BAB II
LANDASAN TEORI

2.1 Sistem Temu Kembali (Information Retrieval)

Information Retrieval pada dasarnya merupakan proses untuk menentukan dokumen dalam koleksi yang harus ditemubalikkan untuk memenuhi keinginan pengguna akan informasi. Sistem Temu Balik Informasi (*Information Retrieval*) digunakan untuk menemukan kembali informasi-informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis. Salah satu aplikasi umum dari sistem temu kembali informasi adalah *search-engine* atau mesin pencarian yang terdapat pada jaringan internet. Pengguna dapat mencari halaman-halaman Web yang dibutuhkannya melalui mesin tersebut. Informasi yang diinginkan pengguna direpresentasikan dalam bentuk *query* dan mengandung satu atau lebih *term* yang akan digunakan dalam pencarian, selain itu dapat juga ditambahkan informasi lain seperti bobot pada tiap *term* tersebut.

Secara prinsip, penyimpanan informasi dan penemuan kembali informasi adalah hal yang sederhana. Misalkan terdapat tempat penyimpanan dokumen-dokumen dan seseorang (*user*) merumuskan suatu pertanyaan (*request* atau *query*) yang jawabannya adalah himpunan dokumen yang mengandung informasi yang diperlukan yang diekspresikan melalui pertanyaan *user*. *User* bisa saja memperoleh dokumen-dokumen yang diperlukannya dengan membaca semua dokumen dalam tempat penyimpanan, menyimpan dokumen-dokumen yang relevan dan membuang dokumen lainnya. Hal ini merupakan *perfect retrieval*, tetapi solusi ini tidak praktis. Karena user tidak memiliki waktu atau tidak ingin menghabiskan waktunya untuk

membaca seluruh koleksi dokumen, terlepas dari kenyataan bahwa secara fisik user tidak mungkin dapat melakukannya.

Keputusan menemubalikkan dokumen dibuat dengan membandingkan *term-term* dari *query* dengan *term* indeks yang terdapat dalam dokumen tersebut. Keputusan yang diambil dapat merupakan keputusan biner *(retrieve / reject*), atau melibatkan perkiraan *relevance degree* dokumen terhadap *query*.

Ukuran efektifitas pencarian ditentukan oleh *precision* dan *recall*. *Precision* adalah rasio jumlah dokumen relevan yang ditemukan dengan total jumlah dokumen yang ditemukan oleh search-engine. Precision mengindikasikan kualitas himpunan jawaban, tetapi tidak memandang total jumlah dokumen yang relevan dalam kumpulan dokumen[1].

$$Precision = \frac{|\{relevant\ documents\}| \cap |\{documents\ retrived\}|}{|\{documents\ retrieved\}|}$$

Recall adalah rasio jumlah dokumen relevan yang ditemukan kembali dengan total jumlah dokumen dalam kumpulan dokumen yang dianggap relevan.

$$Recall = \frac{|\{relevant\ documents\}| \cap |\{documents\ retrived\}|}{|\{relevant\ documents\}|}$$

Suatu sistem temu balik informasi dikatakan ideal jika sistem tersebut dapat menemukan seluruh dokumen yang relevan dan sistem hanya menemukan dokumen yang relevan saja. Akan tetapi, *term-term* yang terdapat di dokumen dan di *query* sering memiliki banyak varian morfologik, sehingga pasangan *term* seperti "*computing*" dan "*computation*" tidak akan dianggap ekuivalen oleh sistem tanpa suatu bentuk *Natural Language Processing* (NLP).

[1] https://id.wikipedia.org/wiki/Sistem_temu_balik_informasi

2.2 Stemming

Stemming merupakan proses untuk mendapatkan *root/stem* atau kata dasar dari suatu kata dalam kalimat dengan cara memisahkan masing-masing kata dari kata dasar dan imbuhannya baik awalan (prefiks) maupun akhiran (sufiks). Sebagai contoh, kata bersama, kebersamaan, menyamai, akan distem ke *root word* nya yaitu "sama".

Algoritma *stemming* banyak dibutuhkan dan diaplikasikan dibidang sistem temu kembali informasi *(Information Retrieval*) dan komputasi linguistik. Dalam sistem temu kembali informasi, algoritma *stemming* digunakan untuk mengurangi perbedaan bentuk dari suatu kata dengan mengembalikannya ke dalam bentuk kata dasar, sehingga proses temu kembali informasi menjadi efektif. Dalam bidang komputasi linguistik, algoritma *stemming* digunakan untuk mengindentifikasi kata dasar yang benar secara linguistik dan membantu analisis dari suatu kalimat[2].

Algoritma *Stemming* untuk bahasa yang satu berbeda dengan algoritma *stemming* untuk bahasa lainnya. Sebagai contoh Bahasa Inggris memiliki morfologi yang berbeda dengan Bahasa Indonesia sehingga algoritma *stemming* untuk kedua bahasa tersebut juga berbeda. Pada teks berbahasa Inggris, proses yang diperlukan hanya proses menghilangkan sufiks. Sedangkan pada teks berbahasa Indonesia lebih rumit/kompleks karena terdapat variasi imbuhan yang harus dibuang untuk mendapatkan *root word* dari sebuah kata.

Imbuhan (*affixes*) pada Bahasa Indonesia terdiri dari awalan (*prefixes*), sisipan (*infixes*), akhiran (*suffixes*), bentuk perulangan (*repeated forms*) dan

---

[2] Rila Mandala, Erry Koryanti, Rinaldi Munir, Harlili, *Sistem Stemming Otomatis untuk Kata dalam Bahasa Indonesia,* Seminar Nasional Aplikasi Teknologi Informasi 2004, Yogyakarta, 2004

confixes (kombinasi dari awalan dan akhiran). Imbuhan-imbuhan yang melekat pada suatu kata harus dihilangkan untuk mengubah bentuk kata tersebut menjadi bentuk kata dasarnya.

Stemming teks berbahasa Indonesia memiliki beberapa masalah yang sangat khusus terhadap bahasa. Salah satu masalah tersebut adalah perbedaan tipe dari imbuhan-imbuhan (*affixes*), yang lain adalah bahwa awalan (*prefixes*) dapat berubah tergantung dari huruf pertama pada kata dasar. Sebagai contoh ”me-” dapat berubah menjadi ”mem-” ketika huruf pertama dari kata dasar tersebut adalah ”b”, misalnya ”membuat” (*to make*), tetapi ”me-” juga dapat berubah menjadi ”meny-” ketika huruf pertama dari kata dasar melekat adalah ”s”, misalnya ”menyapu” (*to sweep*). Selanjutnya ketika ada lebih dari satu imbuhan (*affixes*) yang melekat pada suatu kata, maka urutan untuk menghilangkan imbuhan-imbuhan (*affixes*) pada kata tersebut menjadi sangat penting. Jika dalam proses meghilangkan imbuhan-imbuhan (*affixes*) tersebut kita tidak memperhatikan urutan penghilangan imbuhan-imbuhan (*affixes*) tersebut, maka kata dasar yang benar dari kata tersebut tidak akan ditemukan. Sebagai contoh pada kata ”*di*-beri-*kan*” (*to be given*) yang diturunkan dari kata dasar ”beri” (*to give*). Jika kita menghilangkan akhiran (*suffixes*) ”kan” terlebih dahulu sebelum menghilangkan awalan (*prefix*) ”di-” maka pada proses *stemming* ini kita mendapatkan kata dasar yang benar yaitu ”beri” (*to give*), akan tetapi jika algoritma *stemming* mencoba untuk menghilangkan awalan (*prefixes*) terlebih dahulu sebelum akhiran (*suffixes*) maka hasil kata dasar yang dihasilkan dari proses *stemming* dengan menggunakan algoritma tersebut adalah ”ikan” (*fish*) (setelah menghilangkan awalan ”di” dan ”ber”) dimana ”ikan” merupakan kata

dasar yang valid yang terdapat dalam kamus, tetapi ”ikan” bukan merupakan kata dasar yang benar untuk kata turunan ”diberikan”.

Beberapa algoritma *stemming* telah banyak dikembangkan, diantaranya Lovin (1968) dan Porter (1980) pada bahasa Inggris, Savoy (1993) untuk bahasa Perancis, Xu & Croft (1998) untuk bahasa Spanyol, Ahmad, yudo & Sembok (1996), Idris (2001) untuk bahasa Malaysia dan Arin & Setiono (2002), Nazief & Adriani (1996), Vega (2001) untuk bahasa Indonesia.

Masing-masing *stemmer* memiliki kelebihan dan kekurangannya masing-masing. Efektifitas algoritma stemming dapat dipengaruhi oleh beberapa faktor:

a. Kesalahan dalam proses pemenggalan imbuhan dari kata dasarnya. Kesalahan ini dapat berupa:
   - *Overstemming*: yaitu pemenggalan imbuahan yang melebihi dari yang seharusnya. Contoh: kata “masalah” menjadi “masa”. Kesalahan ini dapat timbul karena bentuk kata dasar yang menyerupai imbuhan.
   - *Understemming*: yaitu pemenggalan imbuhan yang terlalu sedikit dari yang seharusnya. Contoh: kata belajar menjadi lajar. Kesalahan ini dapat timbul karena kekurangan pada aturan pola imbuhan yang didefinisikan.
   - *Unchange*: yaitu kasus khusus dari understemming, dimana tidak terjadi pemenggalan imbuhan sama sekali. Contoh: kata telapak, setelah pemenggalan kata dasar yang didapat tetap telapak. Kesalahn ini dapat ditimbulkan karena kekurangan pada aturan pola imbuhan yang didefinisikan.
   - *Spelling exception*: yaitu huruf pertama kata dasar yang didapat tidak benar yang diakibatkan dari pemenggalan awalan. Contoh: kata memukul menjadi

ukul. Kesalahan ini dapat ditimbulkan karena ada beberapa imbuhan yang berubah bentuk ketika ditempelkan pada suatu kata dasar. Misalnya awalan beR-, meN-, teR-, peR-, akan bergantung pada huruf pertama kata dasar dimana imbuhan tersebut ditempelkan (Contoh: ber- + ajar = belajar, pen- + lihat = penglihatan, pen- + sakit = penyakit), atau sebaliknya ada imbuhan yang mengakibatkan huruf pertama kata dasar yang ditempelinya menjadi luluh. Misalnya meng- / peng- meluluhkan huruf ‘k’ (Contoh: mengarang dari meng- dan karang) atau men- / pen- meluluhkan huruf ‘p’ (Contoh: menuai dari men- dan tuai).

b. Kekurangan dalam perumusan aturan penambahan imbuhan pada kata dasar. Hal ini dapat terjadi karena morfologi bahasa Indonesia yang kompleks, sehingga sangat sulit atau bahkan tidak mungkin untuk merumuskan aturan yang sempurna.

c. Jumlah total aturan imbuhan yang didapat berhubungan dengan efektifitas proses temu kembali. Dimana semakin banyak pola penambahan imbuhan yang dapat dirumuskan, maka proses temu kembali akan semakin efektif.[3]

2.3 Algoritma Nazief & Adriani

Algoritma Nazief & Adriani dikembangkan oleh Bobby Nazief dan Mirna Adriani (1996). Algoritma ini berdasarkan pada aturan morfologi bahasa Indonesia yang luas, yang dikumpulkan menjadi satu grup dan di-enkapsulasi pada imbuhan/*affixes* yang diperbolehkan (*allowed affixes*) dan imbuhan/*affixes* yang tidak diperbolehkan (*disallowed affixes*). Pengelompokan ini termasuk awalan

[3] *Ibid*

(*prefixes*), akhiran (*sufixes*), sisipan (*infixes*) dan kombinasi awalan dan akhiran (*confixes*). Algoritma ini menggunakan kamus kata dasar dan mendukung *recoding*, yakni penyusunan kembali kata-kata yang mengalami proses *stemming* berlebih.

Aturan morfologi Bahasa Indonesia mengelompokkan imbuhan ke dalam beberapa kategori sebagai berikut :

1. *Inflection suffixes* yakni kelompok akhiran yang tidak merubah bentuk kata dasar. Sebagai contoh, kata “duduk” yang diberikan akhiran “-lah” akan menjadi “duduklah”. Kelompok ini dapat dibagi menjadi dua : Particle (P) atau partikel, yakni termasuk di dalamnya “-lah”, “-kah”, “-tah”, dan “-pun”. *Possessive Pronoun* (PP) atau kata ganti kepunyaan, termasuk di dalamnya adalah “-ku” , “- mu”, dan “-nya”.
2. *Derivation Suffixes* (DS) yakni kumpulan akhiran asli Bahasa Indonesia yang secara langsung ditambahkan pada kata dasar yaitu akhiran “-i”, “-kan”, dan “-an”.
3. *Derivation Prefixes* (DP) yakni kumpulan awalan yang dapat langsung diberikan pada kata dasar murni, atau pada kata dasar yang sudah mendapatkan penambahan sampai dengan 2 awalan. Termasuk di dalamnya adalah :
   a. Awalan yang dapat bermorfologi (“me-”, “be-”, “pe-”, dan “te-”).
   b. Awalan yang tidak bermorfologi (“di-”, “ke-” dan “se-”).

Berdasarkan pengklasifikasian imbuhan-imbuhan di atas, maka bentuk kata berimbuhan dalam Bahasa Indonesia dapat dimodelkan sebagai berikut :

[DP+[DP+[DP+]]]KataDasar[[+DS][+PP][+P]]

Dengan model Bahasa Indonesia di atas serta aturan-aturan dasar morfologi Bahasa Indonesia, aturan yang dipergunakan dalam proses stemming algoritma Nazief-Adriani sebagai berikut :

1. Tidak semua kombinasi awalan dan akhiran diperbolehkan. Kombinasi-kombinasi imbuhan yang tidak diperbolehkan, yaitu „be-i‟, „di-an‟, „ke-i‟, „ke-kan‟, „me- an‟, „se-i‟, „se-kan‟, dan yang terakhir „te-an‟.
2. Penggunaan imbuhan yang sama secara berulang tidak diperkenankan.
3. Jika suatu kata hanya terdiri dari satu atau dua huruf, maka proses stemming tidak dilakukan.
4. Penambahan suatu awalan tertentu dapat mengubah bentuk asli kata dasar, ataupun awalan yang telah diberikan sebelumnya pada kata dasar bersangkutan (bermorfologi). Sebagai contoh, awalan “me-” dapat berubah menjadi “meng-”, “men-”, “meny-”, dan “mem- ”. Oleh karena itu, diperlukan suatu aturan yang mampu mengatasi masalah morfologi ini.[4]

Langkah – langkah Algoritma Nazief and Adriani adalah [5]:

1) Kata yang belum di-*stemming* dicari pada kamus. Jika kata itu langsung ditemukan, berarti kata tersebut adalah kata dasar. Kata tersebut dikembalikan dan algoritma dihentikan.
2) Hilangkan *Inflectional suffixes* terlebih dahulu. Jika hal ini berhasil dan sufiks adalah partikel (“-lah”, ”-kah”, “-tah” atau “-pun”), langkah ini dilakukan lagi

[4] Jelita Asian, Hugh E. Williams dan S.M.M. Tahaghoghi, *Stemming Indonesian*, Australian Computer Society Inc., Australia, 2005
[5] *Ibid*

untuk menghilangkan *inflectional possessive pronoun suffixes* (“-ku”, “-mu” atau ”-nya”). Kemudian cek di dalam kamus kata dasar, jika ditemukan, algortima dihentikan, jika tidak lanjut ke langkah nomor 3.

3) Hapus *Derivational Suffix* (“-i” atau ”-an”,”). Jika kata ditemukan dalam kamus kata dasar, maka algoritma berhenti. Jika tidak, maka lanjut ke langkah 3a:
   a. Jika “-an” telah dihapus dan huruf terakhir dari kata tersebut adalah “-k”, maka “- k” juga ikut dihapus. Jika kata tersebut ditemukan dalam kamus maka algoritma berhenti. Jika tidak ditemukan maka lakukan langkah 3b.
   b. Akhiran yang dihapus (“-i”, “- an” atau “-kan”) dikembalikan, lanjut ke langkah 4.
4) Kemudian *Derivational Prefix* (“be-”,”di-”,”ke-”,”me-”,”pe-“,”se-” dan “te-“) dihilangkan.
   a. Langkah 4 berhenti jika:
      - Terdapat kombinasi awalan dan akhiran yang tidak diijinkan seperti pada Tabel 2.1.
      - Awalan yang dideteksi saat ini sama dengan awalan yang dihilangkan sebelumnya.
      - Tiga awalan telah dihilangkan.
   b. Identifikasikan tipe awalan dan hilangkan. Awalan ada dua tipe:
      - Standar: “di-”, “ke-”, “se-” yang dapat langsung dihilangkan dari kata.
      - Kompleks: “me-”, “be-”, “pe”, “te-” adalah tipe-tipe awalan yang dapat bermorfologi sesuai kata dasar yang mengikutinya. Oleh sebab itu, digunakan aturan pada Tabel 2.2 untuk mendapatkan pemenggalan yang tepat.

c. Setelah tidak ada lagi imbuhan yang tersisa, maka algoritma ini dihentikan kemudian kata dasar tersebut dicari pada kamus, jika kata dasar tersebut ditemukan, maka algoritma dihentikan. Tetapi jika kata dasar tersebut tidak ditemukan, maka langkah 4 diulangi. Apabila ditemukan, maka keseluruhan proses dihentikan

d. Apabila setelah langkah 4 kata dasar masih belum ditemukan, maka proses recoding dilakukan dengan mengacu pada aturan pada Tabel 2.2.

5) Jika semua langkah telah dilakukan tetapi kata dasar tersebut tidak ditemukan pada kamus juga maka algoritma ini mengembalikan kata yang asli sebelum dilakukan stemming.

Tabel 2.1 Kombinasi awalan dan akhiran yang tidak diijinkan

Awalan	Akhiran yang tidak diijinkan
be-	-i
di-	-an
ke-	-i, -kan
me-	-an
se-	-i, -kan
te-	-an

Tabel 2.2 Aturan pemenggalan awalan

No	Format Kata	Pemenggalan
1	berV...	ber-V...| be-rV...
2	berCAP...	ber-CAP...where C!='r' and P!='er'...
3	berCAerV...	ber-CAerV...where C!='r'...
4	belajar...	bel-ajar.....
5	beC1erC2...	be-C1erC2...where C1!= {'r' | 'l' }...
6	terV...	ter-V...| te-rV...
7	terCerV...	ter-CerV...where C!='r'...
8	terCP...	ter-CP...where C!='r' and P!='er'...
9	teC1erC2...	te-C1erC2...where C1!='r'...
10	me {l |r |w |y}V...	me-{l|r|w|y}V.....
11	mem{b|f|v}...	mem-{b|f|v}.....
12	mempe{r|l}...	mem-pe.....
13	mem{rV|V}...	me-m{rV|V}...| me-p{rV|V}...
14	men{c|d|j|z}...	men-{c|d|j|z}.....
15	menV...	me-nV...| me-tV...
16	meng{g|h|q}...	meng-{g|h|q}.....
17	mengV...	meng-V...| meng-kV...
18	menyV...	meny-sV.....
19	mempV...	mem-pV...where V!='e'...
20	pe{w|y}V...	pe-{w|y}V.....
21	perV...	per-V...| pe-rV...

No	Format Kata	Pemenggalan
22	perCAP...	per-CAP...where C!='r' and P!='er'...
23	perCAerV...	per-CAerV...where C!='r'...
24	pem{b|f|v}...	pem-{b|f|v}.....
25	pem{rV|V}...	pe-m{rV|V}...| pe-p{rV|V}...
26	pen{c|d|j|z}...	pen-{c|d|j|z}.....
27	penV...	pe-nV...| pe-tV...
28	peng{g|h|q}...	peng-{g|h|q}.....
29	pengV...	peng-V...| peng-kV...
30	penyV...	peny-sV.....
31	pelV...	pe-lV...Exception: for "pelajar", return ajar...
32	peCerV...	per-erV...where C!={r|w|y|l|m|n}...
33	peCP...	pe-CP...where C!={r|w|y|l|m|n} and P!='er'...

Hasil penelitian yang dilakukan oleh Asian (2007)[6], menunjukkan terdapat beberapa kesalahan *stemming* pada algoritma Nazief Adriani, sehingga dilakukan beberapa pengembangan algoritma yaitu:

1. Menggunakan kamus kata yang lebih lengkap
2. Menambahkan aturan-aturan untuk kata-kata majemuk perulangan. Suatu kata perulangan, misalnya "buku-buku", maka stemming dilakukan pada satu kata "buku". Contoh lain adalah "bolak-balik", "berbalas-balasan", "seolah-olah". Untuk kasus seperti ini, proses stemming dilakukan per kata dengan

[6] Jelita Asian, *Effective Techniques for Indonesian Text Retrieval*, A thesis submitted for the degree of Doctor of Philosopy, RMIT University, Melbourne, Victoria, Australia, 2007 hal. 76

menghilangkan tanda hubungnya (-). Jika kata tersbeut memiliki akar kata yang sama, maka akan didapat bentuk tunggal dari kata tersebut. Sebagai contoh, kata “berbalas-balasan”, kata “berbalas” dan “balasan” memiliki akar kata yang sama, yaitu “balas”. Berbeda dengan kata “bolak-balik”, kata “bolak” dan “balik” memiliki akar kata yang berbeda, maka kata “bolak-balik”, akan diproses sebagai kata yang berbeda.

3. Menambahkan aturan awalan dan akhiran, serta aturan lainnya, yaitu:
    a. Menambahkan partikel (inflection suffix) “-pun”, contoh pada kata “siapapun”.
    b. Untuk tipe awalan “te”, ditambahkan aturan ke 34 seperti pada Table 2.3. Sebelumnya, kata “terpercaya” tidak diperolah akar katanya. Dengan menambahkan aturan baru tersebut, maka diperoleh akar kata yang benar yaitu “percaya”.
    c. Untuk tipe awalan “pe”, ditambahkan aturan ke 35 seperti pada Tabel 2.3, sehingga untuk kata seperti “pekerja” dan “peserta” diperoleh akar kata “kerja” dan “serta”.
    d. Untuk tipe awalan “me”, dilakukan perubahan pada aturan ke 12 seperti pada Tabel 2.4, sehingga untuk kata seperti “mempengaruhi”, diperolah akar kata yang benar yaitu “pengaruh”.
    e. Untuk tipe awalan “me”, dilakukan perubahan para aturan ke 16 seperti pada Tabel 2.4, sehingga untuk kata seperti “mengkritik”, diperolah akar kata yang benar yaitu “kritik”.

Tabel 2.3 Penambahan aturan pemenggalan awalan

No	Format	Aturan Pemenggalan
34	terC1erC2...	ter-C1erC2....where C1!=’r’...
35	peC1erC2....	pe-C1erC2....where C1!={r|w|y|l|m|n}

Tabel 2.4 Perubahan aturan pemenggalan awalan

No	Format	Aturan Pemenggalan
12	mempe...	mem-pe...
16	meng{g|h|q|k}...	meng-{g|h|q|k}...

4. Perubahan urutan proses *stemming*. Urutan proses penghilangan imbuhan, berpengaruh terhadap hasil yang didapat. Sebagai contoh kata “bertingkah” dibentuk dari awalan “ber-“ dan kata dasar “tingkah”. Algoritma akan menghilangkan terlebih dahulu akhiran “-kah” dan menghasilkan kata “berting”, selanjutnya menghilangkan awalan “ber-“, sehingga menghasilkan kata “ting”. Kasus seperti ini hanya ada pada beberapa kombinasi awalan dan partikel tertentu. Berikut adalah beberapa perubahan urutan proses stemming:
   a. Jika suatu kata memiliki awalan “be-” dan akhiran “-lah”, hilangkah awalan terlebih dahulu kemudian akhiran. Hal ini digunakan untuk kata-kata seperti “bermasalah” dan “bersekolah”, sehingga diperolah kata dasar yang benar yaitu “masalah” dan “sekolah”. Jika urutannya tidak diubah, maka kata diperolah adalah “masa” dan “seko”.

b. Jika suatu kata memiliki awalan “be-” dan akhiran “-an”, hilangkan awalan terlebih dahulu kemudian akhiran. Sebagai contoh kata “bertahan”, sehingga diperoleh kata dasar yang benar yaitu “tahan”.
c. Jika suatu kata memiliki awalan “me-“ dan akhiran “-i”, hilangkan awalan terlebih dahulu kemudian akhiran. Sebagai contoh kata “mencapai”, sehingga diperolah kata yang benar yaitu “capai”. Tanpa perubahan urutan, akan diperolah kata “capa”.
d. Jika suatu kata memiliki awalan “di-“ dan akhiran “-i”, hilangkan awalan terlebih dahulu kemudian akhiran. Sebagai contoh kata “dimulai”, sehingga diperolah kata yang benar yaitu “mulai”. Tanpa perubahan urutan, akan diperolah kata “mula”.
e. Jika suatu kata memiliki awalan “pe-“ dan akhiran “-i”, hilangkan awalan terlebih dahulu kemudian akhiran. Sebagai contoh kata “petani”, sehingga diperolah kata yang benar yaitu “tani”. Tanpa perubahan urutan, akan diperolah kata “petan”.
f. Jika suatu kata memiliki awalan “ter-“ dan akhiran “-i”, hilangkan awalan terlebih dahulu kemudian akhiran. Sebagai contoh kata “terabai”, sehingga diperolah kata yang benar yaitu “abai”. Tanpa perubahan urutan, akan diperolah kata “aba”.

2.4 Algoritma Porter

Algoritma Porter adalah *stemmer* penggabungan yang diusulkan oleh Porter. Algoritma ini didasarkan pada gagasan bahwa akhiran dalam bahasa Inggris (sekitar 1200) sebagian besar terdiri dari akhiran-akhiran kecil dan sederhana.

Algoritma ini terdiri dari lima tahap, yang mensimulasikan proses infleksi dan derivasi dari kata. Pada setiap langkah, akhiran tertentu dihapus dengan cara substitusi. Aturan substitusi diterapkan ketika suatu kondisi terpenuhi. Salah satu contoh dari kondisi tersebut adalah panjang minimal (jumlah huruf vokal-konsonan) batang yang dihasilkan. Kondisi sederhana lainnya misalnya apakah suatu kata di akhiri dengan huruf konsonan ataukah huruf vokal.

Ketika semua kondisi terpenuhi, maka aturan tersebut diterapkan, dengan menghapus akhiran dan proses dilanjutkan ke tahap berikutnya. Jika kondisi pada tahapan tersebut tidak terpenuhi, proses di lanjutkan untuk mengecek kondisi pada tahap berikutnya, begitu seterusnya sampai semua kondisi telah di coba. Proses ini berlangsung untuk kelima tahapan[7].

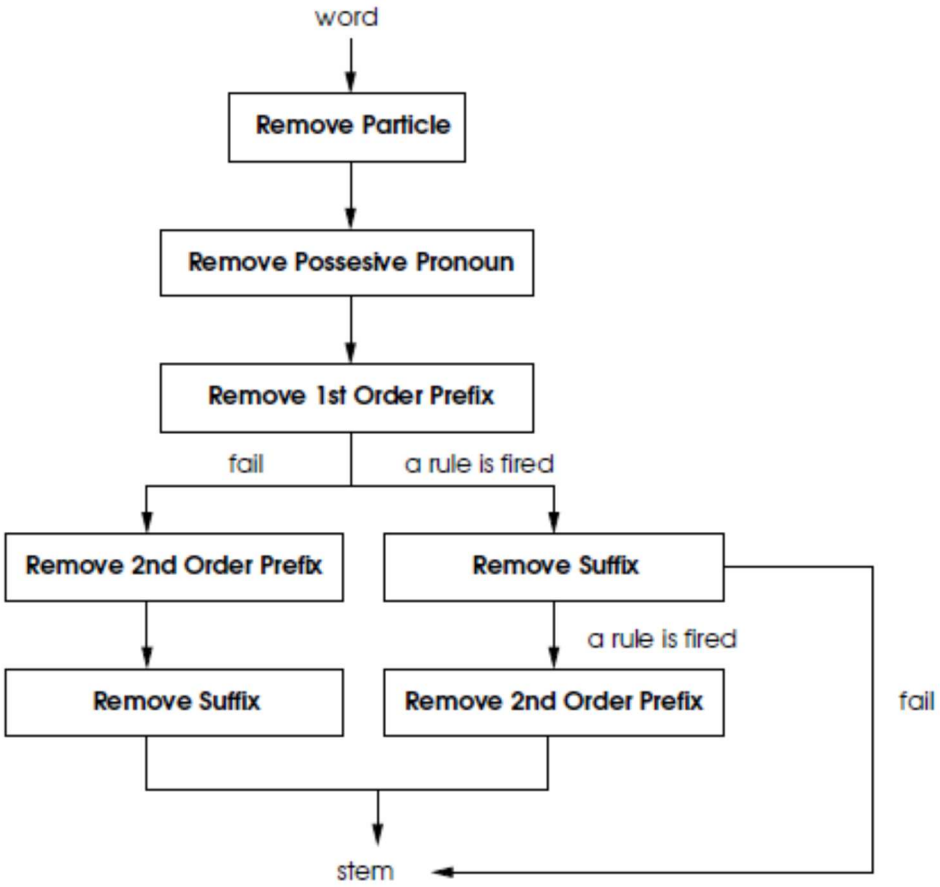
2.5 Algoritma Porter untuk Bahasa Indonesia

Algoritma Porter untuk Bahasa Indonesia dikembangkan oleh Fadillah Z Tala (2003)[8] yang mengadopsi Algortima Porter untuk Bahasa Inggri yang di kembangkan oleh W.B Frakes.

Karena bahasa Inggris datang dari kelas yang berbeda, beberapa modifikasi dilakukan agar Algoritma Porter dapat digunakan sesuai dengan Bahasa Indonesia. Desain dari *stemmer* Porter untuk Bahasa Indonesia dapat dilihat pada gambar.

---

[7] Fadillah Z Tala, *A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia*. Institute for Logic, Language and Computation Universeit Van Amsterdam, 2003 hal. 6
[8] *Ibid*, hal 6



Gambar 2.1 Algoritma Porter untuk Bahasa Indonesia

Struktur pembentukan kata dalam Bahasa Indonesia adalah sebagai berikut:

[awalan-1] + [awalan-2] + dasar + [akhiran] + [kepunyaan] + [sandang]

Masing-masing bagian tersebut (yang dalam kotak bisa ada atau tidak), digabungkan dengan kata dasar membentuk kata berimbuhan. *Stemmer* ini menggunakan *rule base* analisis untuk mencari root sebuah kata. Stemmer ini sama sekali tidak menggunakan kamus sebagai acuan, seperti halnya *stemmer* Nazief Adriani, Ahmad, Vega dan Jelita.

Algoritma / Langkah-langkah Pada Porter Stemmer adalah

1. Menghapus partikel seperti: -kah, -lah, -tah , -pun
2. Mengapus kata ganti (Possesive Pronoun), seperti –ku, -mu, -nya

3. Mengapus awalan pertama. Jika tidak ditemukan, maka lanjut ke langkah 4a, dan jika ada maka lanjut ke langkah 4b.
4. a. Menghapus awalan kedua, dan dilanjutkan pada langkah ke 5a.
   b. Menghapus akhiran, jika tidak ditemukan maka kata tersebut diasumsikan sebagai kata dasar (*root word*). Jika ditemukan maka lanjut ke langkah 5b.
5. a. Menghapus akhiran dan kata akhir diasumsikan sebagai kata dasar (*root word*).
   b. Menghapus awalan kedua dan kata akhir diasumsikan sebagai kata dasar (*root word*).

Aturan Algoritma Porter untuk Bahasa Indonesia.

Terdapat 5 aturan pada Algoritma Porter untuk Bahasa Indonesia. Aturan-aturan tersebut dapat dilihat pada tabel berikut:

Tabel 2.5 Kelompok rule pertama. *Inflectional particles*

Akhiran	**Replacement**	**Measure Condition**	**Additional Condition**	**Contoh**
kah	NULL	2	NULL	Bukukah → buku
lah	NULL	2	NULL	Pergilah → pergi
pun	NULL	2	NULL	Bukupun → buku

Tabel 2.6 Kelompok rule kedua. *Inflectional possesive pronouns*

Akhiran	**Replacement**	**Measure Condition**	**Additional Condition**	**Contoh**
ku	NULL	2	NULL	Bukuku → buku
mu	NULL	2	NULL	Bukumu → buku
nya	NULL	2	NULL	Bukunya → buku

Tabel 2.7 Kelompok rule ketiga. *First order of derivational prefixes*

Awalan	**Replacement**	**Measure Condition**	**Additional Condition**	**Contoh**
meng	NULL	2	NULL	mengukur → ukur
meny	s	2	V...*	menyapu → sapu
men	NULL	2	NULL	menduga → duga
mem	p	2	V...	memilah → pilah
mem	NULL	2	NULL	membaca → baca
me	NULL	2	NULL	merusak → rusak
peng	NULL	2	NULL	pengukur → ukur
peny	s	2	V...	penyapu → sapu
pen	NULL	2	NULL	penduga → duga
pem	p	2	V...	pemilah → pilah
pem	NULL	2	NULL	pembaca → baca
di	NULL	2	NULL	diukur → ukur
ter	NULL	2	NULL	tersapu → sapu
ke	NULL	2	NULL	kekasih → kasih

Tabel 2.8 Kelompok rule keempat. *Second order of derivational prefixes*

Awalan	**Replacement**	**Measure Condition**	**Additional Condition**	**Contoh**
ber	NULL	2	NULL	berlari → lari
bel	NULL	2	ajar	belajar → ajar
be	NULL	2	K*er...	bekerja → kerja

Awalan	Replacement	Measure Condition	Additional Condition	Contoh
per	NULL	2	NULL	perjelas → jelas
pel	NULL	2	ajar	pelajar → ajar
pe	NULL	2	NULL	pekerja → kerja

Tabel 2.9 Kelompok rule kelima. *Derivational suffixes*

Imbuhan	Replacement	Measure Condition	Additional Condition	Contoh
kan	NULL	2	Awalan bukan anggota {ke, peng}	mengukur → ukur
an	NULL	2	ajar	menyapu → sapu
i	NULL	2	K*er...	menduga → duga

Selain itu ada kombinasi awalan dan akhiran yang tidak diperbolehkan[9], yaitu:

Tabel 2.10 *Illegal confix pairs* algortima Porter

Prefiks	Sufiks
ber	i
di	an
ke	i | kan
meng	an
peng	i | kan
ter	an

[9] *Ibid*, hal 5

Dalam sebuah kata, memungkinkan adanya dua awalan yang saling berurutan. Tidak semua awalan bisa ditambahkan pada awalan lainnya dalam sebuah kata. Aturan urutan awalan yang diperbolehkan yaitu:

Tabel 2.11 Urutan awalan ganda yang diperbolehkan

Awalan 1	**Awalan 2**
meng	per
di	Ber
ter	
ke	