PERBANDINGAN ALGORITMA STEMMING PORTER DENGAN ALGORITMA NAZIEF & ADRIANI UNTUK STEMMING DOKUMEN TEKS BAHASA INDONESIA

Ledy Agusta
Fakultas Teknologi Informasi
Universitas Kristen Satya Wacana
ledyagusta@gmail.com

ABSTRACT

Information Retrieval (IR) is a process to retrieve relevant documents from set of documents in a database. Increasing amount of text documents on internet is followed by the increase of the need for effective and efficient IR tools. Search Engine is an application of IR system that depends on indexing and query expansion tools's support. Stemming is a process to transform all words in text document to their rootword form. Rootword then will be saved as index. Stemming is also used for query expansion. The appropriate algorithm will give best performance to IR system, indexing and query expansion. This research compares two Indonesian stemmers, Porter and Nazief & Adriani. 30 Indonesian language text documents have been evaluated. The evaluation of effectiveness and efficiency of the algorithms is conducted by counting the stemming's process time and precision. Based on the result of the evaluation we concluded that Nazief & Adriani is more appropriate for linguistic purpose than Porter.

Keywords: *Indonesian Stemmer, Porter, Nazief & Adriani*

1. Pendahuluan

Pencarian informasi berupa dokumen teks atau yang dikenal dengan istilah *Information Retrieval* (*IR*) merupakan proses pemisahan dokumen-dokumen yang dianggap relevan dari sekumpulan dokumen yang tersedia. Bertambahnya jumlah dokumen teks yang dapat diakses di internet diikuti dengan meningkatnya kebutuhan pengguna akan perangkat pencarian informasi yang efektif dan efisien. Efektif berarti user mendapatkan dokumen yang relevan dengan query yang diinputkan. Efisien berarti waktu pencarian yang sesingkat-singkatnya.

Stemming adalah salah satu cara yang digunakan untuk meningkatkan performa IR dengan cara mentransformasi kata-kata dalam sebuah dokumen teks ke kata dasarnya. 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. Proses stemming pada teks berbahasa Indonesia lebih rumit/kompleks karena terdapat variasi imbuhan yang harus dibuang untuk mendapatkan *root word* dari sebuah kata. Beberapa algoritma stemming Bahasa Indonesia telah dikembangkan sebelumnya. Penggunaan algoritma stemming yang sesuai mempengaruhi performa sistem IR. Dalam penelitian ini akan dibandingkan dua algoritma stemming yaitu algoritma Porter dan algoritma Nazief & Adriani.

Algoritma-algoritma stemming memiliki kelebihan dan kekurangannya masing-masing. Efektifitas algoritma stemming dapat diukur berdasarkan beberapa parameter, seperti kecepatan proses, keakuratan, dan kesalahan. Dalam tulisan ini, penulis akan membandingkan efektifitas algoritma Nazief dan Adriani dengan algoritma Porter untuk proses stemming pada teks berBahasa Indonesia, sehingga akhirnya akan diketahui algoritma manakah yang lebih cepat, lebih akurat atau yang lebih banyak melakukan kesalahan stemming. Tujuan penelitian ini adalah untuk membandingkan kemampuan dan ketepatan algoritma Nazief & Adriani dengan algoritma Porter untuk proses stemming pada teks berBahasa Indonesia.

2. Landasan Teori

2.1 Stemming

Stemming merupakan suatu proses yang terdapat dalam sistem IR yang mentransformasi kata-kata yang terdapat dalam suatu dokumen ke kata-kata akarnya (*root word*) dengan menggunakan aturan-aturan tertentu. Sebagai contoh, kata bersama, kebersamaan, menyamai, akan distem ke root wordnya yaitu "sama". Proses stemming pada teks berBahasa Indonesia berbeda dengan stemming pada teks berbahasa Inggris. Pada teks berbahasa Inggris, proses yang diperlukan hanya proses menghilangkan sufiks. Sedangkan pada teks berbahasa Indonesia, selain sufiks, prefiks, dan konfiks juga dihilangkan.

2.2 Penelitian Terdahulu

Algoritma stemming untuk beberapa bahasa telah dikembangkan, seperti Algoritma Porter untuk teks berbahasa inggris[2], Algoritma Porter untuk teks berbahasa Indonesia[2], Algoritma Nazief & Adriani untuk teks berbahasa Indonesia[4].

Algoritma yang dibuat oleh Bobby Nazief dan Mirna Adriani ini memiliki tahap-tahap sebagai berikut:

1. Cari kata yang akan distem dalam kamus. Jika ditemukan maka diasumsikan bahwa kata tesebut adalah *root word*. Maka algoritma berhenti.
2. *Inflection Suffixes* ("-lah", "-kah", "-ku", "-mu", atau "-nya") dibuang. Jika berupa *particles* ("-lah", "-kah", "-tah" atau "-pun") maka langkah ini diulangi lagi untuk menghapus *Possesive Pronouns* ("-ku", "-mu", atau "-nya"), jika ada.
3. Hapus *Derivation Suffixes* ("-i", "-an" atau "-kan"). Jika kata ditemukan di kamus, maka algoritma berhenti. Jika tidak maka 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. Hapus *Derivation Prefix*. Jika pada langkah 3 ada sufiks yang dihapus maka pergi ke langkah 4a, jika tidak pergi ke langkah 4b.
   a. Periksa tabel kombinasi awalan-akhiran yang tidak diijinkan. Jika ditemukan maka algoritma berhenti, jika tidak pergi ke langkah 4b.
   b. For i = 1 to 3, tentukan tipe awalan kemudian hapus awalan. Jika root word belum juga ditemukan lakukan langkah 5, jika sudah maka algoritma berhenti. Catatan: jika awalan kedua sama dengan awalan pertama algoritma berhenti.
5. Melakukan Recoding.
6. Jika semua langkah telah selesai tetapi tidak juga berhasil maka kata awal diasumsikan sebagai *root word*. Proses selesai.

Tipe awalan ditentukan melalui langkah-langkah berikut:

1. Jika awalannya adalah: "di-", "ke-", atau "se-" maka tipe awalannya secara berturut-turut adalah "di-", "ke-", atau "se-".
2. Jika awalannya adalah "te-", "me-", "be-", atau "pe-" maka dibutuhkan sebuah proses tambahan untuk menentukan tipe awalannya.
3. Jika dua karakter pertama bukan "di-", "ke-", "se-", "te-", "be-", "me-", atau "pe-" maka berhenti.
4. Jika tipe awalan adalah "none" maka berhenti. Jika tipe awalan adalah bukan "none" maka awalan dapat dilihat pada Tabel 2. Hapus awalan jika ditemukan.

Tabel 1. Kombinasi Awalan Akhiran Yang Tidak Diijinkan

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

Tabel 2. Cara Menentukan Tipe Awalan Untuk Kata Yang Diawali Dengan "te-"

Following Characters				Tipe Awalan
Set 1	Set 2	Set 3	Set 4	
"-r-"	"-r-"	-	-	none
"-r-"	Vowel	-	-	ter-luluh
"-r-"	not (vowel or "-r-")	"-er-"	vowel	ter
"-r-"	not (vowel or "-r-")	"-er-"	not vowel	ter-
"-r-"	not (vowel or "-r-")	not "-er-"	-	ter
not (vowel or "-r-")	"-er-"	vowel	-	none
not (vowel or "-r-")	"-er-"	not vowel	-	te

Tabel 3. Jenis Awalan Berdasarkan Tipe Awalannya

Tipe Awalan	Awalan yang harus dihapus
di-	di-
ke-	ke-
se-	se-
te-	te-
ter-	ter-
ter-luluh	ter

Untuk mengatasi keterbatasan pada algoritma di atas, maka ditambahkan aturan-aturan dibawah ini:

1. Aturan untuk reduplikasi.
   - Jika kedua kata yang dihubungkan oleh kata penghubung adalah kata yang sama maka *root word* adalah bentuk tunggalnya, contoh : "buku-buku" *root word*-nya adalah "buku".
   - Kata lain, misalnya "bolak-balik", "berbalas-balasan, dan "seolah-olah". Untuk mendapatkan *root word*-nya, kedua kata diartikan secara terpisah. Jika keduanya memiliki *root word* yang sama maka diubah menjadi bentuk tunggal, contoh: kata "berbalas-balasan", "berbalas" dan "balasan" memiliki *root word* yang sama yaitu "balas", maka *root word* "berbalas-balasan" adalah "balas". Sebaliknya, pada kata "bolak-balik", "bolak" dan "balik" memiliki *root word* yang berbeda, maka *root word*-nya adalah "bolak-balik"
2. Tambahan bentuk awalan dan akhiran serta aturannya.
   - Untuk tipe awalan "mem-", kata yang diawali dengan awalan "memp-" memiliki tipe awalan "mem-".
   - Tipe awalan "meng-", kata yang diawali dengan awalan "mengk-" memiliki tipe awalan "meng-".

Algoritma kedua yang digunakan dalam sistem ini adalah Algoritma Porter. Adapun langkah-langkah algoritma ini adalah sebagai berikut:

1. Hapus *Particle*,
2. Hapus Possesive Pronoun.
3. Hapus awalan pertama. Jika tidak ada lanjutkan ke langkah 4a, jika ada cari maka lanjutkan ke langkah 4b.
4. a. Hapus awalan kedua, lanjutkan ke langkah 5a.
   b. Hapus akhiran, jika tidak ditemukan maka kata tersebut diasumsikan sebagai *root word.* Jika ditemukan maka lanjutkan ke langkah 5b.
5. a. Hapus akhiran. Kemudian kata akhir diasumsikan sebagai *root word*
   b. Hapus awalan kedua. Kemudian kata akhir diasumsikan sebagai *root word*.

Terdapat 5 kelompok aturan pada Algoritma Porter untuk Bahasa Indonesia ini. Aturan tersebut dapat dilihat pada Tabel 4 sampai Tabel 8.

Tabel 4. Aturan Untuk Inflectional Particle

Akhiran	Replacement	Measure Condition	Additional Condition	Contoh
-kah	NULL	2	NULL	bukukah
-lah	NULL	2	NULL	pergilah
-pun	NULL	2	NULL	bukupun

Tabel 5. Aturan Untuk Inflectional Possesive Pronoun

Akhiran	Replacement	Measure Condition	Additional Condition	Contoh
-ku	NULL	2	NULL	bukuku
-mu	NULL	2	NULL	bukumu
-nya	NULL	2	NULL	bukunya

Tabel 6. Aturan Untuk First Order Derivational Prefix

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…	memaksa→ paksa
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…	pemaksa → paksa
pem-	NULL	2	NULL	pembaca → baca
di-	NULL	2	NULL	diukur → ukur
ter-	NULL	2	NULL	tersapu → sapu
ke-	NULL	2	NULL	kekasih → kasih

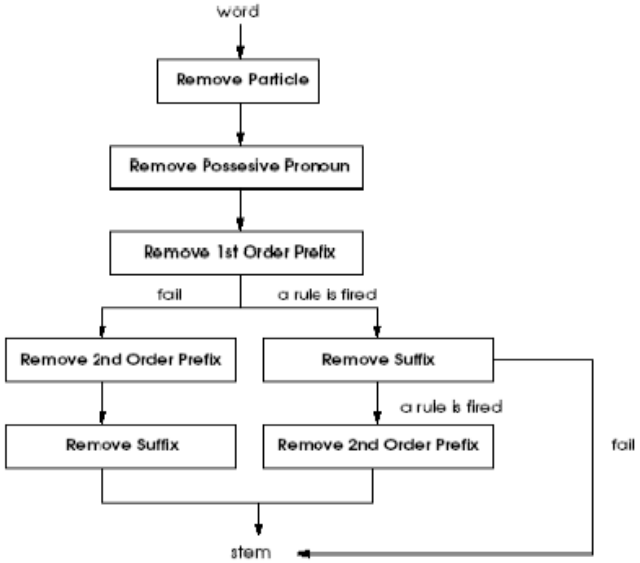
Tabel 7. Aturan Untuk Second Order Derivational Prefix

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*
per-	NULL	2	NULL	*perjelas → jelas*
pel-	NULL	2	*Ajar*	*pelajar → ajar*
pe-	NULL	2	NULL	*pekerja → kerja*

Tabel 8. Aturan Untuk Derivational Suffix

Akhiran	Replacement	Measure Condition	Additional Condition	Contoh
-kan	NULL	2	Prefix bukan anggota {ke, peng}	tarikkan → tarik, mengambilkan → ambil
-an	NULL	2	prefix bukan anggota {di, meng, ter}	makanan → makan, perjanjian → janji
-i	NULL	2	prefix bukan anggota{ber, ke, peng}	Tandai → tanda, mendapati → dapat

Proses stemming menggunakan Algoritma Porter dapat dilihat pada Gambar 1.



Gambar 1. Algoritma Porter

3. Metode Penelitian

Penelitian ini dilakukan dengan:

1. Untuk membandingkan performa masing-masing algoritma maka dibuat aplikasi sederhana proses stemming menggunakan algoritma Porter dan Nazief & Adriani.
2. Menghitung presisi dan waktu proses dari masing-masing algoritma.
3. Menguji menggunakan 30 dokumen sampel.

4. Hasil dan Pembahasan

Proses pembandingan algoritma Porter dengan Algoritma Nazief & Adriani dilakukan dengan membuat program sederhana yang memproses dokumen teks inputan sehingga diketahui stem, waktu proses, presisi dari hasil stemming dokumen tersebut.

4.1 Hasil Pengujian

Uji Coba algoritma dilakukan pada 30 dokumen teks Bahasa Indonesia dengan ukuran dokumen yang bervariasi. Hasil uji coba dokumen teks yang dilakukan pada 30 dokumen teks dapat dilihat pada Tabel 9.

Tabel 9. Tabel Waktu Proses dan Presisi Pada 30 Dokumen Teks

No.	Dokumen Teks	Jumlah kata	Algoritma Porter		Algoritma Nazief & Adriani	
			Waktu Proses (det)	Presisi (%)	Waktu Proses (det)	Presisi (%)
1.	Paper Fungsi.txt	1854	1,518	89,7	86,264	95,9
2.	Keyboard.txt	43	0,048	83,7	1,953	93
3.	Makan.txt	9	0	88,9	0,352	100
4.	Coba.txt	20	0,041	89,5	1,267	98
5.	Abstrak.txt	104	0,05	80,8	4,246	87,5
6.	Bab2.txt	1161	0,74	85,7	44,776	91,2
7.	Bab1.txt	780	0,502	83,5	27,558	93,7
8.	Kesimpulan & Saran.txt	352	0,23	81,8	12,848	91,8
9.	Kuesioner.txt	193	0,129	89,6	8,891	93,3
10.	Daftar Isi.txt	470	0,209	30,2	22,002	30,4
11.	Etika di Milis1.txt	1428	0,602	82,2	62,418	89,8
12.	Guidelines Perancangan.txt	2548	1,752	18	89,58	19
13.	Penelitian.txt	205	1,141	57,1	8,133	59,5
14.	PR.txt	319	0,209	35,1	13,797	97,8
15.	Mbah Soyo.txt	651	0,507	88,6	25,492	98,2
16.	Proses.txt	86	0,101	81,4	2,961	96,5
17.	Reduplikasi.txt	47	0,039	78,7	2,203	95,7
18.	Tata Tertib Sidang.txt	172	0,125	12,2	5,008	98,3
19.	Hacker.txt	69	0,132	8,7	2,203	10,1
20.	Kuesioner2.txt	147	0,132	27,2	6,914	27,9
21.	Algoritma.txt	684	0,13	92,6	1,391	96,3
22.	Etika Di Milis2.txt	677	0,275	85,4	29,406	95,7
23.	Etika di Milis3.txt	784	0,515	38,4	29,156	93,3
24.	Feedback.txt	73	0,471	90,4	2,443	98,6
25.	Mamas.txt	161	0,16	81,4	6,592	97,5
26.	Mailing List.txt	27	0,13	92,6	1,391	96,3
27.	Masih Ada.txt	1439	0,945	87,1	54,093	94,8
28.	Surat Peminjaman.txt	53	0,039	86,8	1,781	90,6
29.	Tata Tertib Milis.txt	118	0,169	83,9	4,957	97,5
30.	Optical Storage.txt	28	0,059	85,7	1,553	100

5. Kesimpulan

Berdasarkan perancangan dan implementasi program, maka diperoleh kesimpulan sebagai berikut:

- Proses stemming dokumen teks berBahasa Indonesia menggunakan Algoritma Porter membutuhkan waktu yang lebih singkat dibandingkan dengan stemming menggunakan Algoritma Nazief & Adriani.
- Proses stemming dokumen teks berBahasa Indonesia menggunakan Algoritma Porter memiliki prosentase keakuratan (presisi) lebih kecil dibandingkan dengan stemming menggunakan Algoritma Nazief & Adriani.
- Pada proses stemming menggunakan Algoritma Nazief & Adriani, kamus yang digunakan sangat mempengaruhi hasil stemming. Semakin lengkap kamus yang digunakan maka semakin akurat pula hasil stemming.

- Kamus yang digunakan mempengaruhi perhitungan presisi. Semakin lengkap kamus yang digunakan maka semakin akurat pula nilai presisinya.

Daftar Pustaka

[1] Asian J., Williams H. E. dan Tahaghogi, S.M.M.. (2005). *Stemming Indonesian*, Melbourne, RMIT University, http://crpit.com/confpapers/CRPITV38Asian.pdf, diakses terakhir tanggal 25 April 2008.

[2] Fadillah Z. Tala*, A Study of Stemming Effect on Information Retrieval in Bahasa Indonesia*, Netherland, Universiteit van Amsterdam, http://ucrel.lancs.ac.uk/acl/P/P00/P00-1075.pdf, diakses terakhir tanggal 25 Juli 2009.

[3] Lu, G. (1999). *Multimedia Database Management Systems*. Norwood. Artech House, Inc.

[4] Nazief, Bobby dan Mirna Adriani, *Confix-Stripping: Approach to Stemming Algorithm for Bahasa Indonesia*, Fakulty of Computer Science University of Indonesia.