Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

Team project ©2017
Dony Pratidana S. Hum | Bima Agus Setyawan S. IIP

BAB II

LANDASAN TEORI

2.1 Google Search Engine

Search engine atau mesin pencari adalah suatu program yang digunakan oleh pengguna Internet (*netter*) untuk mencari dokumen-dokumen didasarkan pada kata kunci atau *key word* dari sebuah informasi yang ingin diketahui. Dengan berpedoman pada kata kunci yang digunakan maka hasil pencarian yang relevan akan menampilkan halaman-halaman yang mengandung unsur kata kunci tersebut (Sarwono, 2010; A Better World Business Club, 2010).

Google semula adalah proyek mahasiswa Stanford University yang bernama Larry Page dan Sergey Brin yang diberi nama BackRub. Nama tersebut pada tahun 1998 diubah menjadi Google, dan setelah itu proyek tersebut kemudian dijadikan sebagai perusahaan swasta Google milik perseorangan sampai saat ini. Google dikenal sebagai alat pencarian yang paling cerdas. Google secara otomatis melakukan pencarian dengan logika Boolean (Sarwono, 2010). Google memiliki beberapa aturan dalam menginterpretasikan *query*, antara lain *logical operator*, *exact matching*, *stemming*, *stop words*, *word limit*, *term proximity*, *term order*, *case insensitive*, dan *character ignored* (Google Guide, 2009). Berikut telaah literatur terhadap beberapa aturan Google dalam interpretasi *query* yang menjadi fokus penelitian ini.

Web Search Engine..., Stephani Marcelli Djojo, FTI UMN, 2012

2.2.1 Operator Boolean

Google menggunakan metode Boolean, yaitu metode yang memungkinkan pengguna membuat kombinasi kata kunci dengan operator AND dan OR untuk menghasilkan pencarian yang sesuai dalam pencariannya (Notess, 2009).

Dalam sebuah pencarian, Google mengembalikan hanya halaman-halaman yang cocok dengan semua kata pencarian yang dimasukkan pengguna. Contohnya dalam pencarian [universitas multimedia nusantara] menemukan halaman yang mengandung "universitas" dan "multimedia" dan "nusantara". Google secara implisit menyertakan operator AND antara kata-kata dalam *query* pencarian. Untuk mendapat halaman yang mengandung beberapa (bukan semua) dari kata yang dicari digunakan operator OR.

2.2.2 Special Character Operator

Special character digunakan Google untuk memaksudkan pengguna dapat menyempurnakan *query* pencarian dan meningkatkan akurasi hasilnya (Google Guide, 2007). *Quote pharse* (" ") dan *tilde* (~) adalah dua *special character* yang digunakan oleh Google.

Sebuah *query* dengan kata-katanya dalam tanda kutip (*quote pharse*) menemukan halaman yang mengandung frase secara tepat. Misalnya, ["Larry Page"] menemukan halaman yang berisi persis frase "Larry Page". Dengan *query* ini tidak ditemukan halaman yang berisi "Larry E. Page" (Hardy, 2010).

Operator *tilde* (~) digunakan oleh Google untuk menemukan kata similar (*stemming*) dan sinonim (Google, 2007). Sekalipun tanpa operator *tilde*

sebenarnya *stemming* secara otomatis dijalankan jika kata dalam *query* pencariannya lebih dari satu kata. Misalnya, *query* [membeli] menemukan halaman yang berisi persis "membeli". Dengan *query* ini tidak ditemukan halaman yang berisi "beli". Berbeda dengan contoh berikut, [membeli ~tv] menemukan halaman yang berisi "membeli" atau "beli" dan "tv" atau "television".

2.2.3 Stemming

Stemming adalah suatu proses pemenggalan semua imbuhan baik yang terdiri dari awalan, akhiran, sisipan, perulangan dan kombinasi dari awalan dan akhiran pada kata turunan (Agusta, 2009). *Stemming* adalah salah satu cara Google dalam memperluas *query*. Cara kerja *stemming* adalah melakukan pemenggalan pada kata *query* sampai ke bentuk kata dasarnya (Smarty, 2008). *Query* [anak-anak bersepeda] akan menemukan halaman yang mengandung kata yang similar dengan kata pencarian, seperti "anak" atau "anak-anak", dan "sepeda" atau "bersepeda".

Jika menginginkan hasil pencarian halaman yang mengandung kata pencarian secara tepat (tanpa *stemming*) dapat dilakukan dengan menggunakan tanda petik dua pada *query*. *Query* ["anak-anak bersepeda"] akan menemukan halaman yang mengandung tepat kata "anak-anak bersepeda" tanpa diikutsertakannya hasil pencarian dari ekspansi *query* dengan *stemming*. Google tidak menjalanan *stemming* jika *query* terdiri dari satu kata saja (Barker, 2005). *Stemming* pada *query* satu kata dapat dilakukan dengan menambahkan operator

UNIVERSITAS MULTIMEDIA NUSANTARA

tilde (~) pada depan kata pencarian (Google Guide, 2009). *Query* [~bersepeda] akan menemukan halaman yang mengandung kata “sepeda” atau “bersepeda”.

2.2.4 Stop Words

Stop words atau kata henti adalah beberapa kata yang umum (di, ke, dari, untuk, seperti, serta, dan) dan pada umumnya tidak menambah makna pada pencarian. Kata-kata tersebut tidak dimasukkan ke dalam *query* pencarian dengan alasan kata-kata tersebut muncul di halaman web begitu banyak dan tidak membantu dalam menemukan hasil yang relevan (Sarwono, 2010).

2.2 Algoritma Stemming Nazief-Andriani

Skema *stemming* Nazief-Adriani dijelaskan pada laporan teknikal dari Universitas Indonesia yang tidak diterbitkan. Algoritma ini berdasarkan pada aturan morfologi bahasa Indonesia yang luas, dikumpulkan menjadi satu grup dan dienkapsulasi pada imbuhan yang diperbolehkan dan imbuhan yang tidak diperbolehkan (Putri, 2009).

```
[DP+[DP+[DP+]]] root-word [[+DS][+PP][+P]]        (2.1)
```

DP : *Derivation Prefix*

DS : *Derivation Suffix*

PP : *Possessive Pronoun* (*Inflection*)

P : *Particle* (*Inflection*)

Algoritma yang dibuat oleh Bobby Nazief dan Mirna Adriani ini memiliki tahap-tahap sebagai berikut (Agusta, 2009):

a. Cari kata yang akan di-*stem* dalam kamus. Jika ditemukan maka diasumsikan bahwa kata tesebut adalah *root word* dan algoritma berhenti.

b. Hapus *Inflection Suffixes* (“-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”). Jika berupa *particles* (“-lah”, “-kah”, “-tah” atau “-pun”) maka langkah ini diulangi lagi untuk menghapus *Possesive Pronouns* (“-ku”, “-mu”, atau “-nya”) jika ada.

c. Hapus *Derivation Suffixes* (“-i”, “-an” atau “-kan”). Jika kata ditemukan di kamus, maka algoritma berhenti. Jika tidak maka ke langkah c.1)

   1) 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 c.2).

   2) Akhiran yang dihapus (“-i”, “-an” atau “-kan”) dikembalikan, lanjut ke langkah d.

d. Hapus *Derivation Prefix*. Jika pada langkah c. ada sufiks yang sempat dihapus maka pergi ke langkah d.1), jika tidak pergi ke langkah d.2).

   1) Periksa tabel kombinasi awalan-akhiran yang tidak diijinkan. Jika ditemukan maka algoritma berhenti, jika tidak pergi ke langkah d.2).

   2) For i = 1 to 3, tentukan tipe awalan kemudian hapus awalan. Jika *root word* belum juga ditemukan lakukan langkah e. Jika sudah ditemukan maka algoritma berhenti. Catatan: jika awalan kedua sama dengan awalan pertama algoritma berhenti.

e. Melakukan *recording*, yaitu mengembalikan kata yang sudah di-*stemming* ke bentuk awalnya. Kata awal ini diasumsikan sebagai *root word*. Proses selesai.

Ditambahkan aturan-aturan dibawah ini untuk mengatasi keterbatasan pada algoritma *stemming* Nazief-Andriani (Agusta, 2009):

a. Aturan untuk reduplikasi
   1) 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".
   2) Kata lain, misalnya "bolak-balik". Untuk mendapatkan *root word*-nya, kedua kata diartikan secara terpisah. Pada kata "bolak-balik", "bolak" dan "balik" memiliki *root word* yang berbeda, maka *root word*-nya adalah "bolak-balik".
b. Tambahan bentuk awalan dan akhiran serta aturannya.
   1) Kata yang diawali dengan awalan "memp-" memiliki tipe awalan "mem-"
   2) Kata yang diawali dengan awalan "mengk-" memiliki tipe awalan "meng-"

UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA

Tabel 2.1 Imbuhan Bahasa Indonesia

Imbuhan		Kata dasar	Alomorf	Bunyi hilang	Contoh		Sifat
					Kata dasar	Diimbuhkan	
Awalan	me-	Berawalan huruf 'm', 'n', 'l', 'r', 'ng', 'ny', 'w', 'y'		k', 'p', 't', 's'	nganga	menganga	Derivatif
					nyanyi	menyanyi	
		Berawalan huruf 'p', 'b', 'f'	mem-		paksa	memaksa	
					protes	memprotes	
					fitnah	memfitnah	
		Berawalan huruf 't', 'd', 'j', 'c'	men-		tulis	menulis	
					capai	mencapai	
		Berawalan huruf 's'	meny-		sapu	menyapu	
		Bersuku satu	menge-		bom	mengebom	
		Berawalan huruf 'a', 'i', 'u', 'e', 'o', 'g', 'h', 'k'	meng-		ambil	mengambil	
					olah	mengolah	
					kunci	mengunci	
	pe-	sama dengan imbuhan me-		sama dengan imbuhan me-	rusak	perusak	
					nyanyi	penyanyi	
			pem-		paksa	pemaksa	
					protes	pemprotes	
					fitnah	pemfitnah	
			pen-		tulis	penulis	
					capai	pencapai	
			peny-		sapu	penyapu	
			penge-		bom	pengebom	
			peng-		ambil	pengambil	
					olah	pengolah	
					kunci	pengunci	
	per-	berawalan huruf 'r'	pe-		redam	peredam	
		kata 'ajar'	pel-		ajar	pelajar	

Sumber: Marsya & Abidin, 2011

UNIVERSITAS MULTIMEDIA NUSANTARA

Tabel 2.1 Imbuhan Bahasa Indonesia (lanjutan)

Imbuhan		Kata dasar	Alomorf	Bunyi hilang	Contoh		Sifat
					Kata dasar	Diimbuhkan	
Awalan	ber-	berawalan huruf 'r' atau bersu-ku awal 'er'	be-		kerja	bekerja	Derivatif
					runding	berunding	
		kata 'ajar'	bel-		ajar	belajar	
	ter-	berawalan huruf 'r' atau (terka-dang) ber-suku awal 'er'	te-		rasa	terasa	
					pergok	terpergok	
	di-, ke-, se-				makan	dimakan	
					tua	ketua	
					sama	sesama	
Akhiran	-i, -an, -kan				main	mainan	
					teman	temani	
					buat	buatkan	
	-kah, -lah, -tah, -pun				bukan	bukankah	Inflektif
					kapan	kapanpun	
	-ku, -mu, -nya				milik	milikku	

Sumber: Marsya & Abidin, 2011

2.3 Metode Evaluasi Paice

Metode evaluasi Paice berfungsi untuk mengevaluasi kualitas dari algoritma *stemming* dengan menghitung kesalahan-kelasahan yang terjadi selama proses *stemming*. Idealnya, sebuah *stemmer* yang baik akan mengelompokkan semua kata dari kelompok semantik yang sama untuk *root* yang sama. Namun karena ketidakteraturan dalam semua bahasa alami, semua *stemmers* mau tidak mau

membuat kesalahan, termasuk yang yang menggunakan daftar kosakata. Dengan kata lain bahwa tidak ada *stemmer* dapat diharapkan untuk bekerja dengan sempurna benar. Akan selalu ada kasus kesalahan di mana kata-kata yang seharusnya digabung tidak akan digabung ke *root* yang sama (*understemming*) atau kasus-kasus dimana kata-kata digabungkan ke *root* yang sama saat mereka benar-benar berbeda (*overstemming*). Sebuah *stemmer* yang baik jelas harus menghasilkan *overstemming* dan *understemming error* sedikit atau sekecil mungkin (Tala, 2003).

Terdapat dua macam *stemming error*, yaitu *understemming* dan *overstemming*. *Understemming errors* di mana kata-kata yang seharusnya memiliki *root* yang sama, pemenggalan imbuhannya tidak sampai pada *root* yang sama. *Overstemming errors* di mana kata-kata dikonversi ke *root* yang sama meskipun kata-kata tersebut sebenarnya memiliki *root* yang berbeda (Paice, 1994).

Paice mendefinisikan tiga kelas dari hubungan antara pasangan-pasangan kata, yaitu sebagai berikut (Tala, 2003).

a. Tipe 0 adalah dua kata yang mempunyai bentuk identik dan sudah tergabung, dengan mengabaikan kemungkinan *homographs* (duplikasi).
b. Tipe 1 adalah dua kata yang berbeda dalam bentuk tetapi memiliki sematik sama, dan
c. Tipe 2 adalah dua kata yang berbeda dalam bentuk dan mempunyai semantik yang berbeda.

UNIVERSITAS MULTIMEDIA NUSANTARA

Dalam Paice, beberapa kata dikelompokkan secara semantik. Misalnya, sekolah, bersekolah, disekolahkan, menyekolahkan, dan persekolahan. Kata sekolah-sekolah tidak dimasukkan ke dalam kelompok kata semantik karena mengandung karakter non-kata (-). *Homographs* juga dihapus untuk memenuhi prasyarat metode evaluasi Paice, yakni kata-kata masukan tidak mengandung duplikat.

Paice menghitung *understemming error* dan *overstemming error* menggunakan dua parameter yang disebut *understemming index* (UI) dan *overstemming index* (OI). *Understemming index* adalah bagian dari pasangan tipe 1 yang tidak berhasil digabungkan oleh algoritma *stemming*. *Overstemming index* adalah bagian dari pasangan tipe 2 yang digabungkan dengan prosedur *stemming*.

Untuk mendapat nilai *understemming index* (UI) ada perhitungan yang perlu dilakukan saat sebelum dan setelah *stemming*. Perhitungan yang dilakukan sebelum *stemming* antara lain *desired merge total* (DMT) dan *global desired merge total* (GDMT). Sedangkan perhitungan setelah *stemming* adalah *unachieved merge total* (UMT) dan *global unachieved merge total* (GUMT).

Desired Merge Total adalah penggabungan semua kata-kata dalam kelompok.

Desired Merge Total

$$DMT_g = 0.5\ N_g\ (N_g - 1) \qquad (2.2)$$

N_g = jumlah kata pada kelompok g.

Untuk kelompok yang hanya berisi 1 bentuk, nilai DMT = 0.

Global Desired Merge Total

$$GDMT = \sum_{i \in n_g} DMT_{gi} \quad (2.3)$$

n_g = jumlah total kelompok semantik.

Unachieved Merge Total adalah ketidakmampuan *stemmer* untuk menggabungkan semua kata-kata ke dalam kelompok semantik tertentu pada kata dasar yang sama.

Unachieved Merge Total

$$UMT_g = 0.5 \sum_{i \in [1..f_g]} n_{gi} (N_g - n_{gi}) \quad (2.4)$$

f_g = jumlah kata dasar yang berbeda di kelompok semantik g.

n_{gi} = jumlah kata yang sudah menjadi kata dasar i.

Global Unachieved Merge Total

$$GUMT = \sum_{i \in n_g} UMT_{gi} \quad (2.5)$$

UNIVERSITAS MULTIMEDIA NUSANTARA

Understemming Index (UI) didapat dari *Global Unachieved Merge Total* dibagi dengan *Global Desired Merge Total*.

Understemming Index

$$UI = GUMT/GDMT \qquad (2.6)$$

Perhitungan *overstemming index* (OI) dibutuhkan perhitungan dari *Wrongly-Merged Total* dan *Desired Non-Merge Total*.

Wrongly-Merged Total adalah jumlah salah penggabungan kata ke dalam kelompok kata dasar tertentu yang berisi kata dasar dari kelompok semantik berbeda.

Wrongly-Merged Total

$$WMT_s = 0.5 \sum_{i \in [1 \ldots f_s]} n_{si} (N_s - n_{si}) \qquad (2.7)$$

f_s = jumlah kata dari semantik lain

N_s = jumlah *items* dari kelompok kata dasar s.

n_{si} = jumlah kata dasar yang berasal dari kelompok semantik ke i yang asli.

UNIVERSITAS
MULTIMEDIA
NUSANTARA

Global Wrongly-Merged Total

$$GWMT = \sum_{i \in n_s} WMT_{si} \quad (2.8)$$

n_s = jumlah kelompok kata dasar.

Desired Non-Merge Total adalah kemungkinan untuk kata dalam suatu kelompok tertentu digabungkan dengan kata-kata dari kelompok semantik yang berbeda.

Desired Non-Merge Total

$$DNT_g = 0.5\ N_g\ (W - N_g) \quad (2.9)$$

W = jumlah semua kata

Global Desired Non-Merge Total

$$GDNT = \sum_{i \in [1 \ldots n_g]} DNT_{gi} \quad (2.10)$$

Overstemming Index (OI) didapat dari *Global Wrongly-Merged Total* dibagi dengan *Global Desired Non-Merge Total.*

Overstemming Index

$$OI = GWMT/GDNT \quad (2.11)$$

UNIVERSITAS MULTIMEDIA NUSANTARA

Berikut ini adalah ilustrasi untuk metode evaluasi Paice.

Tabel 2.2 Contoh kelompok kata semantik

Group	Full words
g1	sekolah bersekolah disekolahkan menyekolahkan persekolahan
g2	seko

Tabel 2.3 Hasil setelah proses *stemming*, UI = 0.6

Group	Full words	Stemmed words
g1	sekolah bersekolah disekolahkan menyekolahkan persekolahan	**seko** **seko** sekolah sekolah sekolah
g2	seko	seko

$DMT_g = 0.5\ N_g\ (N_g - 1)$

$DMT_{g1} = 0.5\ .\ 5\ (\ 5\text{-}1\ ) = 10$

$DMT_{g2} = 0$

$$GDMT = \sum_{i \in n_g} DMT_{gi}$$

$GDMT = 10 + 0 = 10$

$$UMT_g = 0.5 \sum_{i \in [1..f_g]} n_{gi} (N_g - n_{gi})$$

$UMT_{g1} = 0.5 (3(5-3) + 3(5-3)) = 6$

$UMT_{g2} = 0$

$$GUMT = \sum_{i \in n_g} UMT_{gi}$$

$GUMT = 6 + 0 = 6$

UI = GUMT/GDMT

UI = 6/10 = 0.6

Tabel 2.4 *Stem group* sebagai hasil susun ulang dari *stemmed words*, OI = 0.4

Stemmed words	**Original *Semantic* Group**
seko	**g1**
seko	**g1**
seko	g2
sekolah	g1
sekolah	g1
sekolah	g1

$$WMT_s = 0.5 \sum_{i \in [1...f_s]} n_{si} (N_s - n_{si})$$

$WMT_{s1} = 0.5 (1(3-1) + 1(3-1)) = 2$

$WMT_{s2} = 0$

UNIVERSITAS MULTIMEDIA NUSANTARA

$$GWMT = \sum_{i \in n_s} WMT_{si}$$

$GWMT = 2 + 0 = 2$

$DNT_g = 0.5\ N_g\ (W - N_g)$

$DNT_{g1} = 0.5\ .\ 5(6-5) = 2.5$

$DNT_{g2} = 0.5\ .\ 1(6-1) = 2.5$

$$GDNT = \sum_{i \in [1 \ldots n_g]} DNT_{gi}$$

$GDNT = 2.5 + 2.5 = 5$

$OI = GWMT/GDNT$

$OI = 2/5 = 0.4$

UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA