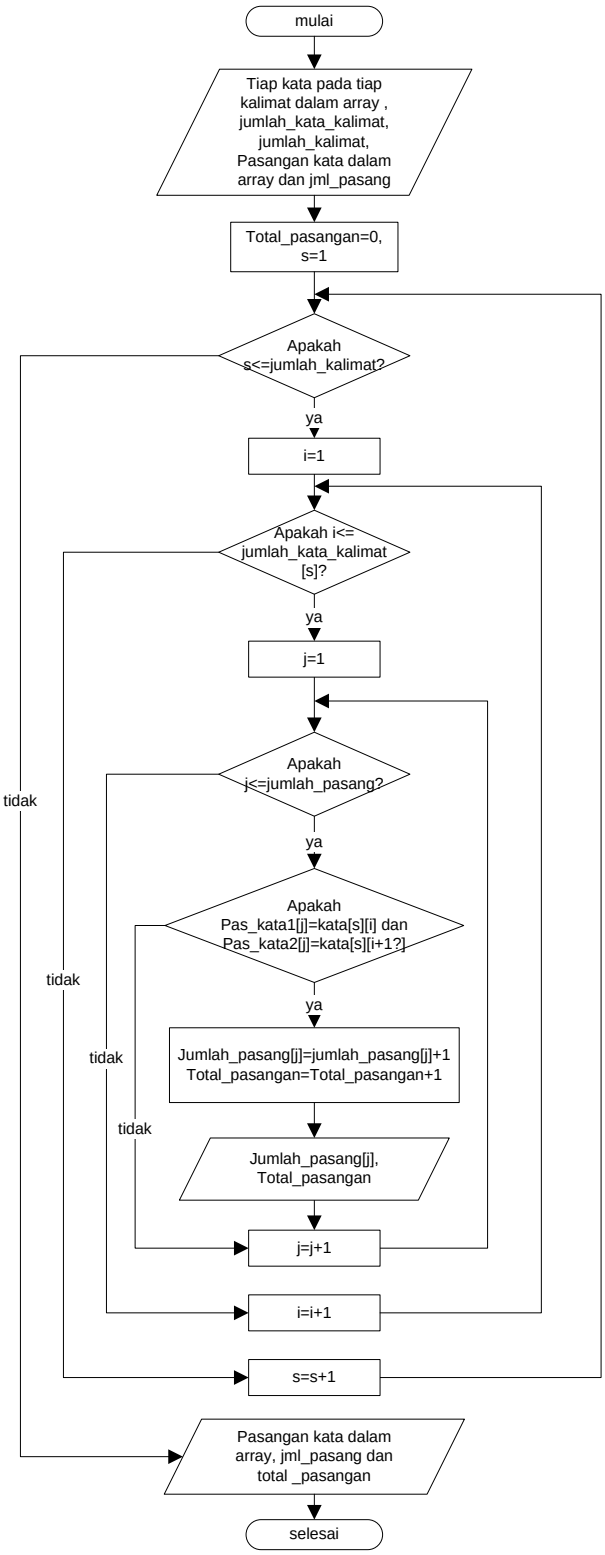
Tabel III.13 Kemungkinan pasang kata yang muncul pada abstrak

Pasangan ke-	Kata 1	Kata 2	Jumlah
1	waduk	wonogiri	0
2	waduk	pengendali	0
3	waduk	banjir	0
4	waduk	bengawan	0
5	waduk	solo	0
6	waduk	multi	0
7	waduk	guna	0
…	…	…	0
Pasangan terakhir	erosi	erosi	0

Setelah pasangan kata ditentukan, kemudian dihitung berapa banyak pasang kata yang muncul diabstrak.

7. Menghitung Banyaknya Tiap Pasang Kata yang Muncul

Penghitungan pasang kata dihitung dari setiap kalimat. Kata pertama dipasangkan dengan kata kedua. Kata kedua dipasangkan dengan kata ketiga dan seterusnya hingga diakhir kalimat. Kata pada akhir kalimat dan kata pada awal kalimat selanjutnya tidak akan dihitung menjadi satu pasang kata. Apabila sudah diakhir kalimat maka pasang kata yang dihitung dari kata pada awal kalimat selanjutnya dengan kata yang keduanya, begitu seterusnya hingga pada kalimat terakhir. Jumlah setiap pasang kata ini akan digunakan pada Algoritma *Baum Welch* dalam penginisialisasian nilai probabilitas transisi. *Flowchart* menghitung banyaknya tiap pasang kata yang muncul dapat dilihat pada gambar III.8.



Gambar III.8 *Flowchart* menghitung banyak tiap pasang kata

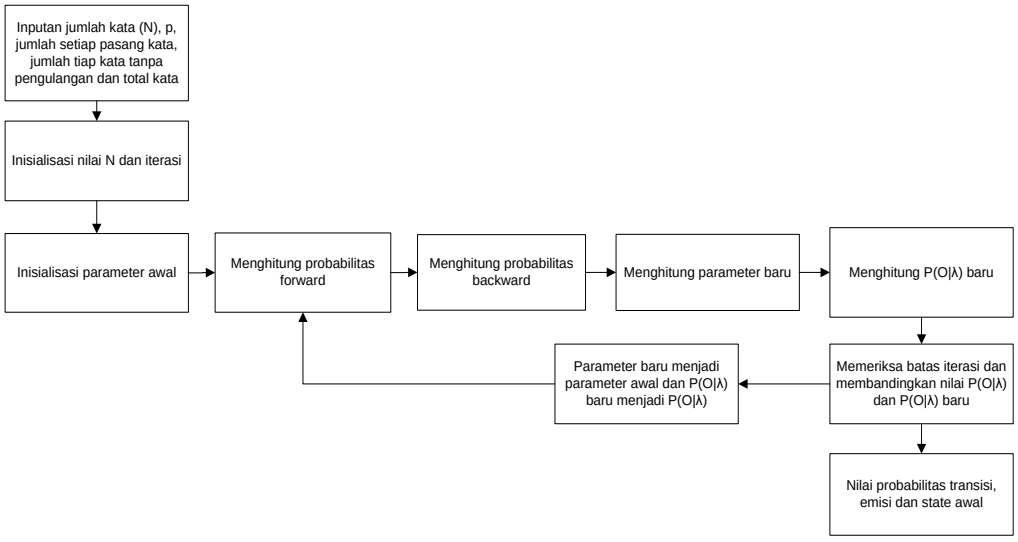
Pada tabel III.14 dapat dilihat jumlah setiap pasang kata yang terdapat pada abstrak.

Tabel III.14 Jumlah tiap pasang kata yang muncul pada abstrak

Pasangan ke-	Kata 1	Kata 2	Jumlah
1	waduk	waduk	0
2	waduk	wonogiri	2
3	waduk	pengendali	0
4	waduk	banjir	0
5	waduk	bengawan	0
6	waduk	solo	0
7	waduk	multi	0
…	…	…	…
Total pasang kata	erosi	erosi	0

3.1.1.2 Probabilistik Menggunakan Algoritma *Baum Welch*

Penentuan suatu kata kunci menggunakan nilai probabilitas kemunculan kata pada abstrak. Algoritma *Baum Welch* memaksimalkan probabilitas rangkaian pengamatan dengan mengestimasi nilai parameter (probabilitas transisi, emisi dan *state* awal) yang ada pada *Hidden Markov Model* atau HMM. Seluruh kata yang ada pada abstrak merupakan hidden *state*, apakah kata itu akan dijadikan kata kunci atau tidak. Proses Algoritma *Baum Welch* dapat dilihat pada Gambar III.9 dalam bentuk b*lock diagram*.



Gambar III.9 *Block diagram* Algoritma *Baum Welch*

Inputan yang digunakan pada Algoritma *Baum Welch* dalam menentukan kata kunci suatu dokumen adalah nilai N atau jumlah kata abstrak tanpa pengulangan kata, konstanta p = 0.1 (digunakan pada inisialisasi probabilitas transisi), jumlah setiap pasang kata (untuk perhitungan probabilitas pasang kata), jumlah kata abstrak tanpa pengulangan (untuk perhitungan probabilitas tiap kata) dan total kata (jumlah keseluruhan kata abstrak). Nilai - nilai tersebut telah dihitung pada proses *prepocessing*.

Nilai yang dihasilkan Algoritma *Baum Welch* berupa kumpulan probabilitas transisi, probabilitas emisi dan *state* awal dalam bentuk array. Nilai parameter ini akan dipakai dalam menentukan kata kunci abstrak. Berikut adalah tahapan pada Algoritma *Baum Welch.*

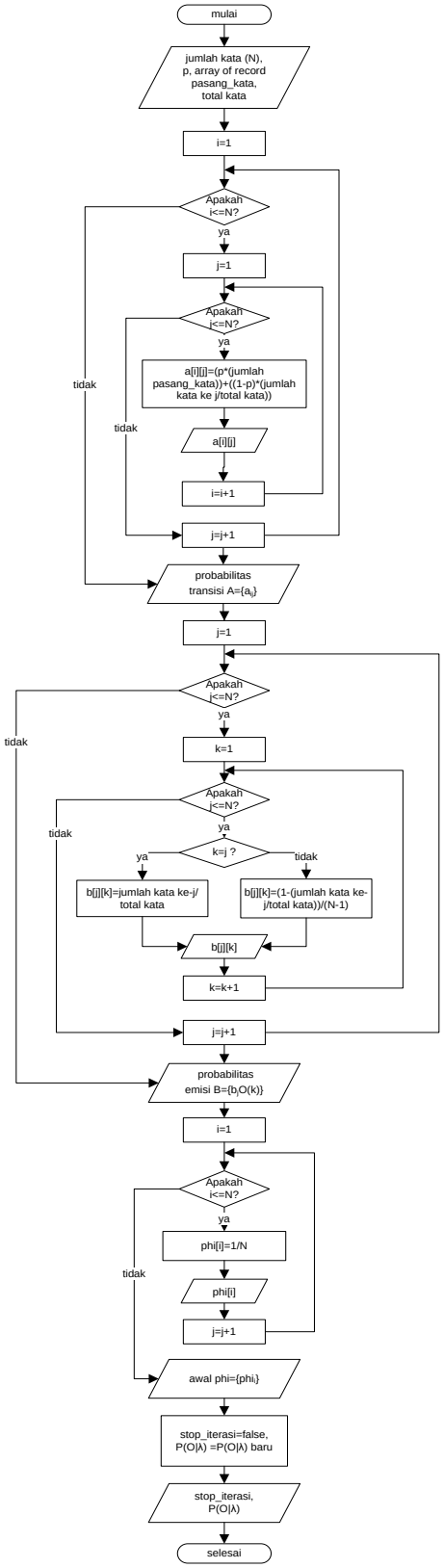
1. Pemberian nilai awal N dan iterasi

Nilai N diisikan dengan jumlah kata tanpa pengulangan kata yaitu 76. Iterasi diisikan dengan nilai 0 karena belum mengalami perhitungan ulang untuk menghitung nilai parameter yang baru.

N = jumlah kata = 76 ; iterasi = 0

2. Pengisian nilai parameter awal

Flowchart prosedur pengisian nilai parameter awal dapat dilihat pada Gambar III.10. Pada prosedur ini nilai parameter dari algoritma *Baum Welch* yaitu probabilitas transisi, probabilitas emisi dan *state* awal diberikan nilai awal serta inisialisasi stop_iterasi bernilai false dan probabilitas pengamatan atau P(O|λ) diisikan nantinya dengan P(O|λ) baru sehingga diawal P(O|λ) belum mempunyai nilai. Hasil dari prosedur ini akan digunakan dalam menghitung probabilitas *forward* dan *backward*.



Gambar III.10 *Flowchart* prosedur pengisian nilai parameter awal

– Menghitung Probabilitas transisi ($A=\{a_{ij}\}$):

Probabilitas transisi (a_{ij}) merupakan probabilitas suatu kata dipengaruhi oleh kata sebelumnya atau dapat dikatakan probabilitas pasangan kata. Jumlah i dan j sebanyak N atau jumlah kata tanpa pengulangan kata pada abstrak yaitu 76. i merupakan *state* i dimana kata pada *state* i dan j merupakan *state* j dimana kata pada *state* j. Contoh pada tabel *state* ke-i=1 adalah kata waduk dan *state* ke-j=1 adalah kata waduk. Kata i dan kata j didapat dari kata pada abstrak tanpa pengulangan kata. Kata yang pertama muncul pada abstrak akan menjadi urutan pertama dan seterusnya.

Tabel III.15 Inisialisasi probabilitas transisi

a_{ij}	**j=1** **(waduk)**	**j=2** **(wonogiri)**	**j=3** **(pengendali)**	**j=4** **(banjir)**	**j=5** **(bengawan)**	**j=6** **(solo)**	**j=7** **(berfungsi)**	**j=…** **(…)**	**j=N=76** **(erosi)**
i=1 **(waduk)**	a_{11} = 0,042056074766 waduk waduk	a_{12} = 0,11682242991 waduk wonogiri	a_{13} = 0, 116822429907 waduk pengendali	a_{14} =0, 116822429907 waduk banjir	a_{15} =0, 008417649533 waduk bengawan	a_{16} = 0, 008417649533 waduk solo	a_{17} = 0, 008417649533 waduk berfungsi	a_{1j} waduk kata ke-j	a_{1N} =0,008417649533 waduk solo
i=2 **(wonogiri)**	a_{21} = 0,042056074766 wonogiri waduk	a_{22} 0,116822429907 wonogiri wonogiri	a_{23} =0, 116822429907 wonogiri pengendali	a_{24} =0, 116822429907 wonogiri banjir	a_{25} =0, 008417649533 wonogiri bengawan	a_{26} = 0, 008417649533 wonogiri solo	a_{27} = 0, 008417649533 wonogiri berfungsi	a_{2j} wonogiri kata ke-j	a_{2N} =0,008417649533 wonogiri erosi
i=3 **(pengendali)**	a_{31} = 0,042056074766 pengendali waduk	a_{32} = 0, 116822429907 pengendali wonogiri	a_{33} =0, 116822429907 pengendali pengendali	a_{34} =0,1168224991 pengendali banjir	a_{35} =0, 008417649533 pengendali bengawan	a_{36} = 0, 008417649533 pengendali solo	a_{37} = 0, 008417649533 pengendali berfungsi	a_{3j} pengendali kata ke-j	a_{3N} =0,008417649533 pengendali erosi
i=4 **(banjir)**	a_{41} = 0,042056074766 banjir waduk	a_{42} = 0, 116822429907 banjir wonogiri	a_{43} =0, 116822429907 banjir pengendali	a_{44} =0, 116822429907 banjir banjir	a_{45} =0,10841131495 banjir bengawan	a_{46} = 0, 008417649533 banjir solo	a_{47} = 0, 008417649533 banjir berfungsi	a_{4j} banjir kata j	a_{4N} =0,008417649533 banjir erosi
i=5 **(bengawan)**	a_{51} = 0,042056074766 bengawan waduk	a_{52} = 0, 116822429907 bengawan wonogiri	a_{53} =0,116822429907 bengawan pengendali	a_{54} =0, 116822429907 bengawan banjir	a_{55} = 0,008417649533 bengawan bengawan	a_{56} =0, 10841131495 bengawan solo	a_{57} = 0, 008417649533 bengawan berfungsi	a_{5j} bengawan kata j	a_{5N} =0,008417649533 bengawan erosi
i=6 **(solo)**	a_{61} 0,042056074766solo waduk	a_{62} = 0, 116822429907 solo wonogiri	a_{63} =0,116822429907 solo pengendali	a_{64} =0, 116822429907 solo banjir	a_{65} =0, 008417649533 solo bengawan	a_{66} = 0, 008417649533 solo solo	a_{67} =0, 10841131495 solo berfungsi	a_{6j} solo kata j	a_{6N} =0,008417649533 solo erosi

a_{ij}	**j=1** **(waduk)**	**j=2** **(wonogiri)**	**j=3** **(pengendali)**	**j=4** **(banjir)**	**j=5** **(bengawan))**	**j=6** **(solo)**	**j=7** **(berfungsi)**	**j=…** **(…)**	**j=N=76** **(erosi)**
i=7 **(berfungsi)**	a_{71} = 0,042056074766 berfungsi waduk	a_{72} = 0, 116822429907 berfungsi wonogiri	a_{73} =0, 116822429907 berfungsi pengendali	a_{74} =0, 116822429907 berfungsi banjir	a_{75} =0,008417649533 berfungsi bengawan	a_{76} = 0, 008417649533 berfungsi solo	a_{77} = 0, 008417649533 berfungsi berfungsi	a_{7j} sumber kata j	a_{7N} =0,008417649533 berfungsi erosi
i=… **(...)**	a_{i1} = .. kata ke-i waduk	a_{i2} katake-i wonogiri	a_{I3} kata ke-i pengendali	a_{i4} kata ke-i banjir	a_{i5} kata ke-i bengawan	a_{i6} kata ke-i solo	a_{i7} kata ke-i berfungsi	a_{ij} kata ke-i kata ke-j	a_{iN} kata ke-i erosi
i=N=76 **(erosi)**	a_{N1} = 0,042056074766 erosi waduk	a_{N2} = 0, 116822429907 erosi wonogiri	a_{N3} =0, 116822429907 erosi pengendali	a_{N4} =0, 116822429907 erosi banjir	a_{N5} =0, 008417649533 erosi bengawan	a_{N6} = 0, 008417649533 erosi solo	a_{N7} = 0, 008417649533 erosi berfungsi	a_{iN} kata ke-i erosi	a_{NN} =0, 08417649533 erosi erosI

Setiap kolom pada Tabel III.15 diisikan dengan persamaan sebagai berikut berdasarkan persamaan II.14:

$$P(pasangan\ kata) = \left(\mathrm{p} * \acute{\mathrm{P}}(pasangan\ kata)\right) + \left((\mathrm{p}-1) * \acute{\mathrm{P}}(kata\ i)\right)$$

Keterangan:

$$\acute{\mathrm{P}}(pasangan\ kata) = \frac{jumlah(pasang\ kata\ i\ kata\ j)}{jumlah(kata\ i)}$$

$$\acute{\mathrm{P}}(kata\ i) = \frac{jumlah(kata\ i)}{total\ kata}$$

$\mathrm{p} = 0.1$, konstanta antara 0 sampai 1

Perhitung nilai a_{ij} sebagai berikut:

a_{11} = $\mathrm{P}(pasangan\ kata)$

= $\left(\mathrm{p} * \acute{\mathrm{P}}(pasangan\ kata)\right) + \left((\mathrm{p}-1) * \acute{\mathrm{P}}(kata\ i)\right)$

= (0,1* Ṕ (waduk waduk)+(0,9* $\acute{\mathrm{P}}(waduk)$)

$=\left(0{,}1 * \left(\frac{jumlah\ pasang\ kata-waduk\ waduk}{jumlah\ kata\ waduk}\right)\right) + \left(0{,}9 * \left(\frac{jumlah\ kata\ waduk}{total\ kata}\right)\right)$

$= \left(0{,}1 * \left(\frac{0}{5}\right)\right) + \left(0{,}9 * \left(\frac{5}{107}\right)\right)$

= 0, 042056074766

a_{12} = $\mathrm{P}(pasangan\ kata)$

= $\left(\mathrm{p} * \acute{\mathrm{P}}(pasangan\ kata)\right) + \left((\mathrm{p}-1) * \acute{\mathrm{P}}(kata\ i)\right)$

= (0,1* Ṕ (waduk wonogiri)+(0,9* $\acute{\mathrm{P}}(wonogiri)$)

=

$\left(0{,}1 * \left(\frac{jumlah\ pasang\ kata-waduk\ wonogiri}{jumlah\ kata\ waduk}\right)\right) + \left(0{,}9 * \left(\frac{jumlah\ kata\ wonogiri}{total\ kata}\right)\right)$

$= \left(0{,}1 * \left(\frac{0}{5}\right)\right) + \left(0{,}9 * \left(\frac{2}{107}\right)\right)$

= 0, 11682242991

a_{13} = (0,1* Ṕ (waduk pengendali)+(0,9* $\acute{\mathrm{P}}(pengendali)$)

=

$\left(0{,}1 * \left(\frac{jumlah\ pasang\ kata-waduk\ pengendali}{jumlah\ kata\ waduk}\right)\right) + \left(0{,}9 * \left(\frac{jumlah\ kata\ pengendali}{total\ kata}\right)\right)$

$= \left(0{,}1 * \left(\frac{0}{5}\right)\right) + \left(0{,}9 * \left(\frac{2}{107}\right)\right)$

$= 0,116822429907$

Dan seterusnya hingga a_{NN} atau $a_{76\,76}$

$a_{76\,76}$ =(0,1* Ṕ (erosi erosi)+(0,9* $(erosi)$)

$= \left(0{,}1 * \left(\frac{jumlah\ pasang\ kata - erosi\ erosi}{jumlah\ kata\ erosi}\right)\right) + \left(0{,}9 * \left(\frac{jumlah\ kata\ erosi}{total\ kata}\right)\right)$

$= \left(0{,}1 * \left(\frac{1}{5}\right)\right) + \left(0{,}9 * \left(\frac{1}{107}\right)\right)$

$= 0,08417649533$

– Menghitung probabilitas emisi (B={ $b_j O(k)$}):

Probabilitas emisi ($b_j O(k)$ merupakan probabilitas pengamatan atau observasi pada *state* j. Jumlah j dan k sebanyak N atau jumlah kata tanpa pengulangan kata pada abstrak yaitu 76. Kata pada abstrak tanpa pengulangan kata yang dijadikan objek pengamatan atau observasi atau O(k). Contoh pada tabel *state* ke-i=1 adalah kata waduk dan *state* ke-j=1 adalah kata waduk. Kata i dan kata j didapat dari kata pada abstrak tanpa pengulangan kata. Kata yang pertama muncul pada abstrak akan menjadi urutan pertama dan seterusnya.

Tabel III.16 Inisialisasi probabilitas emisi

$b_jO(k)$	**k=1** **O(1)=waduk**	**k=2** **O(2)=wonogiri**	**k=3** **O(3)=pengenda**	**k=4** **O(4)=banjir**	**k=5** **O(5)=bengawan**	**k=6** **O(6)=solo**	**k=7** **O(7)=berfungsi**	**k=…** **(…)**	**k=N** **O(N)=erosi**
j=1 **(waduk)**	$b_1O(1)$ =0,046728971963 waduk waduk	$b_1O(2)$ =0,012710280374 waduk wonogiri	$b_1O(3)$ =0, 012710280374 waduk pengendali	$b_1O(4)$ =0, 012710280374 waduk banjir	$b_1O(5)$ =0, 012710280374 waduk bengawan	$b_1O(6)$ =0, 012710280374 waduk solo	$b_1O(7)$ =0, 012710280374 waduk berfungsi	$b_1O(k)$ = … waduk kata ke-j	$b_1O(N)$ =0, 012710280374 waduk erosi
j=2 **(wonogiri)**	$b_2O(1)$ =0,0130841765 wonogiri waduk	$b_2O(2)$ = 0,018691588785 wonogiri wonogiri	$b_2O(3)$ =0,0130841765 wonogiri pengendali	$b_2O(4)$ =0,0130841765 wonogiri banjir	$b_2O(5)$ =0, 0130841765 wonogiri bengawan	$b_2O(6)$ =0, 0130841765 wonogiri solo	$b_2O(7)$ =0, 0130841765 wonogiri berfungsi	$b_2O(k)$ = … merupakan kata ke-j	$b_2O(N)$ =0, 0130841765 wonogiri erosi
j=3 **(pengendali)**	$b_3O(1)$ =0, 0130841765 pengendali waduk	$b_3O(2)$ =0, 0130841765 pengendali wonogiri	$b_3O(3)$ = 0, 018691588785 pengendali pengendali	$b_3O(4)$ =0, 0130841765 pengendali banjir	$b_3O(5)$ =0, 0130841765 pengendali bengawan	$b_3O(6)$ =0, 0130841765 pengendali solo	$b_3O(7)$ =0, 0130841765 pengendali berfungsi	$b_3O(k)$ = … pengendali kata ke-j	$b_3O(N)$ =0, 0130841765 pengendali erosi
j=4 **(banjir)**	$b_4O(1)$ =0, 0130841765 banjir waduk	$b_4O(2)$ =0, 0130841765 banjir wonogiri	$b_4O(3)$ =0, 0130841765 banjir pengendali	$b_4O(4)$ = 0, 018691588785 banjir banjir	$b_4O(5)$ =0, 0130841765 banjir bengawan	$b_4O(6)$ =0, 0130841765 banjir solo	$b_4O(7)$ =0, 0130841765 banjir berfungsi	$b_4O(k)$ = … banjir kata j	$b_4O(N)$ =0, 0130841765 banjir erosi
j=5 **(bengawan)**	$b_5O(1)$ = 0,013208722741 bengawan waduk	$b_5O(2)$ =0, 013208722741 bengawan wonogiri	$b_5O(3)$ =0, 013208722741 bengawan pengendali	$b_5O(4)$ =0, 013208722741 bengawan banjir	$b_5O(5)$ = 0,0093457943925 bengawan bengawan	$b_5O(6)$ =0, 013208722741 bengawan solo	$b_5O(7)$ =0, 013208722741 bengawan berfungsi	$b_5O(k)$ = … bengawan kata j	$b_5O(N)$ =0, 0130841765 bengawan erosi
j=6 **(solo)**	$b_6O(1)$ = 0, 013208722741 solo waduk	$b_6O(2)$ =0, 013208722741 solo wonogiri	$b_6O(3)$ =0, 013208722741 solo pengendali	$b_6O(4)$ =0, 013208722741 solo banjir	$b_6O(5)$ =0, 013208722741 solo bengawan	$b_6O(6)$ = 0, 0093457943925 solo solo	$b_6O(7)$ = 0, 013208722741 solo berfungsi	$b_6O(k)$ = … solo kata j	$b_6O(N)$ =0, 0130841765 solo erosi

$b_jO(k)$	**k=1** **O(1)=waduk**	**j=2** **(wonogiri)**	**k=3** **O(3)=pengendali**	**k=4** **O(4)=banjir**	**k=5** **O(5)=bengawan**	**k=6** **O(6)=solo**	**k=7** **O(7)=berfungsi**	**k=…** **(…)**	**k=N** **O(N)=erosi**
j=7 **(berfungsi)**	$b_7O(k)$ =0, 013208722741 berfungsi waduk	$b_7O(2)$ =0, 013208722741 berfungsi wonogiri	$b_7O(3)$ =0, 013208722741 berfungsi pengendali	$b_7O(4)$ =0, 013208722741 berfungsi banjir	$b_7O(5)$ =0, 013208722741 berfungsi bengawan	$b_7O(6)$ =0, 013208722741 berfungsi solo	$b_7O(k)$ =0, 0093457943925 berfungsi kata j	$b_7O(k)$ = … berfungsi kata j	$b_7O(N)$ =0, 0130841765 berfungsi erosi
j=… **(...)**	$b_jO(1)$ = … kata ke-i waduk	$b_jO(2)$ = … katake-i wonogiri	$b_jO(3)$ = … kata ke-i	$b_jO(4)$ = … kata ke-i banjir	$b_jO(5)$ = … kata ke-i bengawan	$b_jO(6)$ = … kata ke-i solo	$b_jO(7)$ = … kata ke-i berfungsi	$b_jO(k)$ = … kata ke-i kata ke-j	$b_jO(N)$ = … kata ke-i erosi
j=N **(erosi)**	$b_NO(1)$ =0, 013208722741 erosi waduk	$b_NO(2)$ =0, 013208722741 erosi wonogiri	$b_NO(3)$ =0, 013208722741 erosi pengendali	$b_NO(4)$ =0, 013208722741 erosi banjir	$b_NO(5)$ =0, 013208722741 erosi bengawan	$b_NO(6)$ =0, 013208722741 erosi solo	$b_NO(7)$ =0, 0093457943925 erosi berfungsi	$b_NO(k)$ = … erosi kata j	$b_NO(N)$ =0, 0130841765 erosi erosi

Perhitung nilai $b_j(O_k)$ sebagai berikut:

Setiap kolom diatas pada Tabel III.16, dimana j yang bernilai sama dengan k yang diisikan dengan persamaan sebagai berikut berdasarkan persamaan:

$$\text{P(kata observasi k| kata pada } \textit{state} \text{ j)} = \frac{jumlah(kata\ observasi\ k, kata\ pada\ state\ j)}{total\ kata}$$

Kolom yang lainnya dimana j tidak sama dengan k disetiap j yaitu pada k=1 sampai k=N kecuali nilai k sama dengan j diberikan nilai probabilitas = (1- P(kata observasi k| kata pada *state* j))/(N-1), agar nilai tidak bernilai 0 (nol).

Perhitung nilai $b_jO(k)$ sebagai berikut:

$b_1O(1)$ = P(kata observasi 1| kata pada *state* 1)

= P(waduk|waduk)

= 5/107

$= 0,046728971963$

$b_1O(2)$ = (1- P(kata observasi k| kata pada *state* j))/(N-1)

= $(1- 0,046728971963)/(76-1)$

= $0,012710280374$

Dan seterusnya hingga $b_NO(N)$ atau $b_{76}O(76)$

$b_{76}O(76)$= P(kata observasi k| kata pada *state* j)

= P(erosi|erosi)

= 1/107

$= 0,0130841765$

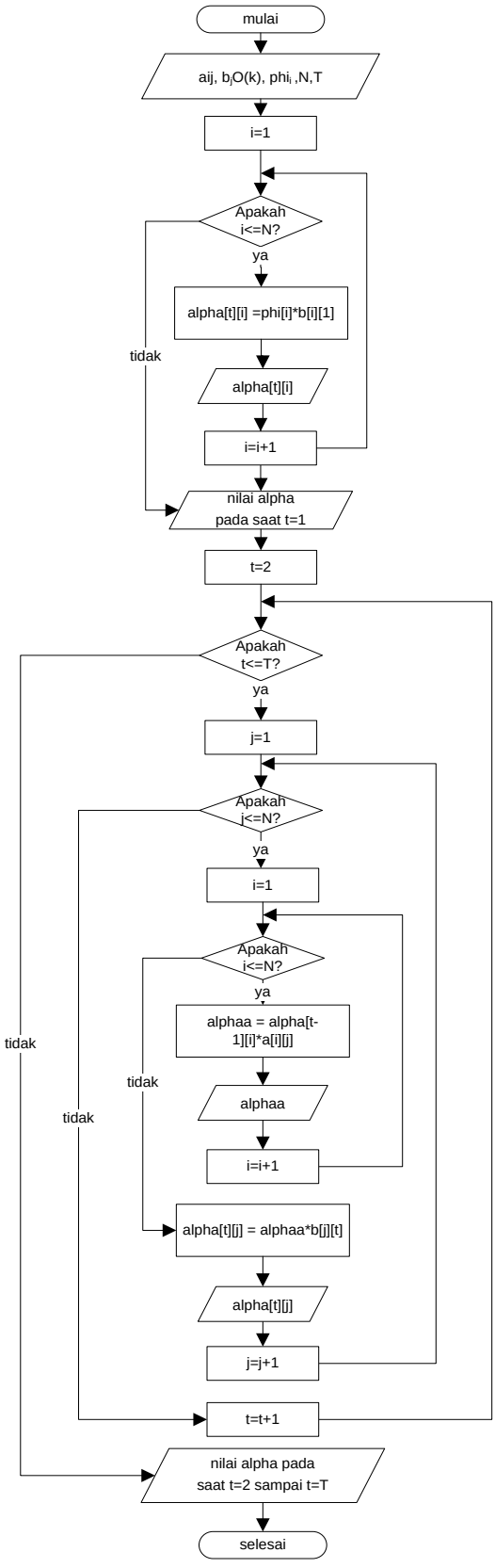
– menghitung nilai *state* awal (phi_i) :

Nilai *state* awal diberikan dengan nilai yang sama yaitu $phi_i = 1/N$, dengan nilai N=76. Phi_1 merupakan nilai *state* awal kata waduk, dimana kata – kata ini merupakan kata abstrak yang tanpa perulangan kata.

phi_i	**1/N**
i=1 (waduk)	1/76 = 0,013157894737
i=2 (wonogiri)	1/76 = 0, 013157894737
i=3 (pengendali)	1/76 = 0, 013157894737
i=4 (banjir)	1/76 = 0, 013157894737
i=5 (bengawan)	1/76 = 0, 013157894737
i=… (...)	…
i=N (erosi)	1/76 = 0, 013157894737

3. Menghitung Probabilitas Forward

Gambar III.11 adalah *flowchart* prosedur menghitung probabilitas forward. Pada prosedur probabilitas forward nilai yang dihasilkan berupa himpunan nilai alpha yang akan digunakan dalam perhitungan parameter baru. Probabilitas forward disebut juga probabilitas maju karena nilai alpha dihitung mulai dari alpha pada t=1, selanjutnya nilai ini yang akan mempengaruhi nilai alpha pada t selanjutnya hingga nilai t=T.



Gambar III.11 *Flowchart* prosedur probabilitas forward

Pada perhitungan probabilitas forward terdapat 2 langkah yaitu inisialisasi dan induksi. Setiap nilai probabilitas forward ($alpha_t(j)$ atau $\propto_t (j)$) akan saling berkaitan.

Inisialisasi

Nilai inisialisasi merupakan nilai probabilitas forward atau alpha pada saat t=1 pada setiap i (i=1 sampai i=N=76). perhitungan inisiliasisasi menggunakan persamaan II.2 :

$alpha_1(1)$= $phi_1.b_1(O_1)$ = 0, 013157894737. 0, 046728971963= 0,00061485489424

$alpha_1(2)$= $phi_2.b_2(O_1)$ = 0, 013157894737. 0, 0130841765= 0,00017215937039

…

$alpha_1(76)$= $phi_{76}.b_{76}(O_1)$ = 0, 013157894737. 0, 0130841765= 0,0001737989344

induksi

Perhitungan induksi menggunakan persamaan II.3. Berikut adalah perhitungan probabilitas forward pada saat t=2 untuk j=1:

$$alpha_t(j) = \left[\sum_{i=1}^{N} alpha_{t-1}(i)a_{ij}\right] b_j(O_t)$$

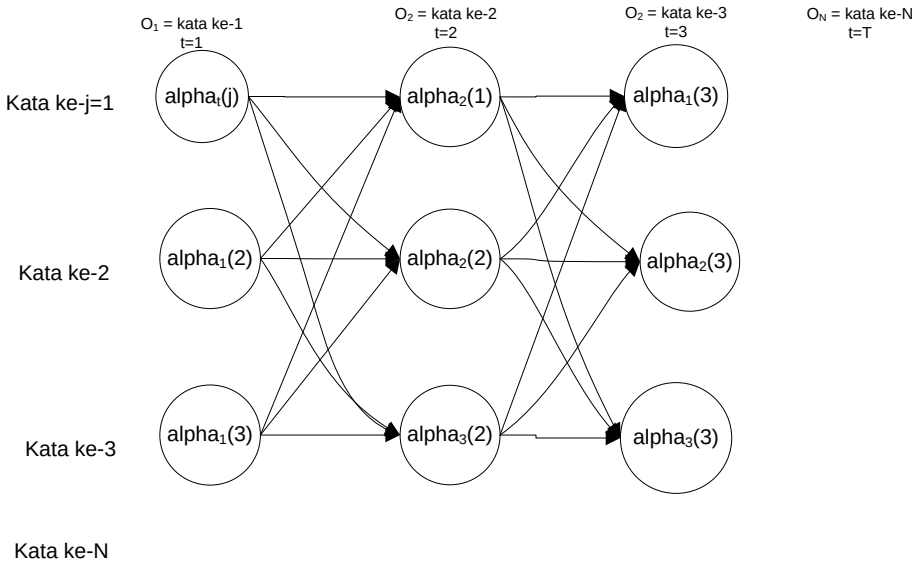
$$alpha_2(1) = \left[\sum_{i=1}^{76} alpha_{2-1}(i)a_{i1}\right] b_1(O_2)$$

$$= [alpha_1(1) \,.\, a_{11} + alpha_1(2).\, a_{21} + .. + alpha_1(76).\, a_{76\,1}].\, b_1(O_2)$$

$$= [0{,}00061485489424 \,.\, 0{,}042056074766 + 0{,}00017215937039 \,.\, 0{,}042056074766 + \cdots + 0{,}0001737989344 \,.\, 0{,}042056074766]0{,}046728971963$$

$$= 7{,}4482223904 \,.\, 10^{-6}$$

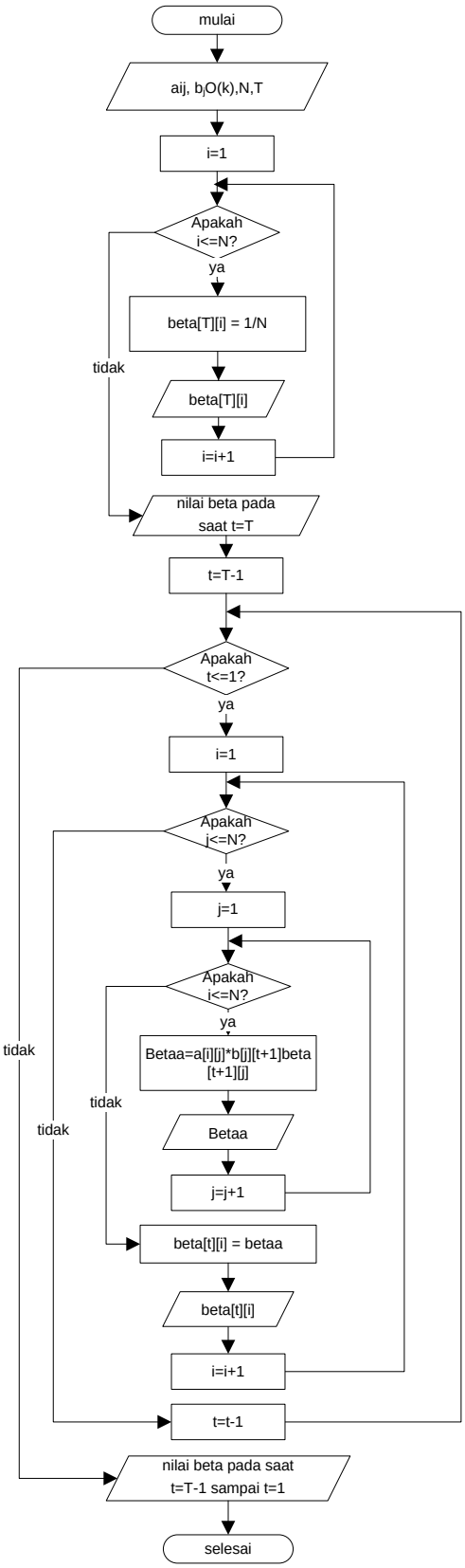
Perhitungan pada saat j=2, j=3, …, j=76 perhitungan dilakukan sama seperti perhitungan diatas hingga t=T. Keterkaitan nilai probabilitas forward dapat dilihat pada gambar trellis berikut.



Gambar III.12 Trellis probabilitas forward

4. Menghitung Probabilitas Backward

Gambar III.13 adalah *flowchart* prosedur perhiutngan probabilitas backward. Probabilitas backward atau disebut juga probabilitas mundur. Dinamai ini karena proses perhitungan dimulai dengan nilai beta pada saat t=T, selanjutnya perhitungan ini akan mempengaruhi nilai beta pada saat t=T-1 seterusnya hingga t=1.



Gambar III.13 *Flowchart* prosedur probabilitas backward

Pada perhitungan probabilitas backward terdapat 2 langkah yaitu inisialisasi dan induksi.

Inisialisasi

Nilai inisialisasi merupakan nilai probabilitas backward atau beta pada saat t=T pada setiap i. Perhitungan probabilitas forwad menggunakan persamaan II.4:

$beta_{76}(1)$ = 1/76 = 0, 013157894737

$beta_{76}(2)$ = 1/76 = 0, 013157894737

$beta_{76}(3)$ = 1/76 = 0, 013157894737

…

$beta_{76}(N)$= 1/76 = 0, 013157894737

induksi

Perhitungan nilai induksi mengunakan persamaan II.5. Berikut adalah perhitungan probabilitas backward pada saat t=T-1 untuk j=1:

$$\text{beta}_t(j) = \left[\sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \text{beta}_{t+1}(j)\right]$$

$$\text{beta}_{75}(1) = \left[\sum_{j=1}^{76} a_{ij} b_j(O_{76}) \text{beta}_{76}(j)\right]$$

$$= [\, a_{11}.b_1(O_{76}).beta_{76}(1) + a_{12}.b_2(O_{75}).beta_{75}(2) + \ldots + a_{1\,76}.b_{76}(O_1).beta_1(76)]$$

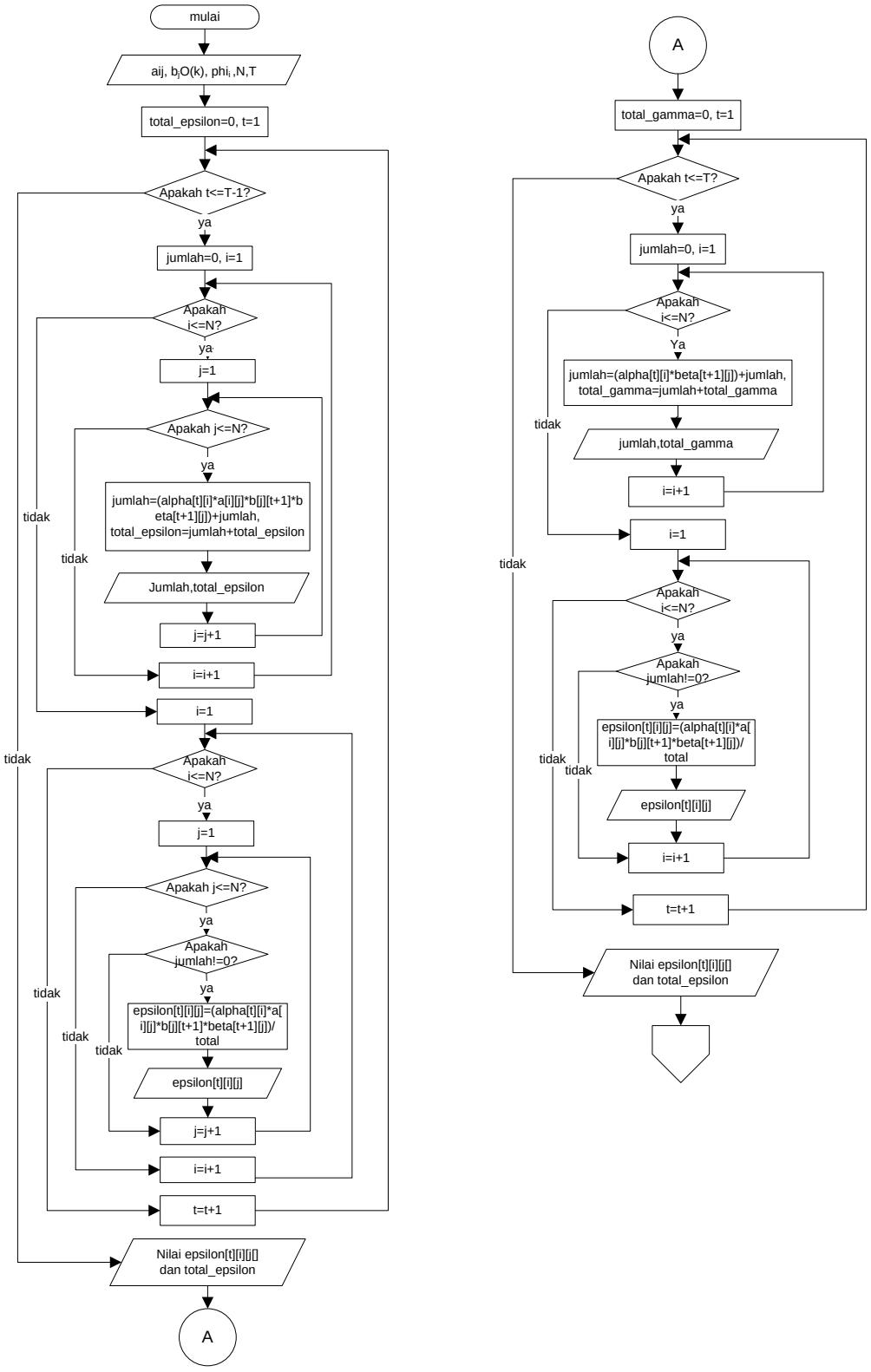
$$= [0{,}042056074766.0{,}012710280374.\ 0{,}013157894737 + 0{,}11682242991.0{,}01308411215.0{,}013157894737 + \cdots + 0{,}0084112149533.0{,}013208722741.0{,}013157894737]$$

$$= 0{,}00020656059749$$

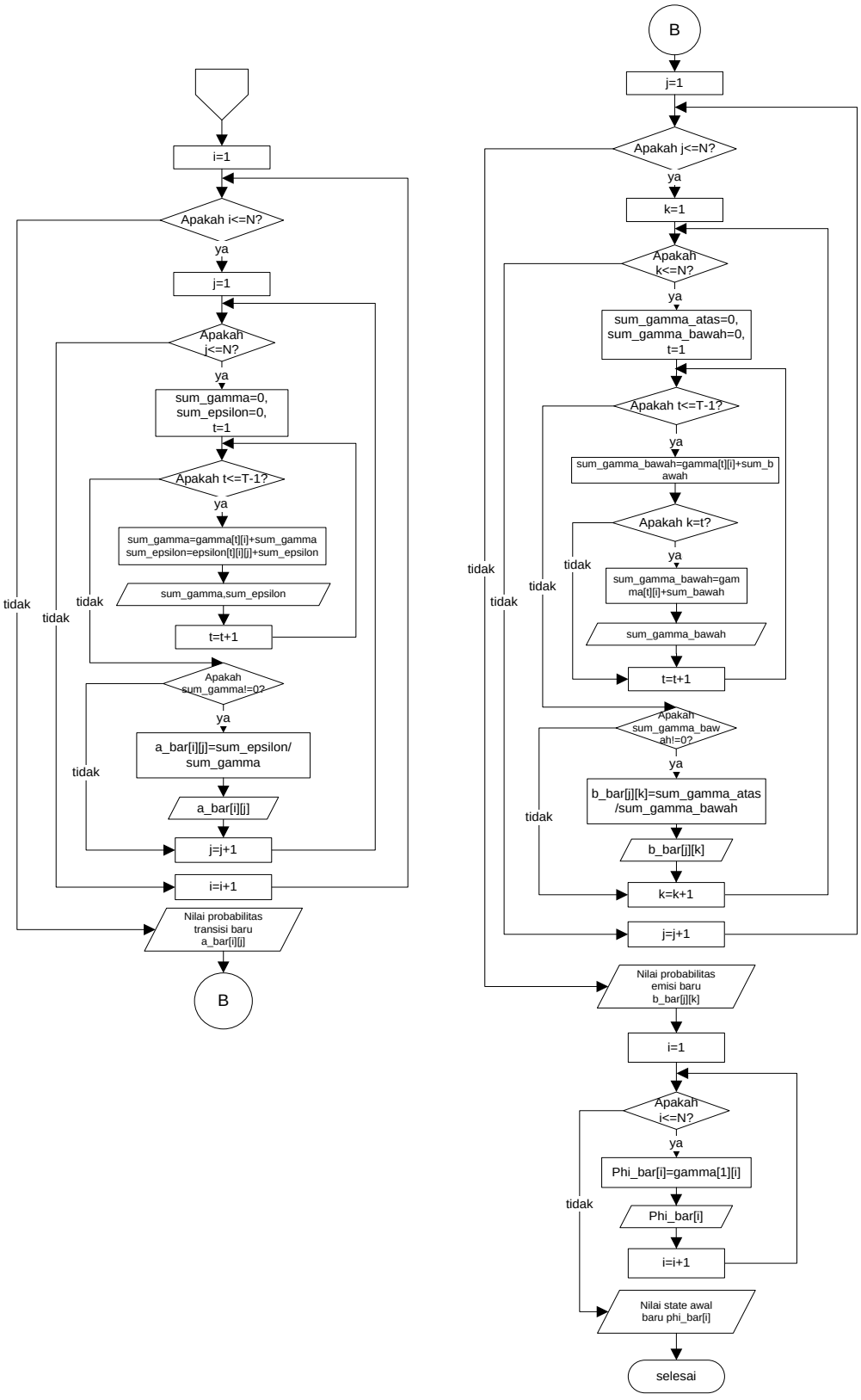
Perhitungan untuk j=2, j=3 sampai j=76 dilakukan sama seperti perhitungan diatas diatas hingga t=1.

5. Menghitung Parameter Baru

Gambar III.14 dan Gambar III.15 adalah *flowchart* prosedur menghitung parameter baru. Hasil perhitungan pada prosedur sebelumnya yaitu nilai a_{ij}, $b_j(o_k)$, phi $_i$, $alpha_t(i)$ dan $beta_t(i)$ akan digunakan dalam prosedur ini untuk menghitung parameter yang baru yaitu probabilitas transisi (A), probabilitas emisi (B) dan *state* awal (phi). Hasil dari prosedur inilah yang akan digunakan untuk menentukan kata kunci apabila parameter ini telah optimal.



Gambar III.14 *Flowchart1* prosedur probabilitas parameter baru



Gambar III.15 *Flowchar2* prosedur probabilitas parameter baru

Tahap pertama dalam menghitung parameter baru adalah menghitung nilai epsilon ($\varepsilon_{t_{(i,j)}}$) dan gamma ($\gamma_t(i)$) terlebih dahulu. Nilai alpha dan beta yang telah dihitung sebelumnya pada langkah 2 dan 3 akan digunakan dalam perhitungan nilai epsilon dan gamma. Setelah itu baru dapat menghitung parameter yang baru. Persamaan untuk menghitung nilai epsilon adalah persamaan II.10 dan persamaan yang digunakan untuk menghitung nilai gamma adalah persamaan II.9.

Perhitungan nilai gamma ($\gamma_t(i)$):

Perhitungan nilai gamma dilakukan dari t=1 hingga t=T dan i=1 hingga i=N.

Pada saat t=1, i=1 hingga i=N

$$gamma_1(1) = \frac{alpha_1(1).beta_1(1)}{\sum_{i=1}^{76} alpha_1(i).beta_1(i)}$$

$$= \frac{0{,}0006148548942 \,.\, 3{,}2090570145.10^{143}}{0{,}0006148548942 \,.\, 3{,}2090570145.10^{143} + \cdots + 0{,}00017379898344 \,.\, 2{,}3954714847.10^{143}}$$

$$= 0{,}055834390095$$

$$gamma_1(2) = \frac{alpha_1(2).beta_1(2)}{\sum_{i=1}^{76} alpha_1(i).beta_1(i)}$$

$$= \frac{0{,}000172159377039 \,.\, 2{,}7276456309.10^{143}}{0{,}0006148548942 \,.\, 3{,}2090570145.10^{143} + \cdots + 0{,}00017379898344 \,.\, 2{,}3954714847.10^{143}}$$

$$= 0{,}013288327463$$

…

$$gamma_1(76) = \frac{alpha_1(76)\ beta_1(76)}{\sum_{i=1}^{76} alpha_1(76).beta_1(76)}$$

$$= \frac{0{,}00017379898344 \,.\, 2{,}3954714847.10^{143}}{0{,}0006148548942 \,.\, 3{,}2090570145.10^{143} + \cdots + 0{,}00017379898344 \,.\, 2{,}3954714847.10^{143}}$$

$$= 0{,}011781211329$$

Pada saat t=2, i=1 hingga i=N

$$gamma_2(1) = \frac{alpha_2(1)\ beta_2(1)}{\sum_{i=1}^{76} alpha_2(i).beta_2(i)}$$

$$= \frac{7{,}4482223904.10^{-6} \,.\, 2{,}3090201636.10^{141}}{7{,}4482223904.10^{-6} \,.\, 2{,}3090201636.10^{141} + \cdots + 1{,}7389917326.10^{-6} \,.\, 1{,}7922962444.10^{141}}$$

$$= 0{,}048666719193$$

$$gamma_2(2) = \frac{alpha_2(2)\ beta_2(2)}{\sum_{i=1}^{76} alpha_2(i).beta_2(i)}$$

$$= \frac{5{,}427349768.10^{-6}\ .\ 2{,}087002993.10^{141}}{7{,}4482223904.10^{-6}\ .\ 2{,}3090201636.10^{141} + \cdots + 1{,}7389917326.10^{-6}\ .\ 1{,}7922962444.10^{141}}$$

$$= 0{,}032052550394$$

…

$$gamma_2(76) = \frac{alpha_2(76)\ beta_2(76)}{\sum_{i=1}^{76} alpha_2(i).beta_2(i)}$$

$$= \frac{1{,}7389917326.10^{-6}\ .\ 1{,}7922962444.10^{141}}{7{,}4482223904.10^{-6}\ .\ 2{,}3090201636.10^{141} + \cdots + 1{,}7389917326.10^{-6}\ .\ 1{,}7922962444.10^{141}}$$

$$= 0{,}0088198057672$$

… (hingga t=T=76)

Pada saat t=76, i=1 hingga i=N

$$gamma_{76}(1) = \frac{alpha_{76}(1)\ beta_{76}(1)}{\sum_{i=1}^{76} alpha_{76}(i).beta_{76}(i)}$$

$$= \frac{1{,}119598959.10^{144}\ .\ 0{,}013157894737}{1{,}119598959.10^{144}\ .\ 0{,}013157894737 + \cdots + 1{,}9322279149.10^{145}\ .\ 0{,}0082644628099}$$

$$= 0{,}041686996211$$

$$gamma_{76}(2) = \frac{alpha_{76}(2)\ beta_{76}(2)}{\sum_{i=1}^{76} alpha_{76}(i).beta_{76}(i)}$$

$$= \frac{5{,}5794350438.10^{145}\ .\ 0{,}0082644628099}{1{,}119598959.10^{144}\ .\ 0{,}013157894737 + \cdots + 1{,}9322279149.10^{145}\ .\ 0{,}0082644628099}$$

$$= 0{,}020774393972$$

…

$$gamma_{76}(76) = \frac{alpha_{76}(76)\ beta_{76}(76)}{\sum_{i=1}^{76} alpha_{76}(i).beta_{76}(i)}$$

$$= \frac{1{,}9322279149.10^{145}\ .\ 0{,}0082644628099}{1{,}119598959.10^{144}\ .\ 0{,}013157894737 + \cdots + 1{,}9322279149.10^{145}\ .\ 0{,}0082644628099}$$

$$= 0{,}0071944312819$$

Perhitungan nilai epsilon ($\varepsilon_{t_{(i,j)}}$) :

Perhitungan nilai epsilon dilakukan dari t=1 hingga t=T-1 , i =1 hingga i=N dan j=1 hingga j=N.

Pada saat t=1, i=1 hingga i=N dan j=1 hingga j=N

$$epsilon_{1_{(1,1)}} = \frac{alpha_1(1)\ a_{11}\ b_1(O_2)\ beta_2(1)}{\sum_{i=1}^{76}\sum_{j=1}^{76} alpha_1(i)\ a_{ij}\ b_j(O_2)\ beta_2(i)}$$

$$= 0{,}0021475136627$$

$$epsilon_{1(1,2)} = \frac{alpha_1(1)\ a_{12}\ b_2(O_2)\ beta_2(1)}{\sum_{i=1}^{76}\sum_{j=1}^{76} alpha_1(i)\ a_{ij}\ b_j(O_2)\ beta_2(i)}$$

$$= 0{,}0079290264856$$

…

$$epsilon_{1(76,76)} = \frac{alpha_1(76)\ a_{76\,76}\ b_{76}(O_2)\ beta_2(76)}{\sum_{i=1}^{76}\sum_{j=1}^{76} alpha_1(i)\ a_{ij}\ b_j(O_2)\ beta_2(i)}$$

$$= 9{,}7932827053 \cdot 10^{-5}$$

Pada saat t=2, i=1 hingga i=N dan j=1 hingga j=N

$$epsilon_{2(1,1)} = \frac{alpha_2(1)\ a_{11}\ b_1(O_3)\ beta_3(1)}{\sum_{i=1}^{76}\sum_{j=1}^{76} alpha_2(i)\ a_{ij}\ b_j(O_3)\ beta_3(i)}$$

$$= 0{,}0019470661222$$

$$epsilon_{2(1,2)} = \frac{alpha_2(1)\ a_{12}\ b_2(O_3)\ beta_3(1)}{\sum_{i=1}^{76}\sum_{j=1}^{76} alpha_2(i)\ a_{ij}\ b_j(O_3)\ beta_3(i)}$$

$$= 0{,}0049206665759$$

…

$$epsilon_{2(76,76)} = \frac{alpha_2(76)\ a_{76\,76}\ b_{76}(O_3)\ beta_3(76)}{\sum_{i=1}^{76}\sum_{j=1}^{76} alpha_2(i)\ a_{ij}\ b_j(O_3)\ beta_3(i)}$$

$$= 7{,}322080326 \cdot 10^{-5}$$

… (hingga t=T-1=75)

Pada saat t=75, i=1 hingga i=N dan j=1 hingga j=N

$$epsilon_{75(1,1)} = \frac{alpha_{75}(1)\ a_{11}\ b_1(O_{76})\ beta_{76}(1)}{\sum_{i=1}^{76}\sum_{j=1}^{76} alpha_{75}(i)\ a_{ij}\ b_j(O_{76})\ beta_{76}(i)}$$

$$= 0{,}0016853628289$$

$$epsilon_{75(1,2)} = \frac{alpha_{75}(1)\ a_{12}\ b_2(O_{76})\ beta_{76}(1)}{\sum_{i=1}^{76}\sum_{j=1}^{76} alpha_{75}(i)\ a_{ij}\ b_j(O_{76})\ beta_{76}(i)}$$

$$= 0{,}0048192564552$$

…

$$epsilon_{75(76,76)} = \frac{alpha_{75}(76)\ a_{76\,76}\ b_{76}(O_{76})\ beta_{76}(76)}{\sum_{i=1}^{76}\sum_{j=1}^{76} alpha_{75}(i)\ a_{ij}\ b_j(O_{76})\ beta_{76}(i)}$$

$$= 6{,}045655325 \cdot 10^{-5}$$

Setelah nilai gamma dan epsilon terhitung seluruhnya, kemudian dapat menghitung parameter yang baru yaitu probabilitas transisi baru (a_bar_{ij}), probabilitas emisi baru ($b_bar_j(O_k)$) dan *state* awal baru (phi_bar_i).

Perhitungan parameter baru:

Perhitungan nilai phi_bar menggunakan persamaan II.6. Nilai phi_bar atau phi baru didapat dari nilai gamma pada saat t=1 dan i=1 hingga i=N.

$phi_bar_1 = gamma_1(1)$ = 0,055834390095

$phi_bar_2 = gamma_1(2)$ = 0,013288327463

… (hingga i=N)

$phi_bar_{76} = gamma_1(76)$ = 0,011781211329

Perhitungan nilai a_bar menggunakan persamaan II.7. Nilai a_bar atau a baru didapat dari sejumlah epsilon sewaktu t=1 hingga t=T-1 dibagi dengan sejumlah gamma sewaktu t=1 hingga t=T-1 untuk semua nilai epsilon dan gamma dengan i dan j =1 hingga idan j=N.

i=1 dan j =1 , t=1 hingga t=T-1

$$a_bar_{11} = \frac{epsilon_{1_{(1,1)}} + epsilon_{2_{(1,1)}} + \cdots + epsilon_{75_{(1,1)}}}{gamma_{1_{(1)}} + gamma_{2_{(1)}} + \cdots + gamma_{75_{(1)}}}$$

$$= \frac{0{,}0016853628289 + 0{,}0019470661222 + \cdots + 0{,}0016853628289}{0{,}055834390095 + 0{,}048666719193 + \cdots + 0{,}049496058679}$$

= 0,040082456535

i=1 dan j =2 , t=1 hingga t=T-1

$$a_bar_{12} = \frac{epsilon_{1_{(1,2)}} + epsilon_{2_{(1,2)}} + \cdots + epsilon_{75_{(1,2)}}}{gamma_{1_{(1)}} + gamma_{2_{(1)}} + \cdots + gamma_{75_{(1)}}}$$

$$= \frac{0{,}0079290264856 + 0{,}0049206665759 + \cdots + 0{,}0048192564552}{0{,}055834390095 + 0{,}048666719193 + \cdots + 0{,}049496058679}$$

= 0,10210456904

… (hingga i=N dan j=N)

i=76 dan j =76 , t=1 hingga t=T-1

$$a_bar_{76\,76} = \frac{epsilon_{1_{(76,76)}} + epsilon_{2_{(76,76)}} + \cdots + epsilon_{75_{(76,76)}}}{gamma_{1_{(76)}} + gamma_{2_{(76)}} + \cdots + gamma_{75_{(76)}}}$$

$$= \frac{9{,}7932827053\,.\,10^{-5} + 7{,}322080326\,.\,10^{-5} + \cdots + 6{,}045655325.\,10^{-5}}{0{,}011781211329 + 0{,}0088198057672 + \cdots + 0{,}0071944312819}$$

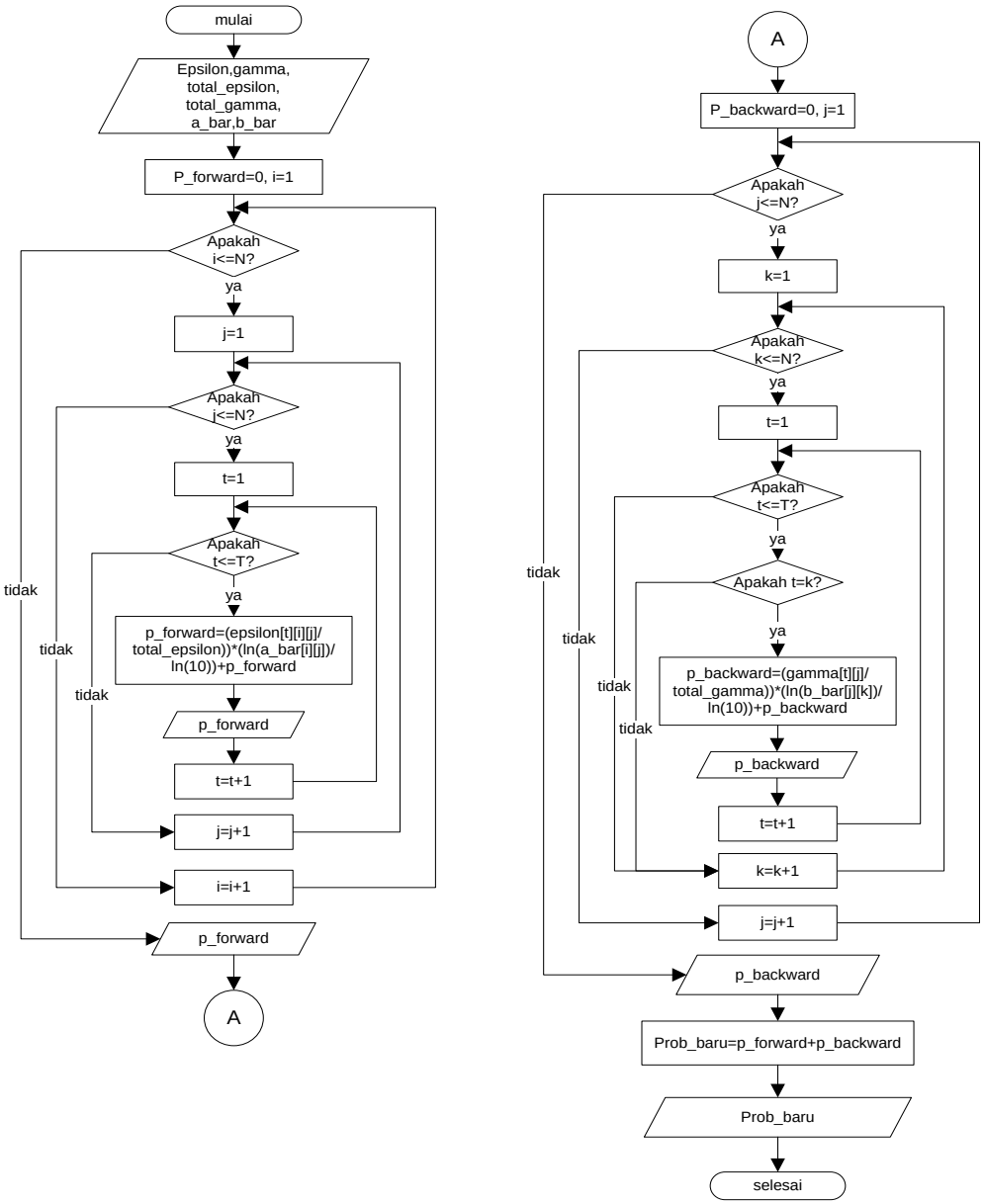
= 0,006460274079

Perhitungan nilai b_bar menggunakan persamaan II.8. Nilai b_bar atau b baru didapat dari sejumlah gamma sewaktu t=1 hingga t=T dibagi dengan sejumlah gamma sewaktu t=1 hingga t=T untuk semua nilai gamma dengan j =1 hingga j=N dan untuk pembilang pada k=t.

j=1 dan k =1 , t=1 hingga t=T $b_bar_1(1) = \frac{gamma_{1_{(1)}}}{gamma_{1_{(1)}} + gamma_{2_{(1)}} + \cdots + gamma_{76_{(1)}}}$ $= \frac{0{,}055834390095}{0{,}055834390095 + 0{,}048666719193 + \cdots + 0{,}041686996211}$ $= 0{,}015099879421$
j=1 dan k =2 , t=1 hingga t=T $b_bar_1(2) = \frac{gamma_{2_{(1)}}}{gamma_{1_{(1)}} + gamma_{2_{(1)}} + \cdots + gamma_{76_{(1)}}}$ $= \frac{0{,}048666719193}{0{,}055834390095 + 0{,}048666719193 + \cdots + 0{,}041686996211}$ $= 0{,}013161451041$
… (hingga j=N dan k=N)
j=76 dan k =76 , t=1 hingga t=T $b_bar_{76}(76) = \frac{gamma_{76_{(76)}}}{gamma_{1_{(76)}} + gamma_{1_{(76)}} + \cdots + gamma_{76_{(76)}}}$ $= \frac{0{,}041686996211}{0{,}055834390095 + 0{,}048666719193 + \cdots + 0{,}041686996211}$ $= 0{,}011273851387$

6. Menghitung P(O|λ) baru

Langkah selanjutnya adalah menghitung nilai P(O|λ) baru untuk dibandingkan dengan P(O|λ) yang lama. *Flowchart* perhitungan nilai P(O|λ) dapat dilihat pada Gambar III.16.



Gambar III.16 perhitungan nilai P(O|λ) baru

Perhitungan nilai P(O|λ) menggunakan persamaan II.11, untuk setiap nilai P_forward menggunakan persamaan dan nilai P_backward menggunakan persamaan II.12 dan persamaan II.13.

Pada iterasi 0, didapat nilai p_forward adalah -1,1790689771 . 10^{141} dan nilai p_backward adalah -9535470777 . 10^{142}, maka didapat nilai P(O|λ) adalah -9653377678 . 10^{142}.

Jika nilai iterasi adalah 0 atau baru pertama melakukan perhitungan nilai P(O|λ), maka nilai P(O|λ) yang dihitung akan dijadikan nilai P(O|λ) yang lama. Proses selanjutnya adalah

Kemudian pada iterasi =1 maka P(O|λ) lama ini akan dibandingkan dengan P(O|λ) baru yang akan dihitung.

7. Memeriksa nilai Iterasi dan membandingkan nilai P(O|λ)

Apabila nilai P(O|λ) baru atau P_bar(O|λ_bar) lebih besar dari P(O|λ) lama atau nilai iterasi lebih kecil batas iterasi maka perhitungan akan diulang dengan menghitung kembali pada proses menghitung probabilitas forward. Semua parameter baru menjadi parameter lama yaitu probabilitas transisi, probabilitas emisi, state awal dan juga P(O|λ) baru menjadi P(O|λ) lama atau P_bar(O|λ_bar) , untuk perhitungan ulang hingga parameter optimal. Nilai parameter dikatakan optimal apabila nilai P(O|λ) baru lebih besar dari P(O|λ) lama.

Jika sebaliknya, nilai P(O|λ) baru lebih kecil dari P(O|λ) lama dan iterasi mencapai batasnya maka stop_iterasi bernilai true dan melanjutkan keproses pemeriksaan batas iterasi dan kondisi stop iterasi.

Nilai P(O|λ) yang didapat pada proses sebelumnya adalah -9653377678 . 10^{142}, iterasi bernilai 0 dan batas iterasi adalah 50, sehingga belum mencapai batas iterasi. Karena nilai P(O|λ) adalah P(O|λ) yang pertama atau iterasi bernilai 0 maka perhitungan kembali pada langkah perhitungan probabilitas forward hingga mendapatkan nilai parameter dan P(O|λ) yang baru. Maka, berlanjut pada proses mengisikan parameter lama dengan parameter baru dan P(O|λ) menjadi P(O|λ)baru.

8. Parameter Baru menjadi Parameter Lama dan P(O|λ) Baru menjadi P(O|λ) lama

Pada proses ini parameter baru akan menjadi parameter lama dan P(O|λ) baru atau P_bar(O|λ_bar) menjadi P(O|λ) lama serta nilai iterasi ditambahkan satu.

$a_{ij} = a_bar_{ij,}$

$b_j(O_k) = b_bar_j(O_k)$,

$phi_i = phi_bar_i$,

P(O|λ) = P_bar(O|λ_bar) atau prob_baru dan

Iterasi=iterasi+1=0+1=1

Selanjutnya kembali pada proses perhitungan probabilitas forward hingga nilai parameter optimum.

9. Memeriksa Batas Iterasi dan kondisi stop iterasi

Pada proses ini akan diperiksa apakah nilai iterasi masih bernilai kurang dari batas iterasi dan stop_iterasi bernilai false. Jika demikian, maka proses akan berlanjut pada perhitungan probabilitas forward hingga parameter optimal. Jika sebaliknya, maka proses perhitungan berhenti dan mendapatkan nilai parameter. Pada contoh kasus, nilai iterasi hanya sampai bernilai 1 karena nilai P(O|λ) pada iterasi ke-2 lebih kecil dari nilai P(O|λ) pada iterasi 1 yaitu -2,5039273104 . 10^{143}.

10. Mendapatkan Nilai Probabilitas Transisi, Probabilitas Emisi dan State Awal

Data keluaran yang didapat pada algoritma *Baum Welch* adalah nilai probabilitas transisi, probabilitas emisi dan state awal. Nilai-nilai ini akan digunakan untuk menentukan kata kunci dokumen. Nilai prbabilitas transisi dapat dilihat pada tabel III.17, nilai probabilitas emisi dapat dilihat pada tabel III.18 dan nilai state awal dapat dilihat pada tabel III.19.

Tabel III.17 Nilai probabilitas transisi

a_{ij}	**j=1** **(waduk)**	**j=2** **(wonogiri)**	**j=3** **(pengendali)**	**j=4** **(banjir)**	**j=5** **(bengawan)**	**j=…** **(…)**	**j=N=76** **(erosi)**
i=1 **(waduk)**	a_{11} = 0,037567827563 waduk waduk	a_{12} = 0,078629927602 waduk wonogiri	a_{13} = 0, 01311131361 waduk pengendali	a_{14} =0, 013371500792 waduk banjir	a_{15} =0, 008417649533 waduk bengawan	a_{1j} waduk kata ke-j	a_{1N} =0, 006020009495 waduk solo
i=2 **(wonogiri)**	a_{21} 0,042568039183 wonogiri waduk	a_{22} 0,023157137684 wonogiri wonogiri	a_{23} =0,062221111736 wonogiri pengendali	a_{24} =0,015150768982 wonogiri banjir	a_{25} =0, 008417649533 wonogiri bengawan	a_{2j} wonogiri kata ke-j	a_{2N} =0, 0068212676391 wonogiri erosi
i=3 **(pengendali)**	a_{31} = 0,044574376097 pengendali waduk	a_{32} = 0,024335317775 pengendali wonogiri	a_{33} =0,015554793171 pengendali pengendali	a_{34} =0,11654432843 pengendali banjir	a_{35} =0, 008417649533 pengendali bengawan	a_{3j} pengendali kata ke-j	a_{3N} =0, 0071426636757 pengendali erosi
i=4 **(banjir)**	a_{41} = 0,065843876034 banjir waduk	a_{42} = 0,02385907817 banjir wonogiri	a_{43} =0,01525036108 banjir pengendali	a_{44} =0,015554520349 banjir banjir	a_{45} =0,10841131495 banjir bengawan	a_{4j} banjir kata j	a_{4N} =0, 0070028262971 banjir erosi
i=5 **(bengawan)**	a_{51} = 0,044684505426 bengawan waduk	a_{52} = 0,02474602714026 bengawan wonogiri	a_{53} =0,015592031983 bengawan pengendali	a_{54} =0,015904475231 bengawan banjir	a_{55} = 0,008417649533 bengawan bengawan	a_{5j} bengawan kata j	a_{5N} =0, 0071604128138 bengawan erosi
i=… **(...)**	a_{i1} kata ke-i waduk	a_{i2} katake-i wonogiri	a_{I3} kata ke-i pengendali	a_{i4} kata ke-i banjir	a_{i5} kata ke-i bengawan	a_{ij} kata ke-i kata ke-j	a_{iN} kata ke-i erosi
i=N=76 **(erosi)**	a_{N1} = 0,048565353364 erosi waduk	a_{N2} = 0, 026740634531 erosi wonogiri	a_{N3} =0,016946163528 erosi pengendali	a_{N4} =0,017285702431 erosi banjir	a_{N5} =0, 0084524224341 erosi bengawan	a_{Nj} erosi kata j	a_{NN} =0, 0077822771868 erosi erosi

Tabel III.18 Nilai probabilitas emisi

$b_jO(k)$	**k=1** **O(1)=waduk**	**k=2** **O(2)=wonogiri**	**k=3** **O(3)=pengendali**	**k=4** **O(4)=banjir**	**k=5** **O(5)=bengawan**	**k=…** **(…)**	**k=N** **O(N)=erosi**
j=1 **(waduk)**	$b_1O(1)$ =0,015424924993 waduk waduk	$b_1O(2)$ =0,012981018068 waduk wonogiri	$b_1O(3)$ =0, 013076950425 waduk pengendali	$b_1O(4)$ =0, 013152023957 waduk banjir	$b_1O(5)$ =0, 013313792058 waduk bengawan	$b_1O(k)$ = … waduk kata ke-j	$b_1O(N)$ =0, 011319696429 waduk erosi
j=2 **(wonogiri)**	$b_2O(1)$ =0,0064331862867 wonogiri waduk	$b_2O(2)$ = 0,027038222562 wonogiri wonogiri	$b_2O(3)$ =0,01301212424 wonogiri pengendali	$b_2O(4)$ =0,013054734627 wonogiri banjir	$b_2O(5)$ =0, 013174147732 wonogiri bengawan	$b_2O(k)$ = … merupakan kata ke-j	$b_2O(N)$ =0, 01274370732 wonogiri erosi
j=3 **(pengendali)**	$b_3O(1)$ =0, 0098375812866 pengendali waduk	$b_3O(2)$ =0,012677947658 pengendali wonogiri	$b_3O(3)$ = 0, 019418132348 pengendali pengendali	$b_3O(4)$ =0, 013104834638 pengendali banjir	$b_3O(5)$ =0, 013219904912 pengendali bengawan	$b_3O(k)$ = … pengendali kata ke-j	$b_3O(N)$ =0, 013401505635 pengendali erosi
j=4 **(banjir)**	$b_4O(1)$ =0, 0098592179945 banjir waduk	$b_4O(2)$ =0, 01210939717 banjir wonogiri	$b_4O(3)$ =0, 013056342153 banjir pengendali	$b_4O(4)$ = 0, 017708011155 banjir banjir	$b_4O(5)$ =0, 013249403401 banjir bengawan	$b_4O(k)$ = … banjir kata j	$b_4O(N)$ =0, 013148577817 banjir erosi
j=5 **(bengawan)**	$b_5O(1)$ = 0,017977442578 bengawan waduk	$b_5O(2)$ =0, 01244608068 bengawan wonogiri	$b_5O(3)$ =0, 013031323118 bengawan pengendali	$b_5O(4)$ =0, 013126279314 bengawan banjir	$b_5O(5)$ = 0,0090845883421 bengawan bengawan	$b_5O(k)$ = … bengawan kata j	$b_5O(N)$ =0, 0071604128138 bengawan erosi
j=… **(...)**	$b_jO(1)$ kata ke-i waduk	$b_jO(2)$ katake-i wonogiri	$b_jO(3)$ kata ke-i pengendali	$b_jO(4)$ kata ke-i banjir	$b_jO(5)$ kata ke-i bengawan	$b_jO(k)$ kata ke-i kata ke-j	$b_jO(N)$ kata ke-i erosi
j=N **(erosi)**	$b_NO(1)$ =0, 018114978031 erosi waduk	$b_NO(2)$ =0, 012519957873 erosi wonogiri	$b_NO(3)$ =0, 013109161007 erosi pengendali	$b_NO(4)$ =0, 0132092184044 erosi banjir	$b_NO(5)$ =0, 013215916047 erosi bengawan	$b_NO(k)$ erosi kata j	$b_NO(N)$ =0, 009702647883 erosi erosi

Tabel III.19 Nilai state awal

phi_i	**1/N**
i=1 (waduk)	0,053307762415
i=2 (wonogiri)	0,013181750812
i=3 (pengendali)	0,012707377815
i=4 (banjir)	0,012971655128
i=5 (bengawan)	0,012797969505
i=… (...)	…
i=N (erosi)	0,011786595682

3.1.1.3 Penentuan Kata Kunci

Kata kunci yang dihasilkan terdiri dari gabungan 2 kata yaitu pasang kata yang terdapat didalam abstrak. Menghitung probabilitas pasang kata yang terdapat pada abstrak. Pasangan kata dapat dilihat pada Tabel III.13. Perhitungan probabilitas pasang kata menggunkan persamaan 2.1.

Probabilitas pasangan ke-1 (waduk wonogiri):

$= Phi_i.b_i(O_i).a_{ij}.b_j(O_j)$

$= Phi_{kata1}.b_{kata1}(O_{kata1}).a_{kata1kata2}.b_{kata2}(O_{kata2})$

$= 0{,}0082644628099 \cdot 0{,}009604741763 \cdot 0{,}09762618136 \cdot 0{,}013027241673$

$= 1{,}00953 \cdot 10^{-7}$

Perhitungan dilakukan sampai pasang kata terakhir.

Setelah probabilitas pasangan kata yang mungkin terdapat pada abstrak dihitung seluruhnya, kemudian diurutkan berdasarkan nilai terbesar. Nilai 5 terbesar akan dijadikan kata kunci abstrak dokumen tersebut. Dipilih 5 karena kata kunci yang akan digunakan untuk abstrak dokumen adalah sejumlah 5 kata kunci. Hasil dari kata kunci beserta nilai yang didapat dapat dilihat pada tabel III.20.

Tabel III.20 Hasil kata kunci dan nilainya

Kata kunci	nilai
laju erosi	$2.9936883429487 . 10^{-6}$
waduk wonogiri	$1.67176946846864 . 10^{-6}$
aliran permukaan	$9.59154052783903 . 10^{-7}$
permukaan dianalisis	$9.18232223600755 . 10^{-7}$
erosi permukaan	$9.08469699116112 . 10^{-7}$
erosi mengendap	$8.9717536174468 . 10^{-7}$
erosi aliran	$8.60028203286994 . 10^{-7}$
waduk multi	$8.54437604551146 . 10^{-7}$
waduk berkurang	$8.29453287441937 . 10^{-7}$
pengendali banjir	$6.93127508506939 . 10^{-7}$
$\sum kata\ kunci = 10$	

Kata kunci yang digunakan pada lingkungan perpustakaan Pusat Penelitian dan Pengembangan Sumber Daya Air berjumlah 5 kata kunci , maka kata kunci yang akan diambil adalah 5 kata kunci teratas atau dengan nilai 5 terbesar yaitu laju erosi, waduk wonogiri, aliran permukaan, permukaan dianalisis dan erosi permukaan.

3.1.3 Penghitungan nilai Kompleksitas Waktu Algoritma *Baum Welch* terhadap seluruh sampel abstrak dokumen

Beberapa abstrak yang diambil dari lingkungan perpustakaan Pusat Penelitian dan Pengembangan Sumber Daya Air yang dijadikan sampel penelitian dalam perhitungan performansi Algoritma *Baum Welch* adalah abstrak sejumlah 40 abstrak yang diambil secara acak. Abstrak yang diteliti dalam penelitian ini dilihat dari kriteria sebagai berikut:

1. abstrak yang mengandung kata tidak penting atau *stopword*
2. abstrak yang mengandung angka
3. abstrak yang mengandung simbol

40 dokumen abstrak diklasifikasikan berdasarkan banyaknya kata dalam abstrak. Hasil dari pengklasifikasian abstrak adalah sebagai berikut:

Tabel III.21 Klasifikasi dokumen abstrak

Total kata abstrak	Jumlah dokumen abstrak
<150 kata	12
150 – 200 kata	19
< 350 kata	9

Terdapat 12 dokumen abstrak yang mengandung kata kurang dari 150 kata, 19 dokumen abstrak yang mengandung 150-200 kata dan 9 dokumen abstrak yang mengandung kurang dari 350 kata.

40 dokumen abstrak kemudian dihitung nilai parameternya dan dihitung nilai kompleksitas waktu asimptotik. Hasil perhitungan yang didapat dari seluruh abstrak dapat dilihat pada Tabel III.17. nilai kompleksitas waktu asimptotik pertama didapatkan dari kata abstrak tanpa penghilangan kata tidak penting *(stopword)* dan nilai kompleksitas waktu asimptotik yang kedua yaitu pada kolom ke-9 didapatkan dari kata abstrak yang telah mengalami penghilangan kata tidak penting *(stopword).*

Tabel III.22 Hasil perhitungan nilai kompleksitas waktu asimptotik dan parameter lain

Abstrak Ke-	Jumlah Seluruh Kata	N	I	Kompleksitas Waktu Asimptotik	Jumlah *Stopword*	N (baru)	I	Kompleksitas Waktu Asimptotik	Jumlah Kata Kunci Awal
1	72	52	1	140608	32	32	1	32768	5
2	82	55	1	166375	35	37	1	50653	5
3	103	64	1	262144	51	30	1	27000	5
4	108	52	1	140608	41	52	1	140608	5
5	109	69	1	328509	36	44	1	85184	5
6	130	89	1	704969	48	67	31	9323653	7
7	130	88	1	681472	48	66	1	287496	4
8	133	79	1	493039	43	79	1	493039	4
9	140	98	1	941192	57	72	1	373248	3
10	143	80	1	57500	50	80	1	57500	4
11	144	94	1	830584	68	94	1	830584	5
12	147	78	1	474552	59	56	1	175616	4
13	153	108	1	1259712	60	108	1	1259712	4
14	155	101	1	1030301	82	59	1	205379	4
15	159	95	1	857375	84	50	1	125000	3
16	169	108	1	1259712	70	71	1	357911	5
17	173	102	1	106758	60	69	1	328509	3
18	176	103	1	1092727	88	72	1	373248	4

Abstrak Ke-	Jumlah Seluruh Kata	N	I	Kompleksitas Waktu Asimptotik	Jumlah *Stopword*	N (baru)	I	Kompleksitas Waktu Asimptotik	Jumlah Kata Kunci Awal
19	179	76	1	1771561	77	75		421875	4
20	183	122	1	1815848	76	76	1	438976	4
21	187	109	1	1295029	76	109	1	1295029	6
22	189	90	1	729000	76	65	1	274625	4
23	191	117	1	1601613	71	81	1	531441	3
24	191	104	1	1124864	79	74	1	405224	5
25	193	112	1	1404928	82	81	1	531441	4
26	195	113	1	1442897	66	77	1	456533	3
27	195	119	1	1685159	76	909	1	729000	4
28	196	114	1	1481544	60	78	1	474552	3
29	197	116	1	1560896	88	87	1	658503	1
30	199	112	1	1404928	84	78	1	474552	3
31	200	99	1	970299	77	72	1	373248	6
32	203	117	1	1601613	84	87	1	1601613	4
33	309	113	1	1442897	93	79	1	493039	3
34	217	114	1	1481544	114	73	1	389017	3
35	225	128	1	2097152	91	128	1	2097152	4
36	227	130	1	2197000	74	101	1	1030301	5

Abstrak Ke-	Jumlah Seluruh Kata	N	I	Kompleksitas Waktu Asimptotik	Jumlah *Stopword*	N (baru)	I	Kompleksitas Waktu Asimptotik	Jumlah Kata Kunci Awal
37	230	75	1	1728000	88	85	1	614125	4
38	230	140	1	2744000	92	101	1	1030301	3
39	259	136	1	2515456	115	105	1	1157625	6
40	334	159	50	4019679	148	159	1	4019679	5

Keterangan : N adalah jumlah kata pada abstrak tanpa pengulangan kata.

I adalah jumlah iterasi yang dihasilkan melalui algoritma *Baum Welch*.

Berikut adalah hasil perhitungan yang diambil beberapa dari setiap klasifikasi dokumen dengan jumlah seluruh kata tiap abstrak memiliki jumlah kata abstrak yang sama.

Tabel III.23 Hasil penelitian dengan jumlah kata abstrak yang sama dari ketiga klasifikasi dokumen

Abstrak Ke-	**Jumlah Seluruh Kata**	**N**	**I**	**Nilai $O(N^4)$**	**N (baru)**	**Jumlah *Stopword***	**I**	**Nilai $O(N^4)$**
6	130	89	1	704969	67	48	31	9323653
7	130	88	1	681472	66	48	1	287496
26	195	113	1	1442897	77	66	1	456533
27	195	119	1	1685159	90	76	1	729000
37	230	75	1	1728000	85	88	1	614125
38	230	140	1	2744000	101	92	1	1030301

Dapat dilihat pada tabel di atas, terdapat beberapa dokumen yang memiliki jumlah abstrak yang sama yaitu abstrak ke-6 dan abstrak ke-7 dengan jumlah seluruh kata 130 kata, abstrak ke-26 dan abstrak ke-27 dengan jumlah seluruh kata 195 kata dan abstrak ke-37 dan abstrak ke-38 dengan jumlah seluruh kata 230 kata. Dari ketiga pasang dokumen yang memiliki jumlah keseluruhan kata abstrak yang sama ini, nilai N yang lebih besar mempengaruhi nilai kompleksitas waktu asimptotik lebih besar pula. Nilai N yang cenderung kecil akan menurunkan nilai $O(N^4)$. Pada abstrak ke-26 dan abstrak ke-27 dengan nilai seluruh kata abstrak adalah 195, pada nilai N yang kecil yaitu 113 memiliki nilai kompleksitas waktu asimptotik adalah 1442897 sedangkan abstrak ke-27 dengan nilai N 119 (lebih besar dari abstrak ke-26) memiliki nilai kompleksitas waktu asimptotik lebih besar yaitu 1685159. Begitu pula dengan kedua pasang dokumen yang lain nilai N yang lebih kecil memiliki nilai kompleksitas waktu asimptotik yang lebih kecil dari pasang abstrak yang lain yang memiliki nilai lebih besar.

Pada keenam dokumen abstrak di atas, Setelah dilakukan penghilangan kata tidak penting (*stopword*) maka nilai kompleksitas waktu asimptotik menjadi kecil dari nilai kompleksitas waktu asimptotik awal sebelum dilakukan penghilangan kata tidak penting (*stopword*) kecuali abstrak ke-6 memiliki nilai kompleksitas lebih besar sebelum dilakukan penghilangan kata tidak penting (*stopword*) dikarenakan memiliki nilai iterasi yang berbeda atau lebih besar dari kelima dokumen abstrak yang lain yaitu bernilai 31. Pada abstrak ke-37 sebelum dilakukan penghilangan kata tidak penting (*stopword*) memiliki nilai kompleksitas 1728000, setelah dilakukan penghilangan kata tidak penting (*stopword*) nilai kompleksitas menjadi 614125. Begitu pun dengan seluruh dokumen yang diuji mengalami penurunan nilai kompleksitas waktu asimptotik saat dilakukan penghilangan kata tidak penting (*stopword*), baik jumlah seluruh kata abstrak <150 kata, abstrak dengan 75-250 kata atau pun abstrak dengan jumlah seluruh kata <350 kata. Penghilangan *stopword* dapat menurunkan nilai kompleksitas waktu asimptotik dikarenakan akan menurunkan nilai N, dimana nilai N adalah faktor yang menentukan besarnya nilai kompleksitas waktu asimptotik algoritma. Abstrak yang memiliki kata tidak penting (*stopword*) lebih sedikit akan mengurangu nilai kompleksitas waktu asimptotik.

Tabel III.24 Hasil penelitian dengan nilai iterasi lebih dari 1

Abstrak Ke-	**Jumlah Seluruh Kata**	**N**	**I**	**Nilai $O(N^4)$**	**N (baru)**	**Jumlah *Stopword***	**I**	**Nilai $O(N^4)$**
6	130	89	1	704969	67	48	31	9323653
40	334	159	50	200983950	159	148	1	4019679

Pada abstrak ke-6 memiliki jumlah N yang lebih kecil yaitu 130 sedangkan abstrak ke-40 dengan N adalah 159. Akan tetapi nilai kompleksitas yang dihasilkan lebih besar pada abstrak ke-40 yang memilliki nilai N yang lebih besar. Ini dikarenakan jumlah iterasi pada abstrak ke-17 bernilai 50, sedangkan abstrak ke-40 bernilai 1. Begitu pula pada abstrak ke-6, pada awal sebelum dilakukan penghilangan kata tidak penting (*stopword*) nilai kompleksitas waktu asimptotik yang dihasilkan adalah 704969. Setelah dilakukan penghilangan kata tidak

penting (*stopword*), nilai N memang menjadi lebih kecil yang akan memperkecil nilai kompleksitas. Namun, pada abstrak ke-6 ini nilai kompleksitas waktu asimptotik menjadi lebih besar setelah dilakukan penghilangan kata tidak penting. Ini dikarenakan nilai I (iterasi) lebih besar dari sebelumnya yaitu 31. Hal ini membuktikan bahwa nilai I (iterasi) dapat mempengaruhi nilai kompleksitas waktu asimptotik, walaupun nilai N lebih kecil dari abstrak yang lain atau dari sebelum dilakukan penghilangan kata tidak penting (*stopword*).

Dari keseluruhan data abstrak yang diujikan, baik dokumen dengan abstrak <150 kata, abstrak dengan 150 – 200 kata dan abstrak dengan <350 kata. 95% dokumen abstrak yang diujikan melakukan iterasi sebanyak 1 kali.

Sebuah kata kunci biasanya terdiri dari kata benda dan kata sifat atau gabungan dari keduanya. Untuk menentukan kata kunci yang tepat pada sebuah dokumen maka sebuah abstrak harus memiliki sedikit angka atau simbol.

Berikut adalah hasil perhitungan angka dan simbol yang terdapat pada abstrak Perpustakaan Pusat Penelitian Dan Pengembangan Sumber Daya Air.

Tabel III.25 Persentase angka dan simbol pada abstrak

Jumlah angka	**Jumlah abstrak**	**Persentase abstrak**	**Jumlah simbol**	**Jumlah abstrak**	**Persentase abstrak**
0	12	30%	0	23	57,5%
1-10	25	62,5%	1-10	16	40%
>10	3	12%	>10	1	2,5%

Abstrak yang mengandung angka lebih dari setengah dari total abstrak yang diujikan yaitu 25 jumlah abstrak atau 62,5% dokumen abstrak. Namun, jumlah abstrak yang mengandung simbol didalamnya sebagian besar bernilai 0 yaitu 57,5% abstrak atau 23 abstrak dari total abstrak yang diujikan. Sebagian besar abstrak pada Pusat Penelitian dan Pengembangan Sumber Daya Air mengandung angka didalamnya, sedangkan yang mengandung simbol lebih sedikit. Untuk itu, selain pengurangan jumlah kata tidak penting (*stopword*) diperlukan juga pengurangan angka pada abstrak untuk memperoleh hasil kata kunci yang sesuai dengan kriteria kata kunci.